

بِسْمِ اللّٰهِ الرَّحْمٰنِ الرَّحِیْمِ

برنامه‌نویسی مقدماتی

(ویژوال بیسیک)

شاخه: کاردانش

زمینه: خدمات

گروه تحصیلی: کامپیوتر

زیرگروه: کامپیوتر

رشته مهارتی: طراحی صفحات وب، تولید چندرسانه‌ای، تصویرسازی رایانه‌ای، برنامه‌نویسی پایگاه داده

شماره رشته مهارتی: ۳-۱۷-۱۰۱-۳۱۲ تا ۳-۱۷-۱۰۱-۳۱۵

کد رایانه‌ای رشته مهارتی: ۶۱۳۷، ۶۱۳۸، ۶۱۳۹، ۶۱۴۰

نام استاندارد مهارتی مبنا: رایانه‌کار پیشرفته و برنامه‌نویسی ویژوال بیسیک مقدماتی

کد استاندارد متولی: ۸۷-۱۵/۱/۲-ف. ه و ۸۴/۸۰/۱/۳/۱-ه

شماره درس: نظری: ۴۶۸، ۴۹۰ و عملی: ۴۶۹، ۴۹۱

| | |
|----------------------|--|
| عنوان و نام پدیدآور | : برنامه‌نویسی مقدماتی (ویژوال بیسیک) [کتاب‌های درسی] ۶۱۲/۳، نظارت بر تألیف و تصویب محتوا: دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش؛ مؤلف: منصور ولی‌نژاد، وزارت آموزش و پرورش، سازمان پژوهش و برنامه‌ریزی آموزشی. |
| مشخصات نشر | : تهران: شرکت چاپ و نشر کتاب‌های درسی ایران، ۱۳۹۵. |
| مشخصات ظاهری | : ۴۴۲ ص. مصور، جدول. |
| شابک | : ۹۷۸-۹۶۴-۰۵-۲۱۸۴-۷ |
| وضعیت فهرست‌نویسی | : فیبا |
| موضوع | : ۱- ویژوال بیسیک (زبان برنامه‌نویسی کامپیوتر) - راهنمای آموزشی (متوسطه) ۲- ویژوال بیسیک (زبان برنامه‌نویسی کامپیوتر) - آزمون‌ها و تمرین‌ها (متوسطه) |
| شناسه افزوده | : ولی‌نژاد، منصور، ۱۳۴۵. الف - سازمان پژوهش و برنامه‌ریزی آموزشی. ب - دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش. ج - اداره کل نظارت بر نشر و توزیع مواد آموزشی. |
| رده‌بندی کنگره | : ۱۳۹۰ ب ۴۳ و ۹/۷۶/۷۳ QA |
| رده‌بندی دیویی | : ۶۱۲/۳ک/۳۷۳ |
| شماره کتاب‌شناسی ملی | : ۲۲۹۲۹۴۱ |

همکاران محترم و دانش آموزان عزیز :

پیشنهادات و نظرات خود را درباره محتوای این کتاب به نشانی
تهران - صندوق پستی شماره ۴۸۷۴/۱۵ دفتر تألیف کتاب‌های درسی فنی و
حرفه‌ای و کاردانش، ارسال فرمایند.

tvoccd@roshd.ir

پیام نگار (ایمیل)

www.tvoccd.medu.ir

وب‌گاه (وب‌سایت)

محتوای این کتاب در کمیسیون تخصصی رشته کامپیوتر دفتر تألیف کتاب‌های درسی فنی و
حرفه‌ای و کاردانش با عضویت : بتول عطاران، محمدرضا شکرریز، محمدرضا یمقانی، سیدحمیدرضا
ضیایی، زهرا عسگری، افشین اکبری و سیدسعید میرباقری تأیید شده است.

وزارت آموزش و پرورش سازمان پژوهش و برنامه‌ریزی آموزشی

برنامه‌ریزی محتوا و نظارت بر تألیف : دفتر تألیف کتاب‌های درسی فنی و حرفه‌ای و کاردانش

نام کتاب : برنامه‌نویسی مقدماتی (ویژوال بیسیک) - ۶۱۲/۳

مؤلف : منصور ولی‌نژاد

ویراستار ادبی : شیوا غمگسار

نظارت بر چاپ و توزیع : اداره کل نظارت بر نشر و توزیع مواد آموزشی

تهران : خیابان ایرانشهر شمالی - ساختمان شماره ۴ آموزش و پرورش (شهید موسوی)

تلفن : ۹-۸۸۸۳۱۱۶۱، دورنگار : ۸۸۳۰۹۲۶۶، کدپستی : ۱۵۸۴۷۴۷۳۵۹

وب‌سایت : www.chap.sch.ir

صفحه‌آرا : آمنه درویش

طراح جلد : بیتا اشرفی مقدم

ناشر : شرکت چاپ و نشر کتاب‌های درسی ایران : تهران - کیلومتر ۱۷ جاده مخصوص کرج - خیابان ۶۱ (داروپخش)

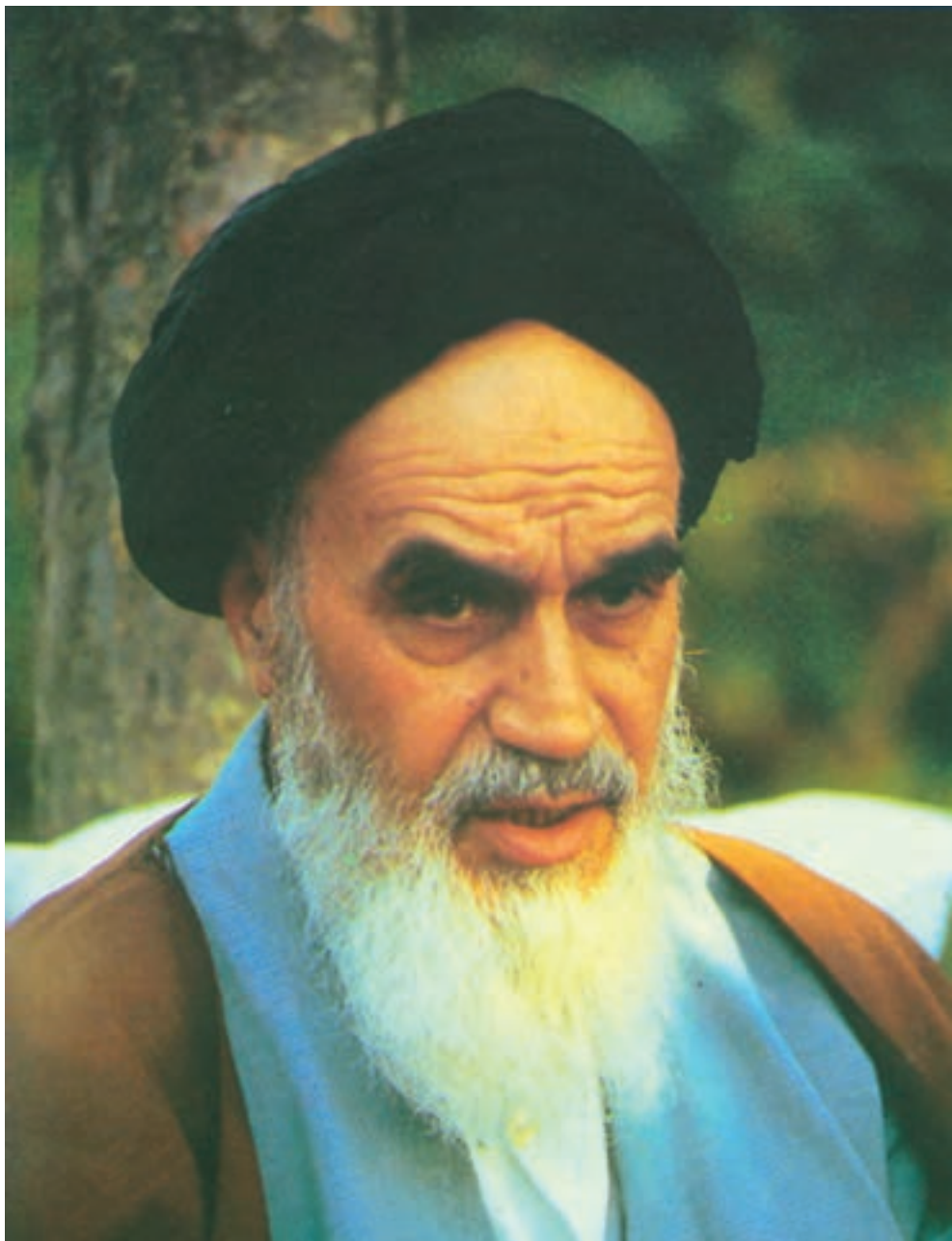
تلفن : ۵-۴۴۹۸۵۱۶۱، دورنگار : ۴۴۹۸۵۱۶۰، صندوق پستی : ۳۷۵۱۵-۱۳۹

چاپخانه : شرکت چاپ و نشر کتاب‌های درسی ایران «سهامی خاص»

سال انتشار و نوبت چاپ : چاپ چهارم ۱۳۹۵

حَقّ چاپ محفوظ است.

شابک ۹۷۸-۹۶۴-۰۵-۲۱۸۴-۷ ISBN 978-964-05-2184-7



بدانید مادام که در احتیاجات صنایع پیشرفته، دست خود را پیش دیگران دراز کنید و به در یوزگی عمر را بگذرانید، قدرت ابتکار و پیشرفت در اختراعات در شما شکوفان خواهد شد.
امام خمینی «قدس سرّه الشریف»

مجموعه کتاب‌های درسی رشته کامپیوتر شاخه کاردانش
(استاندارد وزارت فرهنگ و ارشاد اسلامی)

| رشته تصویرسازی | رشته طراحی صفحات وب | رشته تولید چندرسانه‌ای |
|------------------------------|------------------------------|------------------------------|
| مفاهیم پایه فناوری اطلاعات | مفاهیم پایه فناوری اطلاعات | مفاهیم پایه فناوری اطلاعات |
| سیستم عامل مقدماتی | سیستم عامل مقدماتی | سیستم عامل مقدماتی |
| اطلاعات و ارتباطات | اطلاعات و ارتباطات | اطلاعات و ارتباطات |
| واژه پرداز Word 2007 | واژه پرداز Word 2007 | واژه پرداز Word 2007 |
| صفحه گسترده Excel 2007 | صفحه گسترده Excel 2007 | صفحه گسترده Excel 2007 |
| ارایه مطالب PowerPoint 2007 | ارایه مطالب PowerPoint 2007 | ارایه مطالب PowerPoint 2007 |
| نرم افزارهای اداری تکمیلی | نرم افزارهای اداری تکمیلی | نرم افزارهای اداری تکمیلی |
| بانک اطلاعاتی Access 2007 | بانک اطلاعاتی Access 2007 | بانک اطلاعاتی Access 2007 |
| سیستم عامل پیشرفته | سیستم عامل پیشرفته | سیستم عامل پیشرفته |
| برنامه نویسی مقدماتی | برنامه نویسی مقدماتی | برنامه نویسی مقدماتی |
| طراحی امور گرافیکی با رایانه | طراحی امور گرافیکی با رایانه | طراحی امور گرافیکی با رایانه |
| کاربر FreeHand | کاربر Flash | کاربر Flash |
| کاربر CorelDraw | طراحی صفحات وب مقدماتی | کاربر Director |
| | طراحی صفحات وب پیشرفته | میکس رایانه‌ای |

مجموعه کتاب‌های درسی رشته کامپیوتر شاخه کاردانش
(استاندارد وزارت کار و امور اجتماعی)

| رشته تصویرسازی | رشته طراحی صفحات وب | رشته تولید چندرسانه‌ای | رشته برنامه‌نویسی پایگاه داده |
|-----------------------------|-----------------------------|------------------------------|---|
| مفاهیم پایه فناوری اطلاعات | مفاهیم پایه فناوری اطلاعات | مفاهیم پایه فناوری اطلاعات | مفاهیم پایه فناوری اطلاعات |
| سیستم‌عامل مقدماتی | سیستم‌عامل مقدماتی | سیستم‌عامل مقدماتی | سیستم‌عامل مقدماتی |
| اطلاعات و ارتباطات | اطلاعات و ارتباطات | اطلاعات و ارتباطات | اطلاعات و ارتباطات |
| سیستم‌عامل پیشرفته | سیستم‌عامل پیشرفته | سیستم‌عامل پیشرفته | سیستم‌عامل پیشرفته |
| واژه‌پرداز Word 2007 | واژه‌پرداز Word 2007 | واژه‌پرداز Word 2007 | واژه‌پرداز Word 2007 |
| صفحه گسترده Excel 2007 | صفحه گسترده Excel 2007 | صفحه گسترده Excel 2007 | صفحه گسترده Excel 2007 |
| ارایه مطالب PowerPoint 2007 | ارایه مطالب PowerPoint 2007 | ارایه مطالب PowerPoint 2007 | ارایه مطالب PowerPoint 2007 |
| نرم‌افزارهای اداری تکمیلی | نرم‌افزارهای اداری تکمیلی | نرم‌افزارهای اداری تکمیلی | نرم‌افزارهای اداری تکمیلی |
| طراح گرافیک رایانه‌ای | طراح گرافیک رایانه‌ای | طراح گرافیک رایانه‌ای | برنامه‌نویسی مقدماتی |
| شهروند الکترونیکی | نرم‌افزار گرافیکی Flash Mx | شهروند الکترونیکی | برنامه‌نویسی ویژوال بیسیک پیشرفته (جلد اول) |
| نرم‌افزار گرافیکی FreeHand | طراحی مقدماتی صفحات وب | نرم‌افزار گرافیکی Director | برنامه‌نویسی ویژوال بیسیک پیشرفته (جلد دوم) |
| نرم‌افزار گرافیکی CorelDraw | رایانه کار Interdev | تدوین فیلم و صدا SSP | مدیریت پایگاه داده |
| نرم‌افزار گرافیکی Flash Mx | رایانه کار Dreamweaver | نرم‌افزار گرافیکی Flash Mx | مهارت عمومی برنامه‌نویسی |
| | رایانه کار CIW | نرم‌افزار گرافیکی Authorware | |

فهرست مطالب

| | |
|----|--|
| ۲ | مقدمه |
| ۳ | پیش‌آزمون |
| | واحد کار ۱: توانایی حل مسایل و طراحی الگوریتم مناسب برای آنها |
| ۶ | کلیات |
| ۶ | ۱- شناخت مسایل و ارایه راه‌حل مناسب برای حل آنها |
| ۸ | ۲-۱ الگوریتم |
| ۳۱ | واژه‌نامه |
| ۳۱ | خلاصه مطالب |
| ۳۲ | آزمون نظری |
| ۳۵ | آزمون عملی |
| | واحد کار ۲: توانایی ترسیم فلوچارت |
| ۳۸ | کلیات |
| ۳۸ | ۱-۲ علایم و اشکال در فلوچارت |
| ۴۷ | واژه‌نامه |
| ۴۸ | آزمون نظری |
| ۵۲ | آزمون عملی |
| | واحد کار ۳: توانایی درک و شناخت زبان برنامه‌نویسی ویژوال بیسیک و ایجاد یک برنامه کاربردی |
| ۵۴ | کلیات |
| ۵۴ | ۱-۳ تقسیم‌بندی زبان‌های برنامه‌نویسی |
| ۵۸ | ۲-۳ اجزای تشکیل‌دهنده یک برنامه |
| ۵۹ | ۳-۳ نحوه اجرای برنامه‌ویژوال بیسیک و معرفی اجزای موجود در آن |
| ۷۲ | ۴-۳ برنامه‌نویسی رویدادگرا |
| ۷۳ | ۵-۳ کنترل دکمه فرمان (Command Button) |
| ۷۷ | واژه‌نامه |
| ۷۸ | خلاصه مطالب |
| ۷۹ | آزمون نظری |

| | |
|-----|---|
| ۸۱ | آزمون عملی |
| | واحد کار ۴: توانایی تعریف انواع متغیرها، ثابت‌ها و استفاده از عملگرهای ریاضی ورشته‌ای |
| ۸۴ | کلیات |
| ۸۴ | ۴-۱ نحوه تعریف و استفاده از انواع متغیرها در ویژوال بیسیک |
| ۹۴ | ۴-۲ نحوه انجام عملیات ریاضی در ویژوال بیسیک |
| ۹۶ | ۴-۳ نحوه تعریف و استفاده از ثابت‌ها در ویژوال بیسیک |
| ۹۹ | ۴-۴ متغیرهای ایستا، محلی و عمومی در ویژوال بیسیک |
| ۱۰۰ | ۴-۵ عملگرهای رشته‌ای در ویژوال بیسیک |
| ۱۰۶ | واژه‌نامه |
| ۱۰۶ | خلاصه مطالب |
| ۱۰۷ | آزمون نظری |
| ۱۰۸ | آزمون عملی |
| | واحد کار ۵: توانایی استفاده از دستور شرطی IF و عملگرهای مقایسه‌ای و منطقی |
| ۱۱۲ | کلیات |
| ۱۱۲ | ۵-۱ نحوه استفاده از ساختار IF و عملگرهای مقایسه‌ای در برنامه‌ها |
| ۱۱۵ | ۵-۲ کنترل کادر تصویر (Picture Box) |
| ۱۲۶ | ۵-۳ نحوه استفاده از عملگرهای منطقی برای ترکیب شرطها |
| ۱۲۹ | ۵-۴ کنترل تصویر |
| ۱۳۲ | ۵-۵ اولویت اجرای عملگرها نسبت به یکدیگر |
| ۱۳۲ | ۵-۶ نحوه استفاده از کادرهای پیام در ویژوال بیسیک |
| ۱۳۶ | ۵-۷ نحوه استفاده از کادرهای ورود داده در ویژوال بیسیک |
| ۱۳۸ | ۵-۸ کنترل کادر علامت CheckBox |
| ۱۳۹ | ۵-۹ معرفی مهم‌ترین ویژگی‌های فرم‌ها و کنترل‌های کادر متن، دکمه فرمان و غیره |
| ۱۴۳ | واژه‌نامه |

| | |
|-----|--|
| ۱۴۳ | خلاصه مطالب |
| ۱۴۵ | آزمون نظری |
| ۱۴۷ | آزمون عملی |
| | واحد کار ۶: توانایی استفاده از انواع حلقه‌ها و ساختار Select Case و کنترل دکمه انتخاب |
| ۱۴۹ | کلیات |
| ۱۵۰ | ۱-۶ دستور Select Case |
| ۱۵۶ | ۲-۶ ساختارهای تکرار در ویژوال بیسیک |
| ۱۷۰ | واژه‌نامه |
| ۱۷۰ | خلاصه مطالب |
| ۱۷۲ | آزمون نظری |
| ۱۷۶ | آزمون عملی |
| | واحد کار ۷: توانایی ایجاد و استفاده از انواع رویه‌ها در ویژوال بیسیک |
| ۱۷۸ | کلیات |
| ۱۷۸ | ۱-۷ رویه‌های فرعی (Sub Procedure) |
| ۱۸۴ | ۲-۷ رویه‌های تابعی (Function Procedure) |
| ۱۸۸ | ۳-۷ روش‌های ارسال مقادیر به رویه‌های فرعی و تابعی |
| ۱۹۲ | ۴-۷ نحوه استفاده از نام آرگومان‌ها در رویه‌ها |
| ۱۹۴ | ۵-۷ خروج از یک رویه با استفاده از دستورات Exit Sub و Exit Function |
| ۱۹۵ | ۶-۷ رویه‌های محلی و عمومی |
| ۲۰۳ | ۷-۷ رویه‌های رویداد (vent Procedure) |
| ۲۰۷ | واژه‌نامه |
| ۲۰۷ | خلاصه مطالب |
| ۲۰۸ | آزمون نظری |
| ۲۱۰ | آزمون عملی |
| | واحد کار ۸: توانایی استفاده از انواع رویه‌های آماده در ویژوال بیسیک |

| | |
|-----|---|
| ۲۱۲ | کلیات |
| ۲۱۲ | ۸-۱ توابع رشته‌ای و ویژوال بیسیک |
| ۲۲۴ | ۸-۲ توابع تاریخ و ساعت |
| ۲۳۳ | ۸-۳ کنترل کادر در لیست |
| ۲۳۷ | واژه‌نامه |
| ۲۳۸ | خلاصه مطالب |
| ۲۳۹ | آزمون نظری |
| ۲۴۰ | آزمون عملی |
| | واحد کار ۹: نحوه استفاده از رویدادهای ماوس و صفحه کلید |
| ۲۴۲ | کلیات |
| ۲۴۲ | ۹-۱ رویدادهای ماوس |
| ۲۴۶ | ۹-۲ کنترل خط (Line) |
| ۲۴۷ | ۹-۳ کنترل شکل (Shape) |
| ۲۴۸ | ۹-۴ رویدادهای صفحه کلید |
| ۲۵۶ | واژه‌نامه |
| ۲۵۶ | خلاصه مطالب |
| ۲۵۷ | آزمون نظری |
| ۲۵۸ | آزمون عملی |
| | واحد کار ۱۰: نحوه ایجاد انواع منو در ویژوال بیسیک |
| ۲۶۰ | کلیات |
| ۲۶۰ | ۱۰-۱ نحوه طراحی انواع منو در ویژوال بیسیک |
| ۲۶۸ | مطالعه آزاد |
| ۲۶۸ | ۱۰-۲ نحوه ایجاد و استفاده از رابط گرافیکی چند سندی یا MDI |
| ۲۷۳ | ۱۰-۳ کنترل‌های نوار پیمایش افقی و عمودی |
| ۲۷۹ | ۱۰-۴ کنترل کادر محاوره (CommonDialog) |
| ۲۸۴ | واژه‌نامه |
| ۲۸۴ | خلاصه مطالب |

| | |
|-----|---|
| ۲۸۵ | آزمون نظری |
| ۲۸۶ | آزمون عملی |
| | واحد کار ۱۱: توانایی استفاده از انواع آرایه‌ها در ویژوال بیسیک |
| ۲۸۸ | کلیات |
| ۲۸۹ | ۱۱-۱ تعریف انواع آرایه در ویژوال بیسیک |
| ۳۰۰ | ۱۱-۲ آرایه‌های چند بعدی |
| ۳۰۳ | ۱۱-۳ توابع UBound و LBound |
| ۳۰۴ | ۱۱-۴ تابع Split |
| ۳۰۷ | ۱۱-۵ تابع Join |
| ۳۰۹ | ۱۱-۶ نحوه ارسال آرایه‌ها به رویه‌ها |
| ۳۱۰ | ۱۱-۷ روش‌های مرتب‌سازی آرایه‌ها |
| ۳۱۳ | ۱۱-۸ روش‌های جستجوی داده‌ها در آرایه‌ها |
| ۳۱۹ | واژه‌نامه |
| ۳۱۹ | خلاصه مطالب |
| ۳۲۱ | آزمون نظری |
| ۳۲۲ | آزمون عملی |
| | واحد کار ۱۲: توانایی استفاده از جلوه‌های گرافیکی و چاپ در ویژوال بیسیک |
| ۳۲۴ | کلیات |
| ۳۲۴ | ۱۲-۱ مفهوم سیستم مختصات در ویژوال بیسیک |
| ۳۲۵ | ۱۲-۲ تغییر سیستم مختصات |
| ۳۳۲ | ۱۲-۳ خصوصیات و متدهای گرافیکی |
| ۳۵۴ | ۱۲-۴ تابع QBcolor |
| ۳۵۵ | ۱۲-۵ تابع RGB |
| ۳۵۶ | ۱۲-۶ شیء چاپگر (Printer Object) |
| ۳۶۱ | ۱۲-۷ چندرسانه‌ای (Multimedia) |
| ۳۶۸ | ۱۲-۸ شیء تصویر (Picture) |

| | |
|-----|---|
| ۳۷۲ | واژه‌نامه |
| ۳۷۲ | خلاصه مطالب |
| ۳۷۴ | آزمون نظری |
| ۳۷۶ | آزمون عملی |
| | واحد کار ۱۳: توانایی انجام یک پروژه عملی |
| ۴۰۱ | آزمون پایانی |
| ۴۰۶ | پاسخنامه |
| ۴۰۹ | ضمائم |
| ۴۴۲ | فهرست منابع |

مقدمه

یکی از عمده‌ترین اهداف طراحی و تولید کامپیوترها انجام عملیات ذخیره‌سازی، بازیابی داده‌ها و اطلاعات و انجام انواع محاسبات به وسیله آن‌هاست. برای تحقق بخشیدن به این اهداف وجود دو جزء اصلی یعنی سخت‌افزار و نرم‌افزار الزامی است.

با پیدایش اولین کامپیوترها نیاز به وجود برنامه‌هایی که بتوان با به‌کارگیری آن‌ها کامپیوترها را مورد استفاده قرار داد، احساس شد و از آن‌جا که پردازش در کامپیوترها براساس مبنای باینری یا همان ۰ و ۱ است، اولین نرم‌افزارهایی که توسط متخصصین طراحی گردید به زبان ماشین (۰ و ۱) نوشته شد. این روش تولید نرم‌افزار کار دشوار و وقت‌گیری بود که محدودیت‌های زیادی را در برداشت، بنابراین متخصصان علوم کامپیوتر تصمیم گرفتند تا نرم‌افزارهایی را تولید کنند که بتوانند با استفاده از آن‌ها هر نوع برنامه‌ای را با سرعت و دقت به زبان ماشین تبدیل نمایند. این امر منجر به تولید شاخه ویژه‌ای از نرم‌افزارها به نام زبان‌های برنامه‌نویسی شد. بدین ترتیب به موازات رشد و تکامل صنعت سخت‌افزار، زبان‌های برنامه‌نویسی کامپیوتر نیز خط سیر تکاملی خود را از زبان ماشین و اسمبلی به زبان‌های برنامه‌نویسی سطح بالا، ساخت یافته، شیء‌گرا و ویژوال، طی کرده و هر روزه زبان‌های برنامه‌نویسی کاربردی‌تری را در اختیار برنامه‌نویسان قرار دادند. در حال حاضر محدوده زبان‌های برنامه‌نویسی بسیار گسترده شده و با ظهور سیستم‌عامل‌های ویندوز و رایج‌شدن شبکه‌های کامپیوتری بخصوص اینترنت، این مسأله شدت بیشتری پیدا کرده است.

این پیمانه مهارتی با توجه به نیازهای آموزشی در رابطه با روش حل مسایل و زبان‌های برنامه‌نویسی سطح بالا و مدرن که قابلیت برنامه‌نویسی در محیط سیستم عامل ویندوز را نیز داشته باشند، تألیف شده است. زبان برنامه‌نویسی ویژوال بیسیک نسخه ۶ یکی از رایج‌ترین و کارآمدترین زبان‌های برنامه‌نویسی در دنیاست که از ویژگی‌های بالایی در برنامه‌نویسی حرفه‌ای برخوردار است. این پیمانه مهارتی شامل چهارده واحدکار است که به صورت خودآموز تهیه شده است. در واحدکار اول و دوم دانش‌آموزان با مفاهیم اساسی درباره الگوریتم و فلوچارت آشنا شده و چگونگی یافتن راه‌حل مناسب برای حل مسایل مختلف را به همراه مثال و تمرین‌های متعدد فرامی‌گیرند. از واحد کار سوم تا چهاردهم فراگیر می‌تواند مطالب را به صورت تئوری همراه با تصاویر مناسب مطالعه کرده و سپس مطالب مطالعه شده را با تمرین‌های مرتبط و به صورت عملی و مرحله‌ای انجام دهد. به علاوه سعی شده است که با ارایه آزمون‌های نظری و عملی در پایان هر واحدکار و یک آزمون نظری و عملی در پایان پیمانه مهارتی، مطالب مجدداً مرور شوند. برای بازدهی بالاتر در فراگیری این پیمانه مهارتی تسلط کافی در دروس پیش‌نیاز مانند مبانی و فناوری کامپیوتر، سیستم عامل مقدماتی و پیشرفته (ترجیحاً Windows XP) الزامی است.

مؤلف

Vbbook@yahoo.com

توانایی حل مسایل و طراحی الگوریتم مناسب برای آنها

هدفهای رفتاری

- ۱- نحوه شناخت و بررسی مسایل مختلف را توضیح دهد.
- ۲- مفهوم الگوریتم را تعریف کند و ویژگی‌های آن را بیان نماید.
- ۳- انواع دستورالعمل‌ها را در الگوریتم بیان کند و کاربرد آنها را توضیح دهد.
- ۴- مفهوم عملگر را بداند و انواع عملگرها را بیان کند.
- ۵- عملگرهای ریاضی، منطقی و مقایسه‌ای و کاربرد آنها را بیان کند و حق تقدم آنها را نسبت به یکدیگر توضیح دهد.
- ۶- بتواند انواع الگوریتم‌ها را برای مسایل متفاوت طراحی نماید.

کلیات

روش حل مسایل با استفاده از روش‌ها و تحلیل‌های ریاضی و منطقی، اولین بار به وسیله دانشمند بزرگ ایرانی - خوارزمی - مورد توجه قرار گرفت، وی علاوه بر مباحث مختلفی که در علوم ریاضی و نجوم طرح کرد، اساس روشی را در حل مسایل بنا نهاد که در آینده مدت‌ها مورد استفاده برنامه‌نویسان کامپیوتر قرار گرفت. نام الگوریتم نیز به افتخار وی و از عبارت «الخوارزمی» گرفته شده است.

در این پیمان‌ه مهارتی با نحوه تجزیه و تحلیل انواع مختلف مسایل و جستجو و طراحی راه حل‌های مناسب برای آن‌ها آشنا خواهید شد و در پایان ارایه راه‌حل به دست آمده با توجه به قواعد علم الگوریتم را می‌آموزید.

۱-۱ شناخت مسایل و ارایه راه‌حل مناسب برای حل آن‌ها

انسان از آغاز آفرینش تاکنون همواره در مسیر زندگی خود با مشکلات و مسائل مختلفی روبه‌رو بوده و برای حل مشکلات خود، همواره راه‌حل‌های متفاوتی را تجربه کرده است. از مشکلاتی نظیر خوراک، پوشاک، محل زندگی تا حل مسائل علمی، فلسفی، ریاضی و نظایر آن‌ها؛ بنابراین با توجه به نوع مسایل و تجربیات و پیشرفت‌های علمی، سعی در ارائه راه‌حل‌های جدید کرده است. پیدا کردن راه‌حل برای یک مسأله به نوع آن بستگی دارد، بعضی از مسأله‌ها به سادگی قابل حل بوده اما بعضی دیگر هنوز هم به سادگی قابل حل نیستند.

برای حل هرگونه مسأله جدا از نوع آن می‌توان موارد زیر را در نظر گرفت:

۱- شناخت دقیق مسأله

۲- تجزیه و تحلیل مسأله

۳- طراحی راه‌حل

۱-۱-۱ شناخت مسأله

برای شناخت بهتر یک مسأله باید سه عامل مهم را در نظر بگیرید: مقادیر معلوم، خواسته‌های مسأله (مجهولات) و عملیات محاسباتی.

مقادیر معلوم داده‌ها: مقادیری که در اختیار مسأله قرار می‌گیرند و برای رسیدن به

هدف موردنظر در مسأله موردنیاز هستند.

ارتباط بین داده‌ها و مجهول‌ها (محاسبات): برای رسیدن به نتایج موردنظر معمولاً لازم است تا عملیاتی را روی مقادیر معلوم انجام دهید؛ بخش عمده‌ای از این عملیات با استفاده از فرمول‌های مختلف انجام می‌شود؛ البته محاسبات می‌توانند با توجه به روابط منطقی که بین مقادیر معلوم و خواسته‌های مسأله وجود دارند، انجام گیرند.

خواسته‌های مسأله (مجهولات): مقادیری هستند که معمولاً در اثر انجام عملیات روی مقادیر معلوم حاصل می‌شوند، البته مجهولات می‌توانند از روابط منطقی که در حل مسأله دخالت می‌کنند نیز به وجود آمده و مورد استفاده قرار گیرند.

به عنوان مثال فرض کنید می‌خواهیم محیط یک دایره به شعاع دلخواه را محاسبه کنیم. برای حل این مسأله با روش ارایه شده، ابتدا مقادیر معلوم را مورد توجه قرار می‌دهیم؛ همان‌طور که می‌دانید برای محاسبه محیط هر دایره باید شعاع آن را در اختیار داشته باشیم، بنابراین شعاع دایره (R) به عنوان تنها داده موردنیاز برای حل مسأله کافی است.

محاسباتی که برای رسیدن به محیط دایره لازم است، در واقع فرمول زیر خواهد بود:

$$P=2 \times 3/14 \times R$$

با استفاده از این فرمول‌ها روابط بین داده‌های ورودی و نتایج موردنظر به خوبی تعیین می‌شود و بالاخره به عنوان مرحله آخر، خواسته مسأله (مجهول) که همان محیط (P) دایره است به دست می‌آید.



تمرین:

مقادیر معلوم، محاسبات و خواسته‌های مسأله (مجهولات) را در محاسبه محیط و مساحت یک مستطیل دلخواه مشخص کنید

۲-۱-۱ تجزیه و تحلیل مسأله

معمولاً جستجوی راه‌حل مناسب برای یک مسأله، به‌سادگی امکان‌پذیر نیست. گاهی اوقات راه‌حل مسأله به‌سادگی قابل تشخیص است، مانند محاسبه مجموع دو عدد یا محاسبه مساحت یک دایره اما در بعضی از مواقع به‌دلیل پیچیده بودن مسأله، لازم است

است مسأله اصلی به چند مسأله کوچکتر تقسیم شود که به آن‌ها زیرمسأله می‌گویند. سپس هر زیرمسأله مجدداً بررسی شود تا در صورت دشواری مجدداً به زیرمسأله‌های کوچکتر تقسیم شود و این عمل آنقدر ادامه یابد تا زیرمسأله‌ها به ساده‌ترین شکل حل شوند. در این حالت می‌توان هر زیرمسأله را جداگانه حل کرد و با کنار هم قرار دادن آن‌ها، به حل مسأله اصلی دسترسی پیدا کرد. به عنوان مثال اگر مسأله ساخت یک اتومبیل باشد پیدا کردن راه‌حل برای ساخت آن در مرحله اول، کار دشواری است اما اگر اتومبیل را به بخش‌های مختلف مثل موتور، جعبه‌دنده و اتاق و بدنه تقسیم کنیم و هر یک از این اجزا را مجدداً به کوچک‌ترین جزء، آن‌گاه با طراحی و ساخت اجزای کوچکتر می‌توان اجزای بزرگتر را ایجاد کرد و در نهایت با ترکیب اجزای اصلی، طراحی و ساخت یک اتومبیل امکان‌پذیر خواهد شد.

۳-۱-۱ طراحی راه‌حل

پس از تحلیل مسأله، برای ارائه یک راه‌حل می‌توان به دو روش عمل کرد:

- ۱- استفاده از تجربیات و راه‌حل‌های موجود که در حل مسائل دیگر به کار گرفته شده‌اند.
- ۲- استفاده از روش‌های تفکر منطقی و الگوریتمی که حل مسأله براساس آن صورت گرفته و به صورت مرحله به مرحله انجام می‌شود. به عنوان مثال فرض کنید می‌خواهیم تعداد اعداد زوج بین اعداد ۱ تا ۲۰ را محاسبه کنیم، یک روش ساده این است که اعداد ۱ تا ۲۰ را نوشته و تعداد اعداد زوج را بشمارید، اما با استفاده از روش تفکر منطقی می‌توان به این نتیجه رسید که با توجه به این‌که اعداد زوج یکی در میان قرار می‌گیرند با تقسیم عدد ۲۰ بر ۲ به تعداد ۱۰ عدد دست می‌یابید که از روش اول هم به همین نتیجه خواهید رسید.

۲-۱ الگوریتم

۳-۱-۲ تعریف الگوریتم

به مجموعه‌ای از دستورالعمل‌ها که با زبان دقیق و قابل فهم به همراه جزئیات لازم و به صورت مرحله به مرحله به گونه‌ای اجرا شده که هدف خاصی (حل مسأله) را دنبال کنند و شروع و خاتمه آن‌ها نیز مشخص باشد، الگوریتم می‌گویند.

می‌توان الگوریتم را به یک ماشین تشبیه کرد که مقادیر معلوم را دریافت کرده، روی آن‌ها محاسباتی را انجام می‌دهد و در پایان خواسته‌های مسأله (مجهولات) را ارایه می‌کند.

در واقع رابطه نزدیکی بین مفهوم الگوریتم و نحوه کار کامپیوتر در حل مسایل وجود دارد، بنابراین با استفاده از روش الگوریتم می‌توانید حل مسایل را به گونه‌ای طراحی کنید که برای تبدیل به زبان کامپیوتر نیز قابل فهم باشد. به طور معمول با مفهوم الگوریتم آشنا هستید و از آن استفاده می‌کنید، به عنوان مثال وقتی هر روز برای کسب علم و دانش به مدرسه می‌روید، اعمالی را به ترتیب و به صورت دقیق و کامل انجام می‌دهید یعنی ابتدا از خواب بیدار می‌شوید، دست و صورت خود را می‌شوید، صبحانه می‌خورید و سپس لباس مناسب به تن کرده و بعد از برداشتن وسایل و کتاب‌های موردنیاز از خانه خارج می‌شوید، مسافتی را طی کرده و به مدرسه و کلاس خود می‌روید. اگر هر یک از این اعمال را قبل یا بعد از عمل دیگر انجام دهید، هدف موردنظر که شکل درست به مدرسه رفتن است، انجام نمی‌شود.

می‌توان این مراحل را به شکل خلاصه و قابل فهم‌تری بیان کرد:

- بیدار شدن از خواب
- شستن دست و صورت
- خوردن صبحانه
- پوشیدن لباس مدرسه
- برداشتن کتاب‌های درسی و دفتر و قلم
- خروج از خانه
- رفتن به مدرسه و ورود به کلاس درس

تمرین:



مراحل تهیه نان در نانوائی را به صورت مرحله به مرحله بنویسید.

۲-۱-۲ شرایط الگوریتم

پس از آشنایی با مفهوم الگوریتم، لازم است ویژگی های یک الگوریتم را بشناسید.
الف- استفاده از زبان ساده، دقیق و قابل فهم: این ویژگی سبب می شود تا در انجام دستورالعمل ها همواره یک برداشت یکسان حاصل شود، در غیر این صورت برداشت های متفاوت سبب خواهد شد تا دستورالعمل ها نتایج متفاوتی را به وجود آورند. زبان الگوریتم نیز می تواند یکی از زبان های گفتاری و نوشتاری مانند فارسی، انگلیسی و ... باشد.

ب- استفاده از جزئیات کافی: این ویژگی سبب می شود تا دستورالعمل ها به طور کامل اجرا شوند. وجود موارد نامشخص یا ارایه دستورالعمل ها به صورت کلی و مبهم سبب مخدوش شدن نتایج خواهد شد.

ج- شروع و پایان الگوریتم: در یک الگوریتم باید شروع دستورالعمل ها مشخص باشد. هر الگوریتم یک نقطه شروع دارد که به عنوان اولین دستورالعمل از آن استفاده می شود، به علاوه پایان الگوریتم نیز باید تعیین شود. یک الگوریتم می تواند بیش از یک نقطه پایان داشته باشد.

د- ترتیب انجام دستورالعمل ها: یکی از ویژگی های مهم یک الگوریتم ترتیب اجرای دستورالعمل ها است؛ اگر این کار به درستی انجام نشود، پیش بینی نتیجه کار مشخص نخواهد بود. در یک الگوریتم ترتیب انجام عملیات با استفاده از شماره گذاری دستورالعمل ها از بالا به پایین انجام می شود که البته در صورت نیاز می توان ترتیب اجرای دستورالعمل ها را نیز تغییر داد. در مباحث بعد به این مسأله خواهیم پرداخت.

نکته: اگر در حل مسایل سه عامل اصلی را به دقت مشخص کنید، طراحی یک الگوریتم کار زیاد دشواری نخواهد بود. اگر مقادیر معلوم، خواسته های مسأله، فرمول ها و روابط ریاضی و منطقی بین آن ها را به درستی تعیین کنید، نوشتن الگوریتم های مختلف آسان تر خواهد شد.

ه- جامع بودن: الگوریتم باید به شکلی طراحی شود که با توجه به صورت مسأله و مفروضات آن در تمام حالت ها، نتایج مناسب و صحیحی را ارایه کرده و در حالت های خاص یا داده های ورودی متفاوت، نتایج درستی را ایجاد کند.



نکته: بهتر است در یک الگوریتم از دستورات اضافه که سبب افزایش حجم الگوریتم می‌شود، خودداری نمایید چرا که این کار الگوریتم را شلوغ کرده و باعث سردرگمی می‌شود.

۳-۲-۱ انواع دستورالعمل‌ها در الگوریتم

الگوریتم مجموعه‌ای از دستورالعمل‌هاست. دستورالعمل‌ها انواع مختلفی دارند که عبارتند از:

- الف- دستورالعمل‌های ورودی
- ب- دستورالعمل‌های خروجی
- ج- دستورالعمل‌های محاسباتی
- د- دستورالعمل‌های شرطی
- هـ - دستورالعمل‌های تکرار (حلقه‌ها)

الف- دستورالعمل‌های ورودی

این دستورالعمل‌ها برای دریافت داده‌های ورودی استفاده می‌شوند و معمولاً برای اجرای آن‌ها از عباراتی مانند «بخوان، دریافت کن یا بگیر» استفاده می‌شود.

ب- دستورالعمل‌های خروجی

این دستورالعمل‌ها برای نمایش اطلاعات خروجی یا پیام‌های موردنیاز به منظور راهنمایی کاربر روی صفحه نمایش یا چاپ آن‌ها به وسیله چاپگر استفاده می‌شوند و معمولاً برای اجرای آن‌ها عباراتی مانند «نمایش بده یا چاپ کن» به کار می‌روند.

ج- دستورالعمل‌های محاسباتی

این دستورالعمل‌ها در واقع نحوه‌ارایه و استفاده از فرمول‌ها و انجام عملیات ریاضی و محاسباتی را تعیین می‌کنند و معمولاً برای اجرای آن‌ها از همان شکلی که در ریاضیات وجود دارد، استفاده می‌شود یعنی در سمت راست تساوی عملیات محاسباتی و در سمت چپ تساوی نام یک متغیر به کار می‌رود؛ البته به جای علامت تساوی از علامت فلش نیز استفاده می‌شود.

تعریف متغیر: متغیرها مکان‌هایی هستند که توانایی نگهداری و ذخیره‌سازی انواع داده را دارند. متغیرها در الگوریتم همان کاربرد ریاضی خود را دارا هستند و علاوه بر اعداد می‌توانند متن یا مقادیر منطقی درست یا نادرست، تاریخ، ساعت و نظایر آن‌ها را نگهداری کنند.

نحوه استفاده از عملگرهای ریاضی در الگوریتم: برخی از عملیاتی که در الگوریتم انجام می‌گیرد، عملیات ریاضی و محاسباتی است که برای انجام این گونه دستورات لازم است از عملگرهای ریاضی استفاده شود.

تعریف عملگر: در واقع عملگر یک یا مجموعه عملیاتی است که از پیش در الگوریتم آماده شده است و برای استفاده از آن باید از علایمی که به صورت قراردادی تعریف شده‌اند، استفاده کنید مانند عملگر جمع که می‌تواند دو عدد یا دو متغیر یا ترکیبی از آن‌ها را با هم جمع و حاصل آن را ارایه کند. اعداد و متغیرهایی که عملگر روی آن‌ها عملیات انجام می‌دهد، عملوند نامیده می‌شوند. عملگرهای ریاضی در جدول ۱-۱ ارایه شده‌اند.

جدول ۱-۱

| اولویت | مثال | عنوان | عملگر |
|--------|-------------------------|-----------------------|-------|
| ۱ | $(2 \times 3) + 4 = 10$ | پرانتز | () |
| ۲ | $2 \times 3 = 6$ | ضرب | * |
| ۲ | $12 / 6 = 2$ | تقسیم | / |
| ۳ | $17 \setminus 3 = 5$ | خارج قسمت تقسیم صحیح | \ |
| ۴ | $14 \bmod 5 = 4$ | باقی‌مانده تقسیم صحیح | Mod |
| ۵ | $35 + 14 = 49$ | جمع | + |
| ۵ | $10 - 7 = 3$ | تفریق | - |

عملگرهای ارایه شده در جدول ۱-۱ نسبت به هم دارای حق تقدم در اجرا هستند، به عبارت دیگر در یک عبارت ریاضی که از چند عملگر استفاده شده است، عملگرها

از سمت چپ عبارت ریاضی به ترتیب اجرا می‌شوند. اما همیشه این‌گونه نیست و در هنگام استفاده از چند عملگر ریاضی، اولویت آن‌ها مطابق جدول ۱-۱ از بالا به پایین در نظر گرفته می‌شود، به عنوان مثال عبارت $10/5 \times 3 + 2$ را در نظر بگیرید؛ در این عبارت ابتدا عدد ۳ در ۵ ضرب شده و سپس حاصل ضرب آن یعنی ۱۵، بر عدد ۱۰ تقسیم می‌شود چرا که عملگرهای ضرب و تقسیم نسبت به عملگر جمع حق تقدم دارند. در پایان، حاصل تقسیم یعنی عدد $10/5$ با عدد ۲ جمع و در نتیجه عدد $3/5$ به دست می‌آید.

توجه کنید که عملگرهای ضرب و تقسیم نسبت به هم، جمع و تفریق نیز نسبت به هم دارای اولویت یکسانی هستند و اولویت آن‌ها نسبت به هم از چپ به راست است. به عنوان مثال عبارت $2 \times 6 - 4 + 10$ را در نظر بگیرید؛ در این عبارت ابتدا عدد ۶ در ۲ ضرب شده، سپس عبارت $10 + 4$ محاسبه می‌شود که نتیجه آن عدد ۵۰ خواهد بود و بعد عمل تفریق انجام می‌شود و حاصل تفریق $12 - 50$ یعنی ۳۸ به دست می‌آید.

البته اولویت عملگرها را می‌توان با پرانتز تغییر داد به عنوان مثال اگر عبارت قبل را با پرانتز به شکل $2 \times (6 - 4 + 10)$ تغییر دهید ابتدا عبارت داخل پرانتز محاسبه می‌شود سپس نتیجه در عدد ۲ ضرب خواهد شد.

تمرین:



ترتیب انجام عملیات و حاصل عبارات زیر را تعیین کنید:

$$(2 + 5) / (2 \times 3 - 14)$$

$$2 / (8 + (12 - 7) \times (4 + 27))$$




مثال ۱: فرض کنید می‌خواهید مسأله محاسبه محیط یک مستطیل با طول L و عرض W دلخواه را به صورت یک الگوریتم بنویسید، قبلاً این مسأله را بررسی کرده و سه عامل مقادیر معلوم، فرمول‌های محاسباتی و خواسته‌های مسأله (مجهولات) و رابطه بین آن‌ها را تعیین کرده‌اید، بنابراین الگوریتم حل این مسأله به صورت زیر خواهد بود:

| توضیح | |
|----------------|------------------------------------|
| شروع الگوریتم | ۱- شروع الگوریتم |
| ورود داده‌ها | ۲- L را دریافت کن |
| ورود داده‌ها | ۳- W را دریافت کن |
| محاسبه محیط | ۳- $P \leftarrow 2 \times (L + W)$ |
| نمایش خروجی | ۴- P را نمایش بده |
| پایان الگوریتم | ۵- پایان |

تمرین: 

الگوریتمی بنویسید که محیط یک دایره باشعاع دلخواه R را محاسبه کرده و نمایش دهد.

مثال ۲: الگوریتمی بنویسید تا حاصل تقسیم یک عدد دلخواه را بر عدد دلخواه دیگری محاسبه کرده و نمایش دهد. 

در این مثال ورودی‌ها شامل دو عدد هستند و محاسبات لازم برای رسیدن به نتیجه، عمل ریاضی تقسیم می‌باشد و خواسته مسأله نیز نتیجه تقسیم دو عدد بر یکدیگر است، بنابراین الگوریتم مربوطه بدین صورت خواهد بود.

| توضیح | |
|-------------------|-----------------------|
| شروع | ۱- شروع |
| ورود داده‌ها | ۲- X را دریافت کن |
| ورود داده‌ها | ۳- Y را دریافت کن |
| محاسبه حاصل تقسیم | ۳- $Z \leftarrow X/Y$ |
| نمایش خروجی | ۴- Z را نمایش بده |
| پایان الگوریتم | ۵- پایان |

در این الگوریتم پس از شروع الگوریتم در مراحل ۲ و ۳ دو عدد از ورودی دریافت شده و در متغیرهای X و Y ذخیره می‌شوند. در مرحله ۴ دو متغیر بر هم تقسیم شده و نتیجه تقسیم در متغیر Z ذخیره می‌شود. در مرحله ۵ حاصل تقسیم روی صفحه نمایش داده می‌شود و در مرحله ۶ الگوریتم پایان می‌یابد.

تمرین: 

الگوریتمی بنویسید تا میانگین سه عدد دلخواه را محاسبه کرده و نمایش دهد. الگوریتمی بنویسید که باقیمانده تقسیم دو عدد دلخواه را بر هم محاسبه کرده و نمایش دهد.

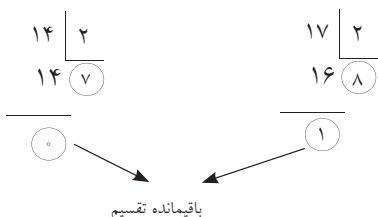
د - دستورالعمل‌های شرطی

گاهی اوقات لازم است با مقایسه مقادیر داده‌های ورودی، اطلاعات خروجی، متغیرها و ... عملیاتی را در الگوریتم هدایت کرده و دستورالعمل‌های خاصی را اجرا کنید یا روند اجرای الگوریتم را با اتخاذ تصمیمات مناسبی کنترل نمایید. در این صورت می‌توانید از دستورات شرطی استفاده کنید.



مثال ۳: الگوریتمی بنویسید که زوج و فرد بودن هر عدد دلخواه را مشخص کند.

در این مسأله داده ورودی شامل یک عدد است و تشخیص زوج یا فرد بودن عدد وارد شده خواسته (مجهول) مسأله است. اگر باقیمانده تقسیم یک عدد بر عدد ۲ صفر شود عدد زوج است و اگر باقیمانده یک باشد عدد موردنظر فرد می‌باشد.



با استفاده از عملگر Mod می‌توان تقسیم‌های فوق را به صورت زیر نوشت:

$$14 \text{ MoD } 2 \rightarrow 0$$

$$17 \text{ MOD } 2 \rightarrow 1$$

در این مرحله با این‌که نکات لازم برای حل مسأله به دست آمده است اما یک مشکل هنوز وجود دارد که یک عدد در هر لحظه نمی‌تواند هم زوج و هم فرد باشد، بنابراین مجبور خواهید بود در زمانی که عدد زوج است یک خروجی و وقتی عدد فرد است خروجی دیگری نمایش داده شود. در چنین حالت‌هایی لازم است از دستورات شرطی استفاده کنید. نحوه استفاده از یک دستور شرطی به صورت زیر است:
 شکل کلی دستورالعمل شرطی به یکی از صورت‌های زیر است.

اگر شرط (ها) آن‌گاه دستور(ات)

اگر شرط (ها) آن‌گاه دستور(ات) در غیر این صورت دستور(ات)

در حالت اول، ابتدا شرط یا شرط‌های آرایه شده بررسی می‌شوند و در صورتی که نتیجه بررسی درست باشد، دستور یا دستورات پس از «آن‌گاه» اجرا می‌شوند، در غیراین صورت (نادرست بودن شرط بررسی شده) دستورالعملی که پس از دستورالعمل شرطی قرار گرفته، اجرا خواهد شد بدون آن‌که دستور یا دستورات پس از «آن‌گاه» اجرا شود. در حالت دوم، شکل کامل تری از دستورالعمل شرطی را ملاحظه می‌کنید که در آن ابتدا شرط یا شرط‌ها مورد بررسی قرار می‌گیرند؛ اگر نتیجه ارزیابی آن‌ها درست باشد، دستورالعمل شرط مانند حالت اول رفتار می‌کند اما اگر نتیجه ارزیابی شرط یا شرط‌ها نادرست باشد بدون آن‌که دستورات بخش «آن‌گاه» اجرا شوند، دستورات موجود در بخش «در غیر این صورت» اجرا خواهند شد سپس دستورالعملی که پس از دستور شرطی قرار دارد، اجرا می‌شود.

جدول ۱ - ۲

| عملگر | عنوان | مثال | نتیجه |
|--------|------------------|---------------|--------|
| = | تساوی | $4=4$ | درست |
| \geq | بزرگ‌تر یا مساوی | $5 \geq 2$ | درست |
| \leq | کوچک‌تر یا مساوی | $100 \leq 15$ | نادرست |
| $>$ | بزرگ‌تر | $29 > 74$ | نادرست |
| $<$ | کوچک‌تر | $7 < 3$ | نادرست |
| $<>$ | نامساوی | $15 <> 8$ | درست |

بنابراین الگوریتم موردنظر بدین صورت خواهد بود:

| توضیح | |
|--|---|
| شروع الگوریتم | ۱- شروع |
| ورود داده | ۲ - X را دریافت کن |
| مقدار باقیمانده تقسیم را در R ذخیره می‌کند. | ۳ - $R \leftarrow X \text{ Mod } 2$ |
| در صورتی که R برابر صفر باشد عبارت «عدد زوج است.» را نمایش می‌دهد. | ۴ - اگر $R = 0$ آن‌گاه «عدد زوج است.» را نمایش بده. |
| در صورتی که R برابر یک باشد عبارت «عدد فرد است.» را نمایش می‌دهد. | ۵ - اگر $R = 1$ آن‌گاه «عدد فرد است.» را نمایش بده. |
| پایان الگوریتم | ۶- پایان |

پس از شروع الگوریتم، در مرحله ۲ یک عدد از ورودی دریافت شده و در متغیر X ذخیره می‌شود سپس در مرحله ۳ باقیمانده تقسیم X بر ۲ محاسبه شده و در متغیر R ذخیره می‌شود. در مرحله ۴ با استفاده از یک دستور شرطی مقدار متغیر R با مقدار صفر مقایسه می‌شود در صورتی که مقدار متغیر نیز صفر باشد نتیجه بررسی شرط درست (True) و در نتیجه عدد زوج خواهد بود و عبارت «عدد زوج است.» نمایش داده می‌شود اما اگر مقدار متغیر R صفر نباشد نتیجه بررسی شرط نادرست (False) بوده و دستور بعد از آن‌گاه اجرا نمی‌شود. در مرحله ۵ مجدداً مقدار R بررسی می‌شود و در صورتی که مقدار آن برابر با یک باشد «عبارت عدد فرد است.» نمایش داده خواهد شد و در صورتی که مقدار آن برابر با یک نباشد عبارت مزبور نمایش داده نمی‌شود.



تمرین:

الگوریتمی بنویسید که بخش‌پذیری یک عدد را بر عدد دیگر تعیین کند

با توجه به مطالب گفته شده و این نکته که یک عدد نمی‌تواند هم زوج و هم فرد باشد الگوریتم مثال قبل را به صورت زیر می‌توان خلاصه‌تر نمود و به جای استفاده از دو دستورالعمل شرطی مسأله را با یک دستورالعمل شرطی حل کرد.

| توضیح | | |
|--|---|---|
| | شروع الگوریتم | ۱- شروع |
| | ورود داده | ۲ - X را دریافت کن |
| | مقدار باقیمانده تقسیم را در R ذخیره می‌کند. | ۳ - $R \leftarrow X \text{ MoD } 2$ |
| در صورتی که مقدار R برابر با صفر باشد عبارت «عدد زوج است.» را نمایش می‌دهد، در غیر این صورت عبارت «عدد فرد است» را نمایش می‌دهد. | | ۴ - اگر $R = 0$ آن‌گاه «عدد زوج است.» را نمایش بده، در غیر این صورت «عدد فرد است.» را نمایش بده |
| | پایان الگوریتم | ۵- پایان |

در این الگوریتم جدید از یک دستورالعمل شرطی استفاده شده است اما در نتیجه اجرای الگوریتم هیچ تغییری به وجود نمی‌آید، در این حالت اگر مقدار باقیمانده یعنی متغیر R برابر با صفر باشد دستوری که در بخش «آن‌گاه» قرار دارد اجرا می‌شود اما اگر مقدار متغیر R برابر با صفر نباشد دستور بخش «در غیر این صورت» اجرا خواهد شد.

تمرین:



الگوریتم بخش پذیری دو عدد بر یکدیگر را با یک دستورالعمل شرطی بنویسید.

مثال ۴: الگوریتمی بنویسید که سه عدد دلخواه را دریافت کرده و بزرگترین مقدار را در بین آنها تعیین کند و نمایش دهد.




در این مسأله سه داده ورودی وجود دارند که آنها را اعداد A، B و C در نظر می‌گیریم، برای پیدا کردن بزرگترین مقدار در بین این سه عدد می‌توانیم از روش مقایسه استفاده کنیم؛ به این منظور ابتدا فرض می‌کنیم مقدار ذخیره شده در متغیر A بزرگترین مقدار است و مقدار آن را در متغیر دیگری به نام MAX ذخیره می‌کنیم، به این دلیل که متغیرهای دیگر یعنی B و C را با مقایسه با این مقدار (متغیر MAX) بررسی کرده و بزرگتر بودن یا نبودن آنها را از مقدار اول مشخص می‌کنیم یعنی متغیر B را با MAX مقایسه کرده و در صورت بزرگتر بودن مقدار B از MAX، متغیر B را در MAX ذخیره می‌کنیم، اگر به همین ترتیب این عملیات را برای متغیر C نیز انجام دهیم، در پایان بزرگترین عدد در متغیر MAX ذخیره خواهد شد.

با توجه به مطالب گفته شده الگوریتم موردنظر به صورت زیر خواهد بود:

| توضیح | |
|--|---|
| شروع الگوریتم | ۱- شروع |
| ورود داده | ۲- اعداد A، B و C را دریافت کن |
| مقدار متغیر A در MAX ذخیره می‌شود. | ۳ - $MAX \leftarrow A$ |
| در صورتی که مقدار B از MAX بزرگتر باشد مقدار B را در MAX ذخیره می‌کند. | ۴- اگر $B > MAX$ آن‌گاه $B \leftarrow MAX$ |
| در صورتی که مقدار C از MAX بزرگتر باشد مقدار C را در MAX ذخیره می‌کند. | ۵ - اگر $C > MAX$ آن‌گاه $C \leftarrow MAX$ |
| نمایش خروجی | ۶ - MAX را نمایش بده |
| پایان الگوریتم | ۷ - پایان |


تمرین:

الگوریتمی بنویسید که چهار عدد دلخواه را دریافت کرده و کوچک‌ترین و بزرگ‌ترین مقدار در بین آن‌ها را به دست آورد، سپس آن را برای اعداد مختلف اجرا و آزمایش کنید.

 **نکته** در صورت نیاز، می‌توان یک دستورالعمل شرطی را در بخش دستورات بعد از «آن‌گاه» یا «در غیر این صورت» دستورالعمل شرطی دیگری قرار داد؛ در این حالت به مجموعه دستورالعمل‌های شرطی، دستورالعمل‌های شرطی تودرتو نیز می‌گویند.

عملگرهای منطقی

گاهی اوقات ممکن است لازم باشد در دستورالعمل‌های شرطی که در حل یک الگوریتم به کار می‌روند بیش از یک شرط مورد بررسی قرار گیرند و با توجه به نتیجه بررسی تمامی شرط‌های ذکر شده سایر دستورالعمل‌ها اجرا شوند. در چنین حالت‌هایی می‌توان از عملگرهای منطقی برای ترکیب دو یا چند شرط استفاده کرد.

 مثال ۵: الگوریتمی بنویسید که مقدار مصرف برق یک مشترک دلخواه را دریافت کرده و براساس جدول ۱-۳، رتبه میزان مصرف وی را مشخص کند.

جدول ۱-۳

| رتبه | میزان مصرف (کیلو وات) | ردیف |
|---------|-----------------------|------|
| کم مصرف | ۰ تا ۵۰ | ۱ |
| عادی | ۵۰ تا ۲۰۰ | ۲ |
| پرمصرف | ۲۰۰ به بالا | ۳ |

برای طراحی الگوریتمی که بتواند رتبه مصرف برق یک مشترک را مشخص کند شرایط مختلفی در نظر گرفته شده است. برای حل این مثال دو روش وجود دارد: روش اول استفاده از دستورالعمل‌های شرطی تودرتو و روش دوم استفاده از عملگرهای منطقی است.

در روش اول لازم است تا برای ردیف‌های ۱ و ۲ در جدول ۳-۱ از یک دستورالعمل شرطی در بخش «آن‌گاه» دستورالعمل شرطی دیگری استفاده شود تا هر دو شرط موردنظر در این ردیف‌ها مورد ارزیابی قرار گیرند. البته برای ردیف ۳ می‌توان از یک دستورالعمل شرطی به تنهایی استفاده کرد. با توجه به این توضیحات می‌توانید الگوریتم زیر را بنویسید:

در این الگوریتم kw به عنوان میزان مصرف برق در نظر گرفته شده است.

۱- شروع

۲- kw را دریافت کن.

۳- اگر $kw \geq 0$ آن‌گاه اگر $kw < 50$ «مشترک کم مصرف است.» را نمایش بده.

۴- اگر $kw \geq 50$ آن‌گاه اگر $kw < 200$ «مشترک عادی است.» را نمایش بده.

۵- اگر $kw \geq 200$ آن‌گاه «مشترک پرمصرف است.» را نمایش بده.

۶- پایان

به طور کلی سه نوع عملگر منطقی که در الگوریتم مورد استفاده قرار می‌گیرند در

جدول ۴-۱ ارائه شده‌اند.

جدول ۴-۱

| عملگر | مفهوم |
|-------|------------|
| AND | «و» منطقی |
| OR | «یا» منطقی |
| NOT | نقیض |

وقتی دو یا چند شرط با عملگر منطقی AND با هم ترکیب می‌شوند، نتیجه مقایسه، زمانی درست (True) خواهد بود که نتیجه تمام شرط‌ها و مقایسه‌ها True باشند؛ به عنوان مثال شرط‌های $X > 0$ و $Y < 5$ را در نظر بگیرید. اگر بخواهیم این دو شرط را با یکدیگر ترکیب کنیم نتیجه ترکیب آن‌ها زمانی درست (True) خواهد بود که نتیجه بررسی هر دو شرط درست (True) باشد. به عنوان نمونه اگر مقدار X برابر ۲ و Y برابر ۱ باشد هر دو شرط درست (True) بوده و نتیجه ترکیب آن‌ها نیز درست (True) خواهد بود.

اما اگر نتیجه بررسی شرط $X > 0$ یا $Y < 5$ یا هر دو شرط نادرست (False) باشد، نتیجه ترکیب آن‌ها نیز نادرست (False) خواهد بود. به عنوان نمونه اگر مقدار X برابر با ۵- و مقدار Y برابر با ۱ باشد شرط $X > 0$ نادرست (False) و شرط $Y < 5$ درست (True) خواهد بود و در نتیجه ترکیب آن‌ها نیز نادرست (False) می‌شود. بنابراین جدول درستی عملگر منطقی AND مطابق جدول ۱-۵ است. در صورتی که دو یا چند شرط را با عملگر منطقی OR با هم ترکیب کنید، نتیجه شرط زمانی نادرست (False) خواهد بود که نتیجه تمام شرط‌ها نادرست (False) باشد و در سایر حالات، نتیجه درست (True) خواهد بود؛ بنابراین جدول درستی عملگر منطقی OR مطابق جدول ۱-۶ است.

جدول ۱-۵

| $X > 0$ | $Y < 5$ | $X > 0$ AND $Y < 5$ |
|---------|---------|---------------------|
| T | F | F |
| F | T | F |
| F | F | F |
| T | T | T |

جدول ۱-۶

| $X > 0$ | $Y < 5$ | $X > 0$ OR $Y < 5$ |
|---------|---------|--------------------|
| T | F | T |
| F | T | T |
| F | F | F |
| T | T | T |

اگر از عملگر NOT روی یک شرط استفاده شود نتیجه درستی آنرا معکوس می‌کند یعنی اگر نتیجه شرط درست (True) باشد آنرا به نادرست (False) و اگر نتیجه شرط نادرست (False) باشد آنرا به درست (True) تبدیل می‌کند. جدول درستی عملگر منطقی NOT مطابق جدول ۱-۷ می‌باشد.

جدول ۱-۷

| $X > 0$ | NOT $X > 0$ |
|---------|-------------|
| T | F |
| F | T |

نکته در جدول‌های ۱-۵، ۱-۶ و ۱-۷، T بیانگر درست (True) و F بیانگر نادرست (False) می‌باشد.



تمرین:

الگوریتمی بنویسید تا معدل یک دانش آموز را دریافت کرده و رتبه وی را براساس شرایطی که در ادامه می آید، تعیین کند:

الف- در صورتی که میانگین نمرات بیشتر از ۱۸ باشد، رتبه ممتاز برای وی اعلام شود.

ب- در صورتی که میانگین نمرات بین ۱۶ تا ۱۸ باشد، رتبه خوب برای وی اعلام شود.

ج- در صورتی که میانگین نمرات بین ۱۲ تا ۱۶ باشد، رتبه متوسط برای وی اعلام شود.

د- در صورتی که میانگین نمرات کمتر از ۱۲ باشد، رتبه ضعیف برای وی اعلام شود.

ه- دستورالعملهای تکرار (حلقه‌ها)

استفاده از دستورالعمل‌هایی که تاکنون فرا گرفته‌اید در حل بعضی از مسایل کافی نیست و لازم است تا برخی از دستورالعمل‌ها را به دفعات تکرار نمایید؛ در این موارد از دستورالعمل تکرار یا همان حلقه‌ها استفاده کنید.

مثال ۶: الگوریتمی بنویسید که مجموع اعداد ۱ تا ۱۰ را محاسبه نموده و نمایش دهد.



اگر برای حل این مسأله بخواهید از دستورالعمل‌هایی که تاکنون ارائه شده‌اند استفاده کنید در این حالت الگوریتمی به صورت زیر به دست می آید.

۱- شروع

۲- $SUM \leftarrow 0$

۳- $SUM = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 + 10$

۴- SUM را نمایش بده

۵- پایان

با این حال مشاهده می کنید که الگوریتم شکل نامناسبی دارد و اگر داده‌ها زیادتر شوند استفاده از این روش عملاً غیرممکن است. برای حل این مشکل می توان دستورالعمل تکرار را به کار برد.

دستورالعمل حلقه از اجزای مختلفی تشکیل می‌شود که عبارتند از:

شمارنده حلقه: یک متغیر عددی است که تعداد دفعات تکرار دستورالعمل‌ها را در حلقه کنترل می‌کند. مقدار شمارنده در هر بار اجرای حلقه افزایش یا کاهش می‌یابد.

مقدار اولیه: مقدار اولیه حلقه قبل از شروع حلقه تعیین می‌شود و به وسیله آن می‌توان مقدار اولیه را برای شمارنده حلقه تعیین کرد.

شرط حلقه: برای کنترل تعداد دفعات تکرار حلقه، باید از یک شرط استفاده کرد. شرط موجود در حلقه، نقطه پایان تکرار دستورالعمل‌ها را در حلقه مشخص می‌کند و باید به گونه‌ای تنظیم شود تا از ایجاد حلقه نامحدود جلوگیری کند. برای ایجاد شرط در یک حلقه می‌توان از دستورالعمل‌های شرطی استفاده کرد.

دستورات حلقه: بخش دیگر در حلقه، دستورالعمل‌هایی هستند که داخل حلقه تکرار می‌شوند. این دستورالعمل‌ها با توجه به نیاز مسأله انتخاب می‌شوند.

بنابراین می‌توان الگوریتم مثال ۸ را به صورت زیر را ارائه نمود:

| توضیح | |
|---|--|
| | ۱- شروع |
| | ۲- $N \leftarrow 1$ |
| | ۳- $SUM \leftarrow 0$ |
| | ۴- $SUM \leftarrow SUM + N$ |
| | ۵- $N \leftarrow N + 1$ |
| | ۶- اگر $10 \leftarrow N$ آن‌گاه به مرحله ۴ برو |
| | ۷- $SUM - N$ را نمایش بده |
| | ۸- پایان |
| تنظیم مقدار اولیه برای شمارنده حلقه | |
| محاسبه مجموع اعداد افزایش شمارنده حلقه شرط حلقه | |

در این الگوریتم پس از شروع، ابتدا عدد ۱ در متغیر N ذخیره می‌شود که به عنوان شمارنده حلقه از آن استفاده می‌شود. در مرحله بعد مقدار صفر در متغیر SUM ذخیره می‌شود که برای محاسبه مجموع اعداد از آن استفاده می‌شود. در مرحله ۴ مجموع اعداد محاسبه می‌شود.

در مرحله بعد مقدار شمارنده حلقه N یک واحد افزایش می‌یابد، در مرحله ۶ یک

جدول ۸-۱

| N | SUM | خروجی |
|---------------|---------------|-------|
| ۱ | ۱ | ۵۵ |
| ۲ | ۳ | |
| ۳ | ۶ | |
| ۴ | ۱۰ | |
| ۵ | ۱۵ | |
| ۶ | ۲۱ | |
| ۷ | ۲۸ | |
| ۸ | ۳۶ | |
| ۹ | ۴۵ | |
| ۱۰ | ۵۵ | |
| ۱۱ | | |

دستورالعمل شرطی مقدار شمارنده را بررسی می کند تا تعداد دفعات تکرار دستورالعمل ها از مقدار ۱۰ بیشتر نشود. در این حالت اگر مقدار شمارنده کوچک تر یا مساوی ۱۰ باشد اجرای الگوریتم به مرحله ۴ منتقل می شود. دستورات مراحل ۴، ۵ و ۶ آن قدر تکرار می شوند تا مقدار شمارنده از مقدار ۱۰ بیشتر شود و در نتیجه اجرای حلقه ایجاد شده خاتمه می یابد و اجرای الگوریتم با اجرای مرحله ۷ ادامه یافته و با نمایش میانگین اعداد در مرحله ۸ خاتمه می یابد. اگر الگوریتم را اجرا و آزمایش کنید جدول بررسی عملکرد آن به صورت جدول ۸-۱ خواهد بود.

مثال ۷: الگوریتمی بنویسید که اعداد طبیعی کوچک تر از ۵۰ را نمایش دهد.



| توضیح | |
|-------------------------------------|---------------------------------------|
| | ۱- شروع |
| | ۲- $N \leftarrow 1$ |
| تنظیم مقدار اولیه برای شمارنده حلقه | ۳- N را نمایش بده |
| نمایش خروجی | ۴- $K \leftarrow N + 1$ |
| افزایش شمارنده حلقه | ۵- اگر $N < 50$ آن گاه به مرحله ۳ برو |
| دستورات حلقه | ۶- پایان |
| شرط حلقه | |

مثال ۸: الگوریتمی بنویسید که اعداد زوج کوچک تر یا مساوی عدد طبیعی و

دلخواه N را نمایش دهد. این الگوریتم مشابه الگوریتم قبلی است با این تفاوت که مقدار شمارنده از ۲ شروع شده و هر بار به میزان ۲ واحد افزایش می یابد.



| توضیح | |
|--|---|
| خاتمه الگوریتم در صورتی که داده ورودی عدد یک باشد. | ۱- شروع |
| | ۲- N را دریافت کن |
| | ۳- اگر $N=1$ آن‌گاه پایان |
| | ۴- $K \leftarrow 2$ |
| | ۵- K را نمایش بده |
| | ۶- $K \leftarrow K + 2$ |
| | ۷- اگر $K \leq N$ آن‌گاه برو به مرحله ۵ |
| | ۸- پایان |

تمرین:



الگوریتمی بنویسید که مضارب کوچک‌تر از 100 عدد 3 را نمایش دهد.



مثال ۹: الگوریتمی بنویسید که مجموع و تعداد اعداد طبیعی موجود بین اعداد

طبیعی دلخواه M و N را نمایش دهد (با فرض این که M کوچک‌تر از N باشد).

- ۱- شروع
- ۲- M و N را دریافت کن
- ۳- $SUM \leftarrow 0$ و $NO \leftarrow 0$
- ۴- $K \leftarrow M+1$
- ۵- $SUM \leftarrow SUM+K$
- ۶- $NO \leftarrow NO+1$
- ۷- $K \leftarrow K+1$
- ۸- اگر $K < N$ آن‌گاه برو به مرحله ۵
- ۹- SUM و NO را نمایش بده
- ۱۰- پایان

نکته‌ای که باید در این الگوریتم بدان اشاره کنیم نحوه محاسبه مجموع و تعداد اعداد

طبیعی است. برای این کار از متغیر NO برای شمارش تعداد اعداد بین M و N استفاده می‌شود. متغیر K برای ایجاد اعداد طبیعی بین M و N به کار می‌رود. در صورت اجرای این الگوریتم به ازای $M=3$ و $N=9$ ، جدول بررسی عملکرد به صورت زیر خواهد بود:

جدول ۱-۹

| M | N | K | SUM | NO | خروجی | |
|---|--------------|--------------|---------------|--------------|-------|---|
| ۳ | ۹ | ۴ | ۶ | ۶ | | |
| | ۴ | ۵ | ۴ | ۴ | | |
| | ۳ | ۶ | ۹ | ۴ | | |
| | ۲ | ۷ | ۱۵ | ۳ | | |
| | ۱ | ۸ | ۲۲ | ۴ | | |
| | ۳ | ۹ | ۳۰ | ۵ | ۳۰ | ۵ |

تمرین:



الگوریتم قبل را به گونه‌ای تنظیم کنید که بدون در نظر گرفتن فرض $M < N$ ، توانایی ارائه پاسخ صحیح را داشته باشد، به عبارت دیگر در صورتیکه کاربر عدد بزرگ‌تر را برای M و عدد کوچک‌تر را برای N وارد کند، پاسخ الگوریتم درست باشد.



مثال ۱۰: الگوریتمی بنویسید که مقسوم علیه‌های عدد طبیعی و دلخواه M را محاسبه کند. به این منظور باید از یک حلقه استفاده کنید و متغیر M را هر بار بر یک متغیر K که به عنوان مقسوم علیه در نظر گرفته می‌شود و مقدار اولیه آن ۱ است، تقسیم نمایید سپس باقیمانده این تقسیم را بررسی کنید. در صورتی که مقدار باقیمانده مساوی صفر باشد به این معنی است که متغیر K میتواند مقسوم علیه متغیر M باشد، بنابراین متغیر K نمایش داده می‌شود. برای محاسبه باقیمانده تقسیم صحیح نیز می‌توانید از عملگر Mod استفاده کنید.

اکنون الگوریتم موردنظر به صورت زیر خواهد بود:

۱ - شروع

۲ - M را دریافت کن

۳ - $K \leftarrow 1$

۴ - اگر $M \text{ Mod } K = 0$ آن‌گاه K را نمایش بده

۵ - $K \leftarrow K + 1$

۶ - اگر $K \leq M$ آن‌گاه برو به مرحله ۴

۷ - پایان



مثال ۱۱: الگوریتمی بنویسید که مجموع ارقام هر عدد طبیعی و دلخواه را محاسبه کرده و نمایش دهد.

برای طراحی الگوریتم قبل و محاسبه مجموع ارقام یک عدد، ابتدا باید ارقام عدد را یکی یکی از آن جدا کنید؛ به این منظور می‌توانید از عملگر Mod و محاسبه باقیمانده تقسیم عدد مربوطه بر عدد ۱۰ استفاده نمایید، سپس خارج قسمت عدد مربوطه را که بر ۱۰ تقسیم می‌شود، محاسبه کنید. از این عدد می‌توانید برای به دست آوردن رقم بعدی در عدد اصلی استفاده نمایید. این عملیات می‌تواند در یک حلقه تا رسیدن به خارج قسمت صفر ادامه یابد. برای محاسبه خارج قسمت تقسیم صحیح نیز می‌توانید از عملگر ۱ استفاده کنید؛ به عنوان نمونه به این مثال توجه کنید:

$$\begin{array}{r}
 17 \overline{) 5} \\
 \underline{15} \\
 2
 \end{array}$$

خارج قسمت تقسیم صحیح ← ۳

باقیمانده تقسیم صحیح ← ۲

$$17 \overline{) 5} = 3$$

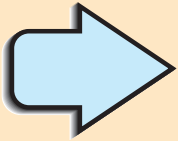
با توجه به مطالب ارایه شده، اگر بخواهیم ارقام یک عدد را از آن جدا کنیم روش ریاضی این عملیات به صورت بعد خواهد بود:

به عنوان مثال اگر عدد مورد نظر ۴۲۵۷ باشد:

$$\begin{array}{r}
 4257 \overline{) 10} \\
 \underline{4250} \\
 (7) \\
 \overline{) 4250} \\
 \underline{ 420} \\
 (5) \\
 \overline{) 420} \\
 \underline{ 40} \\
 (2) \\
 \overline{) 40} \\
 \underline{ 0} \\
 (2)
 \end{array}$$

بنابراین می توان الگوریتم زیر را برای حل این مثال ارایه کرد:

- ۱ - شروع
- ۲ - M را دریافت کن
- ۳ - $SUM \leftarrow 0$
- ۴ - $DIGIT \leftarrow M \text{ Mod } 10$
- ۵ - $SUM \leftarrow SUM + DIGIT$
- ۶ - $M \leftarrow M \setminus 10$
- ۷ - اگر $M <> 0$ آن گاه برو به مرحله ۴
- ۸ - SUM را نمایش بده
- ۹ - پایان



Learn in English

Algorithm

To make a computer do anything, you have to write a computer program. To write a computer program, you have to tell the computer, step by step, exactly what you want it to do. The computer then «executes» the program, following each step mechanically, to accomplish the end goal.

When you are telling the computer *what* to do, you also get to choose how it's going to do it. That's where computer algorithms come in. The algorithm is the basic technique used to get the job done. Let's follow an example to help get an understanding of the algorithm concept.

Let's say that you have a friend arriving at the airport, and your friend needs to get from the airport to your house. Here are four different algorithms that you might give your friend for getting to your home:

* **The taxi algorithm**

1 - Go to the taxi stand

Get in a taxi

2 - Give the driver my address

* **The call-me algorithm:**

1 - When your plane arrives, call my cell phone.

2 - Meet me outside baggage claim.

* **The rent-a-car algorithm:**

1 - Take the shuttle to the rental car place.

2 - Rent a car.

3 - Follow the directions to get to my house.

* **The bus algorithm:**

1 - Outside baggage claim.

2 - Transfer to bus on Main Street.

3 - Get off on 20 street.

4 - Walk two blocks north to my house.

All four of these algorithms accomplish exactly the same goal, but each algorithm does it in completely different way. Each algorithm also has a different cost and a different travel time. Taking a taxi, for example, is probably the fastest way, but also the most expensive. Taking the bus is definitely less expensive, but a whole lot slower. You choose the algorithm based on the circumstances.

واژه‌نامه

| | |
|--------------|-----------|
| Circumstance | وضعیت |
| Driver | راننده |
| Execute | اجرا کردن |
| Expensive | گران |
| Fast | سریع |
| Program | برنامه |
| Rent | اجاره |

خلاصه مطالب

- برای دستیابی به روش حل مسایل، شناسایی سه پارامتر مقادیر معلوم، محاسبات و خواسته‌های مسأله (مجهولات) کار را آسانتر می‌کند.
- الگوریتم مجموعه‌ای از دستورالعمل‌هاست که به صورت مرحله به مرحله اجرا می‌شوند و هدف مشخصی را دنبال می‌کنند و دارای شروع و خاتمه می‌باشند.
- انواع دستورالعمل‌ها در الگوریتم عبارتند از: ورودی، محاسباتی، خروجی، شرطی و حلقه‌ها.
- دستورالعمل‌های ورودی برای دریافت داده‌ها و دستورالعمل‌های خروجی برای نمایش داده و اطلاعات خروجی یا پیام‌های موردنیاز استفاده می‌شوند.
- دستورالعمل‌های محاسباتی برای انجام عملیات ریاضی و محاسباتی استفاده می‌شوند.
- دستورالعمل‌های شرطی برای انجام مقایسه‌ها و کنترل روند اجرای الگوریتم استفاده می‌شوند.
- دستورالعمل‌های تکرار (حلقه‌ها) برای ایجاد تکرار اجرای دستورالعمل‌ها استفاده می‌شوند.
- عملگر، یک یا مجموعه عملیاتی است که در ماشین الگوریتم به صورت از پیش آماده تعریف شده است.

آزمون نظری

۱ - نتیجه اجرای الگوریتم زیر چیست؟

۱ - شروع

۲ - $SUM \leftarrow 0$

۳ - $I \leftarrow 1$

۴ - $SUM \leftarrow SUM + I$

۵ - $I \leftarrow I + 1$

۶ - اگر $I \leq 100$ آن گاه برو به مرحله ۴

۷ - SUM را نمایش بده

۸ - پایان

الف - اعداد ۱ تا ۹۹ را نمایش می دهد

ب - مجموع اعداد ۰ تا ۹۹ را نمایش می دهد.

ج - مجموع اعداد ۱ تا ۱۰۰ را نمایش می دهد.

د - مجموع اعداد ۱ تا ۹۹ را نمایش می دهد.

۲ - کدام گزینه در رابطه با حاصل الگوریتم زیر درست است؟

۱ - شروع کن

۲ - $I \leftarrow 2$

۳ - I را نمایش بده

۴ - $K \leftarrow I \times 2$

۵ - $I \leftarrow I + 2$

۶ - K را نمایش بده

۷ - اگر $I < 10$ آن گاه برو به مرحله ۵ در غیر این صورت برو به مرحله ۸

۸ - پایان

ب - ۲ و ۴ و ۶ و ۸

الف - ۲ و ۴ و ۴ و ۴ و ۴

د - ۲ و ۴ و ۴ و ۴

ج - ۲ و ۴ و ۶ و ۸ و ۱۰

۳ - نتیجه اجرای الگوریتم زیر چیست؟

۱ - شروع

۲ - $I \leftarrow 1$

۳ - اگر $I \bmod 2 = 0$ آن گاه I را نمایش بده.

۴ - $I \leftarrow I + 1$

۵ - اگر $I < 100$ آن‌گاه برو به مرحله ۳

۶ - پایان

الف- اعداد بین صفر و ۱۰۰

ب- اعداد طبیعی فرد کوچک‌تر از ۱۰۰

ج- اعداد طبیعی زوج کوچک‌تر از ۱۰۰

د- اعداد بین ۱ و ۱۰۰

۴ - کدام عملگر از اولویت بالاتری برخوردار است؟

الف- + ب- / ج- Mod د- -

۵ - در صورتیکه $X = 5$ ، $Y = -10$ و $Z = 27$ باشد، نتیجه عبارت زیر چیست؟

($X > 0$) یا ($Y < 20$) و ($Z >= X$)

الف- درست ب- نادرست ج- ۱- د- ۰

۶ - حاصل عبارت زیر کدام است؟

$5 + 10 - 20 / 4 \times 3$

الف- ۰ ب- $3/75$ ج- ۳ د- ۱۳

۷ - حاصل عبارت زیر چیست؟

$17 \text{ Mod } 3$

الف- ۲ ب- ۳ ج- ۴ د- ۵

۸ - حاصل عبارت زیر چیست؟

2514

الف- ۱ ب- ۲ ج- ۵ د- ۶

۹ - در کدام مرحله از مراحل حل یک مسأله ارتباط بین داده‌ها و مجهول‌ها مشخص

می‌شود؟

الف- شناخت مسأله ب- تجزیه مسأله

ج- طراحی راه‌حل د- تحلیل مسأله

۱۰ - نتیجه اجرای الگوریتم زیر چیست؟

۱ - شروع

۲ - $K \leftarrow 0$ و $SUM \leftarrow 0$

۳ - N را دریافت کن

۴ - اگر $N \text{ Mod } K = 0$ آن گاه $SUM \leftarrow SUM + K$

۵ - $K \leftarrow K + 1$

۶ - اگر $K \leq N$ آن گاه برو به مرحله ۴

۷ - SUM را نمایش بده

۸ - پایان

الف- مجموع باقیمانده تقسیم N بر K ب- مجموع مقسوم علیه های عدد N

ج- مجموع مقسوم علیه های عدد K د- مجموع مقسوم علیه های عدد K بر N

۱۱ - The algorithm is the basic technique used to write a for running in the computer.

a- commands

b- statments

c- program

d- step by step

۱۲- الگوریتم را تعریف کنید و برای آن یک مثال بنویسید.

۱۳- مراحل حل مسأله را نام ببرید.

۱۴- انواع دستورالعمل ها را در الگوریتم نام ببرید.

۱۵- شرایط اصلی یک الگوریتم خوب را توضیح دهید.

۱۶- کاربرد عملگرهای منطقی و انواع آنها را توضیح دهید.

۱۷- دستورالعمل های شرطی و کاربرد آنها را توضیح دهید.

۱۸- دستورالعمل های تکرار و کاربرد آنها را توضیح دهید.

۱۹- عوامل مؤثر در شناخت و حل مسائل را توضیح دهید.

۲۰- منظور از جامع بودن الگوریتم چیست؟

۲۱- عوامل اصلی در شناخت دقیق یک مسأله را توضیح دهید.

آزمون عملی

- ۱ - الگوریتمی بنویسید که مضارب کوچک‌تر از ۱۰۰۰ عدد ۵ را محاسبه کرده و نمایش دهد.
- ۲ - الگوریتمی بنویسید که دما را برحسب فارنهایت دریافت کرده و معادل آن را به سانتی‌گراد نمایش دهد (با توجه به فرمول $C = \frac{5}{9}(F - 32)$).
- ۳ - عدد طبیعی و دلخواه N را دریافت کرده و $N!$ را محاسبه کرده و نمایش دهد.
- ۴ - الگوریتمی بنویسید که حقوق یک کارمند را دریافت کرده و میزان مالیات حقوق وی را مطابق جدول زیر پس از محاسبه نمایش دهد.

| حقوق | مالیات |
|-------------------------------------|-------------|
| کمتر از ۴/۰۰۰/۰۰۰ ریال | صفر |
| از ۴/۰۰۰/۰۰۰ ریال تا ۵/۰۰۰/۰۰۰ ریال | ۲ درصد حقوق |
| از ۵/۰۰۰/۰۰۰ ریال تا ۷/۰۰۰/۰۰۰ ریال | ۳ درصد حقوق |
| از ۷/۰۰۰/۰۰۰ ریال به بالا | ۵ درصد حقوق |

- ۵ - الگوریتمی بنویسید که نمرات ۱۰ درس یک دانش‌آموز را دریافت کرده و معدل و میانگین نمرات وی را نمایش دهد.
- ۶ - الگوریتمی بنویسید که عدد طبیعی و دلخواه N را دریافت کرده و ارقام آن را معکوس کند (مثلاً عدد ۲۴۸۵ به ۵۸۴۲ تبدیل شود).
- ۷ - الگوریتمی بنویسید که مقدار زمان را براساس ثانیه دریافت کرده و مقدار ساعت‌ها، دقیقه‌ها و ثانیه‌های آن را محاسبه کرده و نمایش دهد.
- ۸ - الگوریتمی بنویسید که عدد طبیعی و دلخواه M را دریافت کرده و اعداد زوج کوچک‌تر از آن را نمایش دهد.
- ۹ - الگوریتمی بنویسید که دو عدد طبیعی دلخواه را دریافت کرده و بزرگ‌ترین مقسوم‌علیه مشترک آن‌ها را محاسبه نموده و نمایش دهد.

توانایی ترسیم فلوجارت

هدف‌های رفتاری

- پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:
- ۱ - مفهوم فلوجارت را توضیح دهد.
 - ۲ - علایم ترسیم فلوجارت و نحوه استفاده از آن‌ها را توضیح دهد.
 - ۳ - بتواند انواع فلوجارت را برای مسایل مختلف طراحی کند.

کلیات

در ارایه الگوریتم‌ها به منظور حل مسایل مختلف از جملات فارسی یا سایر زبان‌ها به همراه تعدادی از علائم قراردادی استفاده می‌شود، البته در الگوریتم‌های ساده و کوچک، این روش به خوبی شما را برای تبدیل الگوریتم به برنامه موردنظر کمک می‌کند اما با پیچیده شدن الگوریتم و افزایش دستورالعمل‌ها این کار دشوار خواهد شد، بنابراین لازم است از روشهای مفیدتری استفاده کنید؛ یکی از این روش‌ها، ترسیم و طراحی فلوچارت است. فلوچارت (نمودار گردش) در واقع مجموعه‌ای از اشکال و ترسیمات قراردادی است که دستورالعمل‌ها و ترتیب اجرای آنها را مطابق با الگوریتم موردنظر نمایش می‌دهد. فلوچارت تمام ویژگیهای الگوریتم را داشته، علاوه بر این امکان درک بهتر از نحوه اجرای دستورالعمل‌ها را نیز به وجود می‌آورد.

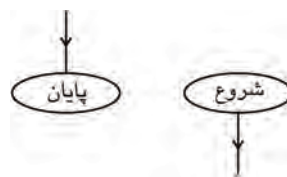
فلوچارت‌ها را می‌توانید پس از طراحی الگوریتم یا به طور مستقل پس از بررسی مسأله و انتخاب روش حل آن ترسیم کنید.

۱-۲-۱ علائم و اشکال در فلوچارت

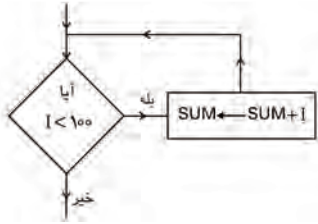
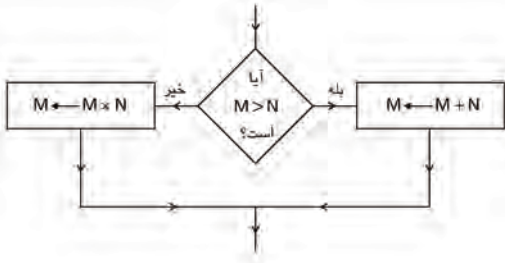
برای تبدیل یک الگوریتم به فلوچارت از علائم قراردادی به جای استفاده از جملات استفاده می‌شود و به جای هر یک از دستورالعمل‌ها، می‌توانید شکل و علامت معادل آن را به کار بگیرید و براساس ترتیب اجرای دستورالعمل‌ها این اشکال را با استفاده از خطوط فلش دار به یکدیگر متصل نمایید. لازم به ذکر است که تعداد فلش‌هایی که می‌تواند به هر علامت وارد شود، نامحدود است اما در تمام علائم بجز علامت شرط فقط یک فلش می‌تواند خارج شود.

۱-۲-۱-۱ علائم شروع و پایان

برای دستورالعمل‌های شروع کن و پایان از علامت بیضی به صورت زیر استفاده کنید:

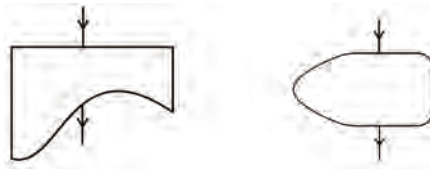


مانند:

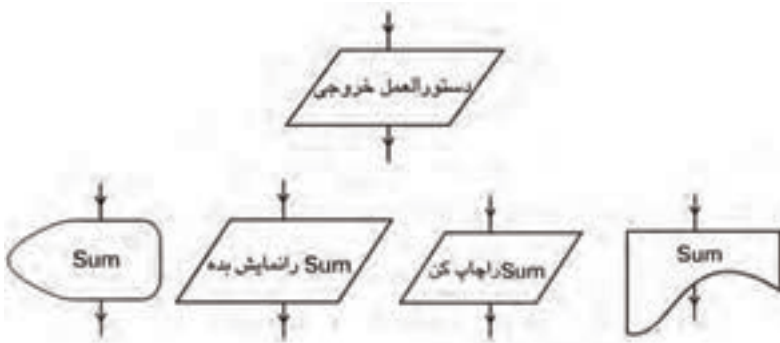


۵-۱-۲ علایم خروجی

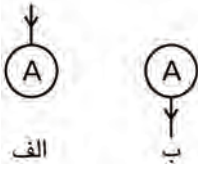
با توجه به اینکه اطلاعات خروجی می‌توانند به چاپگر یا صفحه نمایش ارسال شوند از این علایم استفاده می‌شود:



به علاوه می‌توانید از علامت متوازی الاضلاع به جای دستورالعمل خروجی نیز استفاده کنید.



۶-۱-۲ علامت اتصال



گاهی اوقات ممکن است فلوجارت به اندازه‌ای بزرگ باشد که در یک صفحه کاغذ قرار نگیرد؛ در چنین شرایطی می‌توانید از علامت اتصال استفاده کنید (شکل الف) و ادامه فلوجارت را در صفحه دیگری که آن هم با یک علامت اتصال دیگر شروع می‌شود (شکل ب)، ادامه دهید و در داخل هر دو علامت یک حرف از حروف الفبا یا یک عدد مثبت قرار دهید.



مثال ۱: الگوریتم و فلوجارتی بنویسید که وزن و بهای واحد یک کالا را دریافت کرده و بهای کل آن را محاسبه کند.



۱ - شروع

۲ - W را دریافت کن

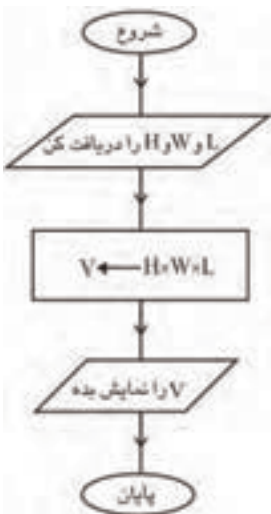
۳ - P را دریافت کن

۴ - $P \times W$ را نمایش بده

۵ - پایان



مثال ۲: الگوریتم و فلوجارتی بنویسید که حجم یک مکعب را محاسبه کند.



۱ - شروع

۲ - L و W و H را دریافت کن

۳ - $V \leftarrow H \times W \times L$

۴ - V را نمایش بده

۵ - پایان

فلوجارت این الگوریتم به این صورت خواهد بود:

تمرین: 

فلوچارتی رسم کنید که محیط یک مربع دلخواه را محاسبه کند.



مثال ۳: الگوریتم و فلوچارتی بنویسید که دو عدد را دریافت کرده و حاصل تفریق، جمع، تقسیم و ضرب آنها را نمایش دهد.

- ۱ - شروع
- ۲ - M و N را دریافت کن
- ۳ - $M - N$ را نمایش بده
- ۴ - $M + N$ را نمایش بده
- ۵ - $M \times N$ را نمایش بده
- ۶ - M / N را نمایش بده
- ۷ - پایان

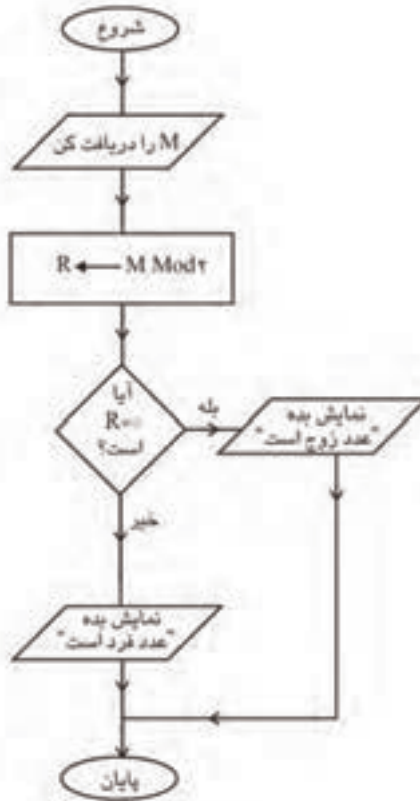
تمرین: 

فلوچارتی رسم کنید که میانگین سه عدد دلخواه را محاسبه کند.

مثال ۴: الگوریتم و فلوچارتی بنویسید که عدد طبیعی و دلخواه M را دریافت کرده و زوج یا فرد بودن آن را معین کند.

- ۱ - شروع
- ۲ - M را دریافت کن
- ۳ - اگر $M \text{ MOD } 2 = 0$ آن گاه «عدد زوج است» را نمایش بده در غیراینصورت «عدد فرد است» را نمایش بده
- ۴ - پایان

فلوچارت الگوریتم قبل به این صورت خواهد بود:



تمرین:



فلوچارتی رسم کنید که تعدادی عدد دلخواه را دریافت کرده و بزرگ‌ترین مقدار را محاسبه کند و نمایش دهد.

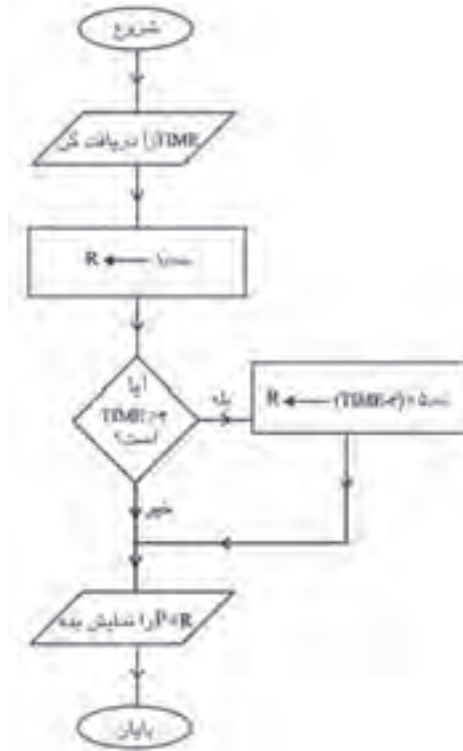


مثال ۵: فلوچارتی بنویسید که با توجه به شرایط زیر هزینه پارکینگ را برای یک

اتومبیل دلخواه محاسبه نموده و نمایش دهد.

الف- هزینه پارکینگ تا سه ساعت از زمان ورود ۰/۰۰۰/۰۰۰ اریال

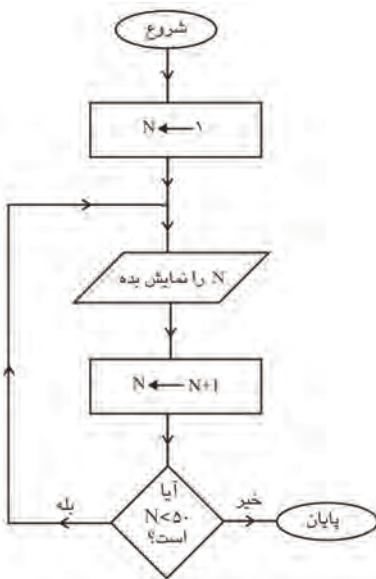
ب- هزینه پارکینگ بعد از سه ساعت هر ساعت ۵/۰۰۰ ریال



تمرین:



فلوچارتی رسم کنید که میانگین دمای یک شهر را در طول یک هفته محاسبه کرده و نمایش دهد.



مثال ۶: فلوچارتی رسم کنید که اعداد طبیعی



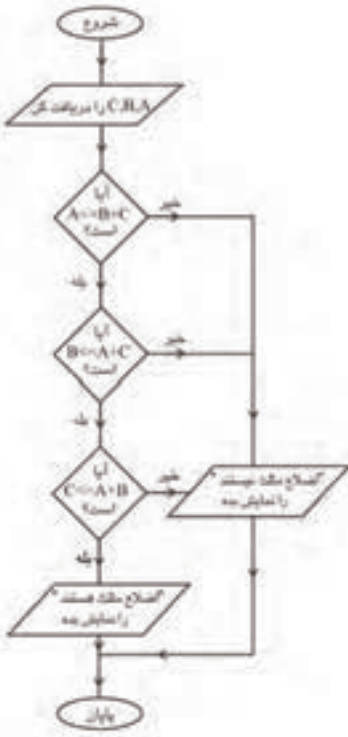
کوچکتر از ۵۰ را نمایش دهد.

تمرین: 

فلوچارتی رسم کنید که اعداد زوج کوچک‌تر یا مساوی عدد طبیعی و دلخواه N را نمایش دهد.



مثال ۷: فلوچارتی رسم کنید که سه عدد دلخواه را دریافت کرده و معین کند سه عدد می‌توانند اضلاع یک مثلث باشد (شرط مثلث بودن سه عدد این است که: $A \leq B + C$ یا $B \leq A + C$ یا $C \leq A + B$).



تمرین: 

فلوچارتی رسم کنید که دو عدد را دریافت کرده و بخش پذیری عدد اول را بر عدد دوم بررسی کند.



مثال ۸: الگوریتم و فلوچارتی بنویسید که عدد طبیعی و دلخواه m را دریافت کرده و مقسوم علیه‌های آن را نمایش دهد.

۱ - شروع

۲ - M را دریافت کن

۳ - $K \leftarrow 1$

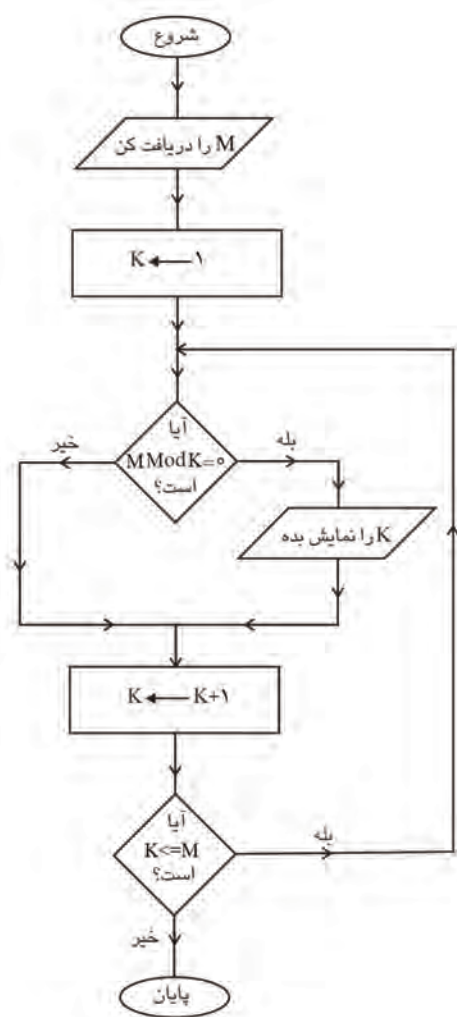
۴ - اگر $M \text{ Mod } K = 0$ آن‌گاه K را نمایش

بده

۵ - $k \leftarrow K + 1$

۶ - اگر $K \leq M$ آن‌گاه برو به مرحله ۴

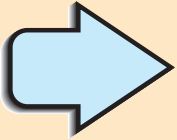
۷ - پایان



تمرین:



فلوچارتی رسم کنید که دو آرایه یک بعدی را دریافت کرده و حاصل جمع آن‌ها را در آرایه یک بعدی جداگانه‌ای ذخیره نموده و نمایش دهد.



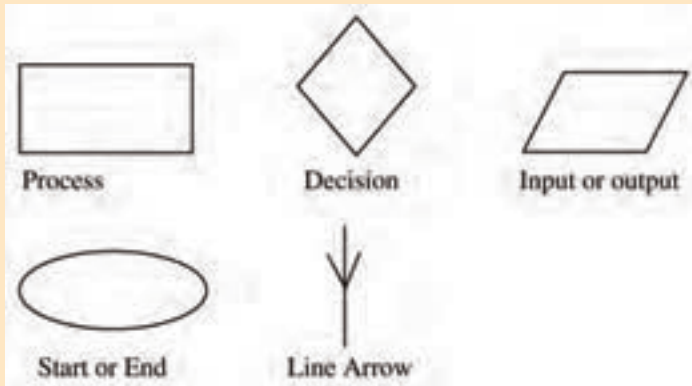
Learn in English

Flowchart Symbols

Flowcharts use special shapes to represent different types of actions or steps in a process.

Lines and arrows show the sequence of the steps and their relationships among them.

Flowcharts Shapes



واژه‌نامه

Decision
Flowchart

شرطی
مجموعه‌ای از اشکال هندسی که نوع دستورالعمل‌ها و
ترتیب اجرای آن‌ها را نمایش می‌دهد.

Lines and arrows

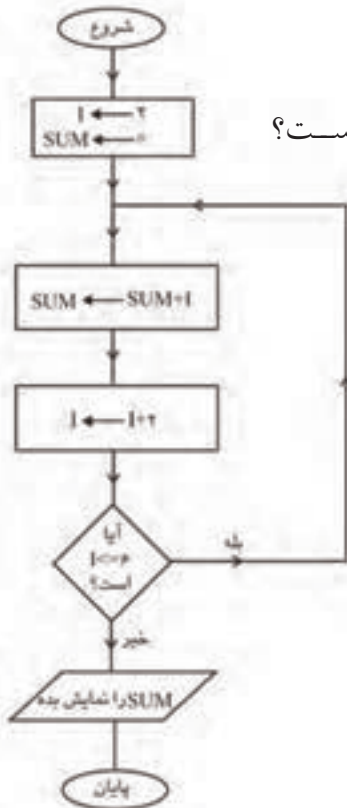
خطوط فلش‌دار

Process
Relationship
Sequence
Step

پردازش
رابطه
ترتیب
مرحله

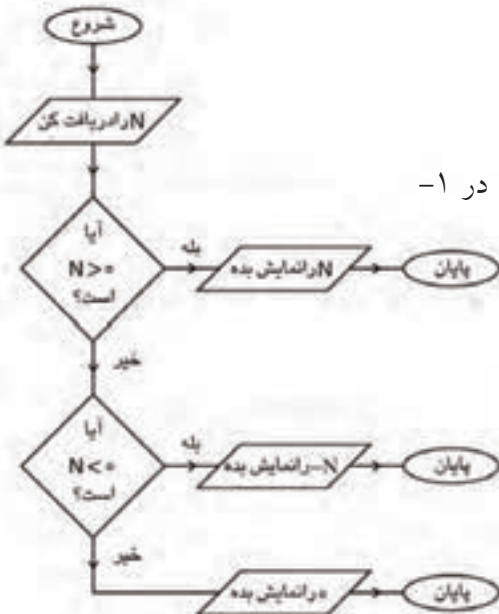
آزمون نظری

- ۱ - کدام گزینه در رابطه با اجرای فلوجارت زیر درست است؟
- الف- نمایش اعداد زوج کوچکتر از ۶
 - ب- نمایش اعداد زوج مساوی ۶
 - ج- نمایش مجموع اعداد زوج کوچکتر یا مساوی ۶
 - د- مجموع اعداد زوج کوچکتر از ۶



- ۲ - عملکرد فلوجارت زیر چیست؟

- الف- نمایش قرینه یک عدد
- ب- نمایش قدرمطلق یک عدد
- ج- نمایش عدد به صورت گرد شده
- د- نمایش حاصل ضرب هر عدد دلخواه در ۱-



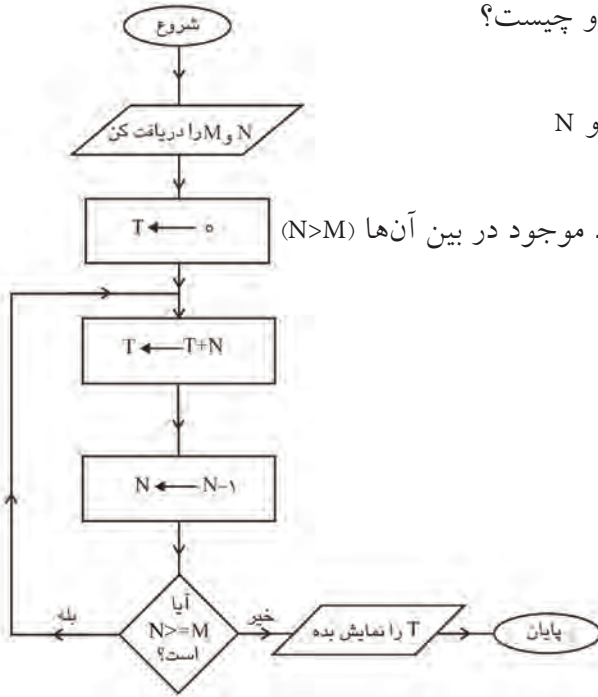
۳ - خروجی فلوجارت روبه‌رو چیست؟

الف- اعداد بین M و N

ب- مجموع اعداد بین M و N

ج- حاصل ضرب M در N

د- مجموع M و N و اعداد موجود در بین آنها ($N > M$)



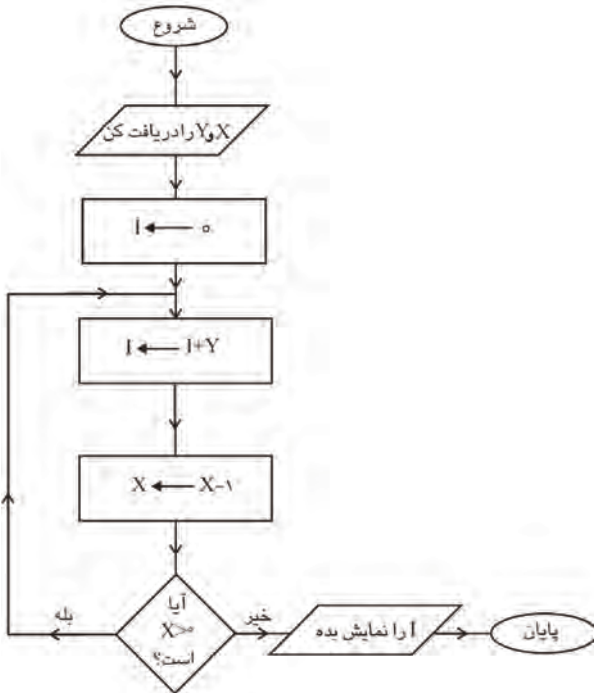
۴ - خروجی این فلوجارت چیست؟

الف- حاصل جمع X و Y

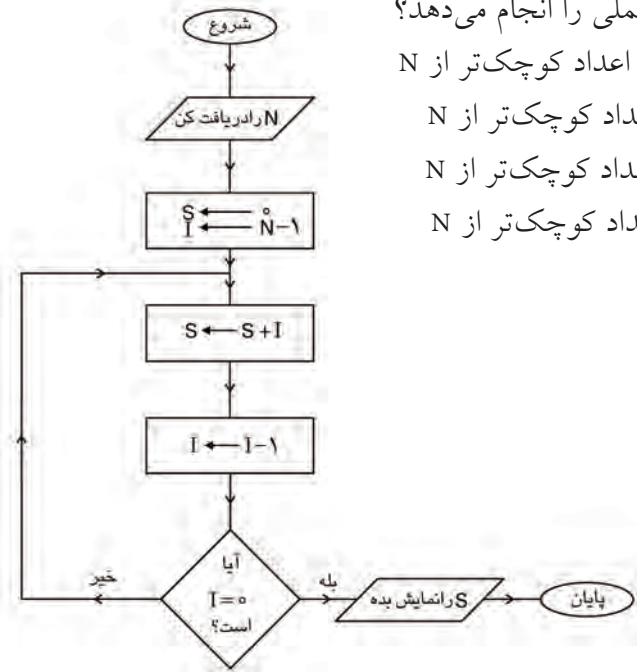
ب- حاصل ضرب X در Y

ج- مجموع اعداد بین X و Y

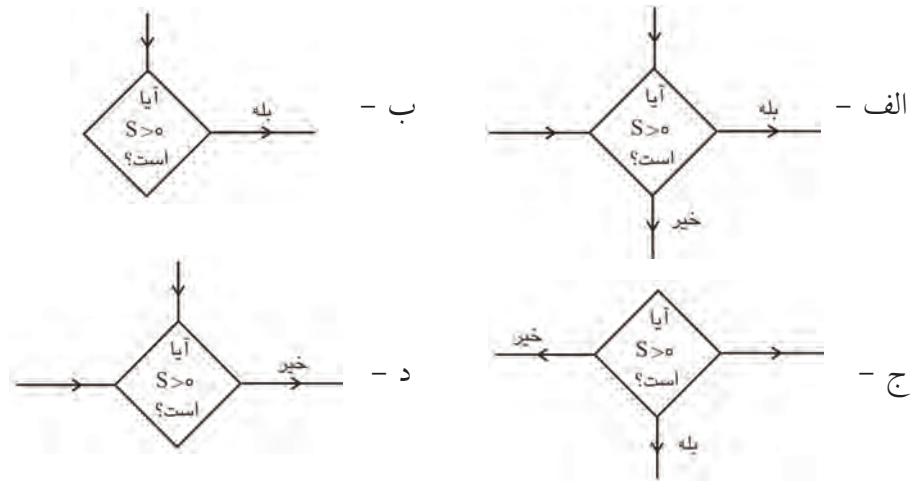
د- باقیمانده تقسیم Y بر X



- ۵ - فلوجارت روبه‌رو چه عملی را انجام می‌دهد؟
- الف- محاسبه حاصل ضرب اعداد کوچک‌تر از N
 - ب- محاسبه حاصل جمع اعداد کوچک‌تر از N
 - ج- محاسبه حاصل تفریق اعداد کوچک‌تر از N
 - د- محاسبه حاصل تقسیم اعداد کوچک‌تر از N



۶ - کدام گزینه شکل مناسب را برای دستورالعمل شرط در فلوجارت نمایش می‌دهد؟



۷ - یک فلوجارت می‌تواند حداکثر از یک استفاده کند.

الف- نقطه شروع - ب- انتساب - ج- نقطه پایانی - د- خروجی

۸ - کدام گزینه شکل مناسب برای عمل انتساب را نمایش می‌دهد؟

الف- $Z = X + Y$ - ب- $X + Y = Z$ - ج- $Z \leftarrow X + Y$ - د- $X + Y \rightarrow Z$

۹- در فلوجارت شکل  نشانه چیست؟

الف- پایان ب- محاسبه

ج- اتصال د- خروجی

10 – The flowcharts use to draw graphical representation of a process.

a- lines

b- shapes

c- rectangles

d- circles & rectangles

۱۱ - مفهوم و کاربرد فلوجارت را توضیح دهید.

۱۲ - شکل کلی نحوه استفاده از حلقه در فلوجارت را توضیح دهید.

۱۳ - شکلی را که برای شروع و پایان فلوجارت استفاده می‌شود، توضیح دهید.

۱۴ - اشکالی را که برای ورودی و خروجی مورد استفاده قرار می‌گیرند، توضیح دهید.

۱۵ - شکلی را که برای استفاده از شرط در فلوجارت استفاده می‌شود، توضیح دهید.

آزمون عملی

- ۱ - فلوچارتی رسم کنید که مضارب عدد ۵ را که کوچکتر از ۵۰۰ هستند، محاسبه کرده و نمایش دهد.
- ۲ - فلوچارتی رسم کنید که اعداد دو رقمی زوج را نمایش دهد.
- ۳ - فلوچارتی رسم کنید که میزان حافظه کامپیوتر را براساس مگابایت دریافت کرده و براساس واحدهای دیگر نمایش دهد.
- ۴ - فلوچارتی رسم کنید که ریشه معادله درجه اول یک مجهولی را محاسبه نموده و نمایش دهد.
 $ax + b = c$
- ۵ - فلوچارتی رسم کنید که ساعت کار و دستمزد هر ساعت کار یک کارگر را دریافت کرده و حقوق وی را محاسبه کند.
- ۶ - فلوچارتی رسم کنید که بتواند با شرایط زیر هزینه استفاده از اینترنت را برای یک مشترک در مدت یک هفته محاسبه کرده و نمایش دهد.
الف- هزینه ۵۰ ساعت اول هر ارتباط ۲۰ / ۰۰۰ ریال
ب- از ۵۰ ساعت به بالا به ازای هر ساعت ۲/۵۰۰ ریال
۷- فلوچارتی رسم کنید که دو عدد را دریافت کرده و عدد اول را به توان عدد دوم برساند.

توانایی درک و شناخت زبان برنامه‌نویسی ویژوال بیسیک و ایجاد یک برنامه کاربردی

هدف‌های رفتاری

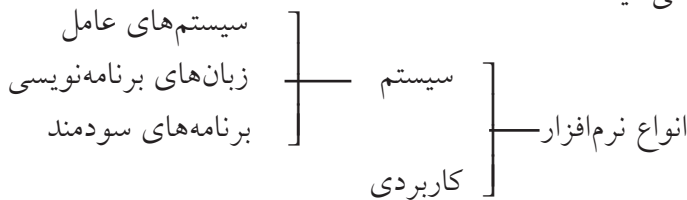
پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

- ۱- زبان برنامه‌نویسی را تعریف کند و انواع آنرا توضیح دهد.
- ۲- روش برنامه‌نویسی ساخت یافته و شیء‌گرا را توضیح دهد.
- ۳- زبان‌های برنامه‌نویسی از نوع مفسر و مترجم را توضیح داده و تفاوت آن‌ها را بیان کند.
- ۴- زبان برنامه‌نویسی ویژوال بیسیک را توضیح داده و ویژگی‌های آنرا بیان کند.
- ۵- انواع نگارش‌های زبان برنامه‌نویسی ویژوال بیسیک را توضیح دهد.
- ۶- اجزای تشکیل‌دهنده یک برنامه در ویژوال بیسیک را بیان کند.
- ۷- وارد محیط ویژوال بیسیک شده با اجزای آن کار کند.
- ۸- پروژه و فرم‌های مورد نیاز خود را ایجاد و طراحی نماید.
- ۹- خصوصیات مربوط به فرم مانند Name و Caption را توضیح داده و با کنترل‌های برچسب و کادر متن کار کند.
- ۱۰- با پنجره کدنویسی کار کرده و مفهوم رویداد را بیان کند.
- ۱۱- خصوصیات فرم و کنترل‌ها را با استفاده از کدنویسی تغییر دهد.
- ۱۲- با رویداد Load فرم و رویداد Change کنترل کادر متن کار کند.
- ۱۳- فرم و پروژه را ذخیره کرده و برای پروژه نامی تعیین کند.

کلیات

کامپیوتر از دو جزء اصلی سخت افزار و نرم افزار تشکیل شده است، برای استفاده از هر کامپیوتر لازم است تا داده‌ها و دستورالعمل‌ها برای پردازش به آن داده شود و نتیجه پردازش داده‌ها یعنی اطلاعات ارایه شود یا به عبارت دیگر کاربر بتواند با سخت افزار ارتباط برقرار کند، در این جاست که نقش نرم افزار به عنوان یکی از اجزای اصلی در کامپیوتر کاملاً قابل مشاهده است. در این واحد کار با تعریف و تاریخچه تولید و طراحی زبان‌های برنامه نویسی آشنا خواهید شد.

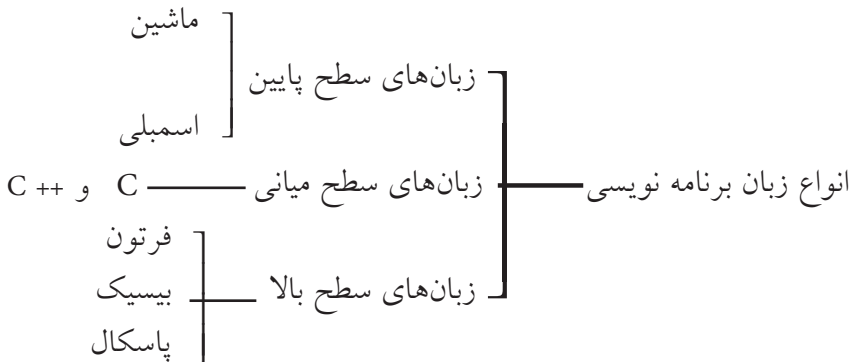
نرم افزارها مجموعه‌ای از داده‌ها و دستورالعمل‌ها هستند که به وسیله برنامه نویس و براساس قواعد مشخص نوشته می شوند و سخت افزار را قابل استفاده می کنند. نرم افزارها به دو دسته کلی سیستمی و کاربردی تقسیم می شوند، در شکل زیر انواع نرم افزارها را مشاهده می کنید:



۱-۳ تقسیم بندی زبان های برنامه نویسی

به مجموعه‌ای از قواعد و دستورالعمل‌های تعریف شده، زبان برنامه نویسی می گویند.

به طور کلی می توان زبان‌های برنامه نویسی را به صورت زیر تقسیم بندی کرد:



زبان‌های برنامه‌نویسی با توجه به نزدیکی که به زبان ماشین یا همان ۰ و ۱ دارند به سه دسته تقسیم می‌شوند؛ زبان‌های سطح پایین بیشتر به زبان ماشین نزدیک هستند و با ظهور اولین نسل از کامپیوترها این زبان برنامه‌نویسی مورد استفاده قرار گرفت که برنامه‌نویسی با آن نیز کار مشکلی است. با ساخت نسل دوم کامپیوترها، زبان دیگری به نام زبان اسمبلی به وجود آمد که این زبان نیز به زبان ماشین نزدیک بود، ولی استفاده از آن ساده‌تر از زبان ماشین است. پس از نسل دوم و ارایه نسل سوم از کامپیوترها، زبان‌های سطح میانی و سطح بالا به وجود آمدند که به زبان‌های محاوره‌ای و نوشتاری نزدیک‌تر بوده و برنامه‌نویسی با آن‌ها به مراتب راحت‌تر از زبان‌های سطح پایین می‌باشد. از آن زمان تاکنون کیفیت و کمیت زبان‌های برنامه‌نویسی تغییرات زیادی کرده است و برای تهیه برنامه‌ها در محیط‌ها و کاربردهای مختلف، زبان‌های برنامه‌نویسی متفاوتی استفاده می‌شود.

۱-۳ زبان‌های برنامه‌نویسی سطح بالا

زبان برنامه‌نویسی فرترن: یکی از اولین زبان‌های برنامه‌نویسی سطح بالاست و در برنامه‌نویسی‌های علمی که نیاز به دقت بالا در محاسبات است، از آن استفاده می‌شود. از این زبان نسخه‌هایی نیز برای برنامه‌نویسی تحت سیستم عامل ویندوز طراحی شده است.

زبان برنامه‌نویسی پاسکال: این زبان یک زبان برنامه‌نویسی سطح بالا با ویژگی ساخت یافته است و برای برنامه‌نویسی‌های علمی و تجاری مورد استفاده قرار می‌گیرد. نسخه پیشرفته آن با نام دلفی دارای ویژگی‌های برنامه‌نویسی برای محیط ویندوز است.

زبان برنامه‌نویسی C: این زبان نیز از نوع زبان‌های برنامه‌نویسی سطح بالا با ویژگی ساخت یافته است و از توانایی بالایی در سطوح مختلف برنامه‌نویسی برخوردار است. به عبارت دیگر این زبان، زبان برنامه‌نویسی سیستم است و با آن می‌توان هر نوع نرم‌افزاری را طراحی و تولید کرد.

نسخه دیگری از این زبان با نام ++C علاوه بر ویژگی‌های زبان C، دارای قابلیت برنامه‌نویسی شیء‌گرا نیز می‌باشد. نسخه‌های دیگری از این زبان با نام ++Visual C و C#.Net برای برنامه‌نویسی در محیط ویندوز طراحی و ارایه شده‌اند.

زبان برنامه نویسی ویژوال بیسیک: زبان ویژوال بیسیک در واقع حاصل توسعه و ارتقای زبان بیسیک^۱ است. بیسیک اولیه در حدود سال ۱۹۶۴ در کالج دارت موث^۲ به وسیله آقایان توماس کورتز^۳ و جان کمنی^۴ با هدف گسترش برنامه نویسی بین دانش آموزان و دانشجویان طراحی و ساخته شد. از آن زمان نسخه های متعدد و متفاوتی از آن مانند QUICK BASIC, GWBASIC, ارایه شد و همواره سعی در افزایش قابلیت های آن به عنوان یک زبان سطح بالا شده است. با ارایه سیستم عامل ویندوز ۹۵ و ۹۸، فقدان یک زبان برنامه نویسی آسان و قدرتمند برای استفاده در سیستم عامل های مذکور کاملاً مشهود بود، از این رو مایکروسافت در سال ۱۹۹۱ نسخه اول ویژوال بیسیک را با امکانات یک زبان برنامه نویسی قدرتمند و حرفه ای برای برنامه نویسی در ویندوز ارایه کرد. آخرین نسخه تکامل یافته آن برای استفاده در سیستم عامل جدید مایکروسافت نیز با نام VISUAL BASIC.NET طراحی و ارایه شده است. در این پیمانانه مهارتی نحوه برنامه سازی با زبان برنامه نویسی ویژوال بیسیک نسخه ۶ را فراخواهید گرفت، اما قبل از هر چیز به بررسی ویژگی های این زبان برنامه نویسی خواهیم پرداخت. یکی از مهم ترین ویژگی های زبان برنامه نویسی ویژوال بیسیک رابط گرافیکی آن است. رابط گرافیکی کاربر (GUI)^۵ در ویژوال بیسیک یکی از کارآمدترین رابط های گرافیکی در زمینه برنامه نویسی است که به وسیله آن می توان به آسانی برنامه های تحت سیستم عامل ویندوز را ایجاد کرده و حتی قبل از اجرا، شکل ظاهری آن را مشاهده کرد یا این که برنامه را به صورت یک مفسر یعنی به صورت خط به خط اجرا نموده و عکس العمل برنامه را بررسی کرد. البته این موارد گوشه ای از ویژگی های متعدد رابط گرافیکی ویژوال بیسیک است. توسعه سریع برنامه (RAD)^۶ یکی دیگر از ویژگی های این زبان است. منظور از توسعه سریع برنامه یا RAD در ویژوال بیسیک این است که طراحی و تولید برنامه ها در ویژوال بیسیک به دلیل وجود ابزارهای مناسب به سرعت انجام می شود، بنابراین هزینه های تولید نرم افزار به طور قابل توجهی کاهش می یابد.

ویژگی دیگر زبان برنامه نویسی ویژوال بیسیک ویژگی مدیریت رویدادها^۷ و اتفاقات می باشد. ویژوال بیسیک یکی از زبان های برنامه نویسی رویدادگر است.

1 - BASIC (Beginner's All purpose Symbolic Instruction Code)

2 - Dartmouth

3 - Thomas Kurtz

4 - John Kemeny

5 - Graphical User Interface

6 - Rapid Application Development

7 - Event Handling

مزیتی که این گونه زبان‌ها دارند در این است که برنامه‌نویس می‌تواند از قبل دستورات لازم را برای رویدادهایی که ممکن است در هنگام اجرای برنامه توسط کاربر رخ دهد، برنامه‌سازی کند. وجود محیط^۱ IDE نیز یکی از ویژگی‌های مهم این زبان است. محیط IDE به برنامه‌نویس اجازه می‌دهد تا برنامه‌های خود را به سهولت و سرعت طراحی، تولید، خطایابی و اجرا کند. این امکانات به وسیله ابزارهای متعددی که به صورت یکپارچه در رابط گرافیکی ویژوال بیسیک قرار داده شده است، قابل دسترسی است.

علاوه بر مواردی که گفته شد، دسترسی به برنامه‌های کاربردی ویندوز به وسیله توابع یکی دیگر از ویژگی‌های این زبان است. توابع API، توابع داخلی ویندوز هستند که ویژوال بیسیک را قادر می‌سازد تا با استفاده از فرامین خاصی بتواند به امکانات داخلی موجود در ویندوز دستیابی پیدا کند و برنامه‌نویس را نیز قادر می‌سازد تا در صورت نیاز با استفاده از این توابع، برنامه‌هایی را با توانایی‌های مورد نظر ایجاد کند.

یکی دیگر از جنبه‌هایی که تفاوت شگرفی بین ویژوال بیسیک و سایر نسخه‌های قبلی بیسیک ایجاد می‌کند، امکان استفاده از برنامه‌نویسی به روش شیء‌گراست. این ویژگی سبب می‌شود تا ویژوال بیسیک بتواند خواسته‌های برنامه‌نویس در رابطه با تعریف و به کارگیری اشیاء و کلاس‌های جدید را که سبب راحت‌تر شدن برنامه‌نویسی می‌شود، برطرف کند. در برنامه‌نویسی ساخت‌یافته، برنامه‌ها با استفاده از رویه‌ها به بخش‌های مختلف تقسیم می‌شوند که به صورت مجزا از هم قرار می‌گیرند. در برنامه‌نویسی شیء‌گرا با استفاده از اشیاء می‌توان مجموعه‌ای از دستورالعمل‌ها و داده‌ها را در عنصر واحدی به نام شیء قرار داد و در زمان مورد نظر از هر یک از بخش‌های شیء مربوطه استفاده کرد.

ویژگی دیگری که در نحوه کار با یک زبان برنامه‌نویسی مد نظر قرار می‌گیرد نحوه کشف، تصحیح و برخورد با اشتباهات و خطاهایی است که در هنگام طراحی یا اجرای برنامه‌ها رخ می‌دهد. ویژوال بیسیک علاوه بر این که امکانات بسیار مناسبی در زمینه کشف خطاهای نوشتاری و منطقی برنامه در اختیار برنامه‌نویس می‌گذارد، به وی امکان می‌دهد با استفاده از فرامین مناسب، خطاهای غیرقابل‌پیش‌بینی را نیز در هنگام اجرا تشخیص داده و نحوه ارایه راه‌حل مناسب را برای راهنمایی کاربران در اختیار آنان قرار دهد.

نگارش حرفه‌ای ویژوال بیسیک علاوه بر ویژگی‌های نگارش آموزشی، امکان استفاده از کنترل‌های مربوط به بانک‌های اطلاعاتی، طراحی کنترل‌های ActiveX و هم‌چنین

به کارگیری و یزارددهای مناسب برای تسهیل امر برنامه نویسی را نیز در اختیار برنامه نویسان قرار می دهد و در آن امکان برنامه نویسی تحت شبکه برقراری ارتباط با بانک های اطلاعاتی را دارد.

۲ - ۳ اجزای تشکیل دهنده یک برنامه

آنچه که قبل از طراحی و ایجاد یک برنامه کاربردی لازم است بدانید اجزای تشکیل دهنده یک برنامه است. در واقع اصلی ترین جزء تشکیل دهنده یک برنامه کاربردی در ویژوال بیسیک، پروژه (Project) است که از اجزای کوچک تری نظیر فرم ها، ماژول کد و ماژول کلاس تشکیل می شوند. فرم ها در واقع همان پنجره ها هستند که در سیستم عامل ویندوز از اجزای اصلی یک برنامه به شمار می روند و اطلاعات مربوط به شکل ظاهری برنامه و کنترل های موجود در آن را نگهداری می کنند. در شکل ۱-۳ پنجره برنامه ماشین حساب را به همراه کنترل های موجود روی آن مشاهده می کنید. از کنترل ها به منظور هدایت عملکرد برنامه، انجام عملیات و تنظیمات مورد نظر در برنامه ها استفاده می شود، عناصری مانند دکمه های فرمان، کادرهای متن، کادرهای لیست، دکمه های انتخاب و کادرهای علامت نمونه هایی از کنترل هستند. ماژول های کد، قطعاتی شامل کد هستند که دستورالعمل های مورد نیاز برنامه در آن ها قرار دارد. ماژول های کلاس نیز مانند ماژول های کد شامل قطعات کد هستند و از آن ها در برنامه نویسی شیء گرا برای تعریف انواع کلاس ها استفاده می شود.



شکل ۱-۳



نکته یک برنامه کاربردی می‌تواند از چند پروژه تشکیل شود.

۳-۳ نحوه اجرای برنامه ویژوال بیسیک و معرفی اجزای موجود در آن

قبل از هرگونه اقدام برای طراحی یک پروژه لازم است نحوه اجرای برنامه ویژوال بیسیک و اجزای موجود در پنجره آن را فرا بگیرید، به این منظور مراحل زیر را به ترتیب اجرا کنید:

برای اجرای برنامه ویژوال بیسیک از طریق منوی Start به گروه برنامه‌ها All Programs بروید و در آنجا گزینه Microsoft Visual Studio 6.0 را انتخاب نموده و سپس از منویی که ظاهر می‌شود روی میانبر Microsoft Visual Basic 6.0 کلیک کنید. پنجره اصلی برنامه به همراه کادر محاوره New Project نمایش داده خواهد شد (شکل ۲-۳).



شکل ۲-۳

در کادر محاوره New Project سه زبانه به همراه سه دکمه و یک کادر لیست مشاهده می‌شود. در کادر محاوره New Project روی زبانه New کلیک کنید و در کادر لیست


موجود در آن آیکن Standard EXE را انتخاب کنید، سپس روی دکمه Open کلیک کنید (شکل ۲-۳). پروژه‌هایی که از نوع Standard EXE طراحی می‌شوند به شما اجازه می‌دهند که از پروژه ایجاد شده، فایل‌های اجرایی مستقل برای اجرا در محیط ویندوز طراحی کنید که به آن‌ها Application نیز می‌گویند. پس از انجام مرحله دوم، پنجره ویژوال بیسیک مطابق شکل ۳-۳ نمایش داده می‌شود.




شکل ۳-۳ پنجره طراحی برنامه در حالت Standard EXE

همان‌طور که در شکل ۳-۳ مشاهده می‌کنید در پنجره ویژوال بیسیک اجزا و بخش‌های مختلفی وجود دارد.

پنجره طراحی فرم: به وسیله پنجره طراحی فرم که در قسمت مرکزی پنجره ویژوال بیسیک قرار دارد می‌توان تغییرات لازم را روی فرم برنامه که در داخل پنجره طراحی است، اعمال کرد.

پنجره طراحی فرم را می‌توانید با استفاده از دکمه View Object  در پنجره پروژه، فشردن کلید ترکیبی Shift+F7 یا انتخاب گزینه Object از منوی View فعال کنید.


پنجره پروژه (Project Explorer): در قسمت بالا و سمت راست پنجره ویژوال بیسیک پنجره یا مرورگر پروژه قرار دارد که اجزای تشکیل دهنده برنامه مانند پروژه‌ها و فرم‌ها

را به صورت ساختار درختی نمایش می‌دهد. با استفاده از این بخش برنامه‌نویس علاوه بر مشاهده نمای کلی از اجزای تشکیل دهنده برنامه خود، توانایی دسترسی، ویرایش، ذخیره‌سازی یا حذف هر یک از اجزا را نیز دارد. پنجره پروژه را می‌توانید با استفاده از دکمه  Project Explorer در نوار ابزار استاندارد، فشردن کلید ترکیبی Ctrl+R یا انتخاب گزینه Project Explorer از منوی View فعال کنید.

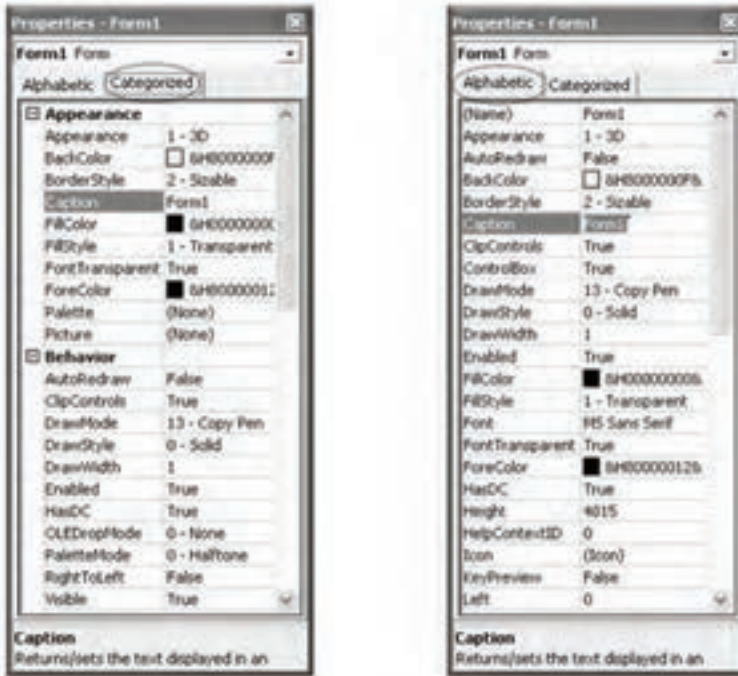
پنجره خصوصیات (Properties Window): در زیر پنجره پروژه، پنجره خصوصیات قرار گرفته است که به وسیله آن می‌توانید خصوصیات و ویژگی‌های مربوط به فرم‌ها و کنترل‌ها را مشاهده و تنظیم کنید.

خصوصیت، یک ویژگی از فرم یا کنترل است که می‌توانید با تنظیم این ویژگی، فرم و کنترل را مطابق میل خود طراحی و استفاده کنید. خصوصیات و ویژگی‌های یک فرم یا کنترل می‌توانند انواع مختلفی از داده‌های عددی و غیر عددی را با توجه به وظیفه‌ای که به عهده دارند، کسب کنند.

این خصوصیات می‌توانند شکل ظاهری مثل رنگ عناوین، رنگ زمینه، ابعاد کنترل یا فرم را تنظیم کنند. همچنین می‌توانند روی عملکرد کنترل یا فرم مثل فعال بودن یا مخفی شدن آن‌ها تأثیر بگذارند که به مرور با تعداد قابل توجهی از آن‌ها به طور کامل آشنا خواهید شد.


پنجره خصوصیات را می‌توانید با استفاده از دکمه  Properties Window در نوار ابزار استاندارد، فشردن کلید F4 یا انتخاب گزینه Properties Window از منوی View فعال کنید.

در پنجره خصوصیات با استفاده از دکمه‌های Alphabetic و Categorized می‌توانید نحوه نمایش اسامی خصوصیات را به صورت الفبایی یا موضوعی تعیین کنید (شکل ۳-۴). وقتی یک خصوصیت را انتخاب می‌کنید در قسمت پایینی پنجره خصوصیات توضیحات خلاصه‌ای در رابطه با خصوصیت انتخاب شده ارائه می‌شود.




شکل ۳-۴

پنجره تعیین موقعیت (Form Layout Window): به طور معمول در زیر پنجره خصوصیات قرار دارد و موقعیت نمایش پنجره برنامه را روی دسکتاپ و در زمان اجرا نشان می‌دهد؛ به علاوه با استفاده از این پنجره می‌توانید موقعیت پنجره‌های برنامه را روی دسکتاپ تنظیم کنید.

پنجره تعیین موقعیت را می‌توانید با استفاده از دکمه  در نوار ابزار استاندارد یا انتخاب گزینه Form Layout Window از منوی View فعال کنید.

جعبه‌ابزار (Toolbox): در سمت چپ پنجره ویژوال بیسیک و طراحی فرم، جعبه ابزار (Toolbox) حاوی آیکن کنترل‌های مختلف وجود دارد.

جعبه ابزار را می‌توانید با استفاده از دکمه  در نوار ابزار استاندارد یا انتخاب گزینه Toolbox از منوی View فعال کنید.

 **مثال ۱:** یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۳-۵ و جدول ۳-۱ طراحی کنید که یک پنجره معمولی را روی دسکتاپ نمایش دهد.

جدول ۳-۱

| مقدار | خصوصیت |
|-------|--------|
| ۳۶۰۰ | Height |
| ۵۲۵۰ | Width |



شکل ۳-۵

۱ - برنامه ویژوال بیسیک را اجرا کنید و در کادرمحاوره New Project آیکن Standard

EXE را انتخاب کنید، سپس روی دکمه Open کلیک کنید.

۲ - در پنجره پروژه روی علامت **+** در کنار پوشه Forms کلیک کنید تا آیکن فرم برنامه نمایان شود، سپس روی آیکن فرم در پنجره پروژه دابل کلیک کنید تا پنجره طراحی فرم نمایش داده شود.

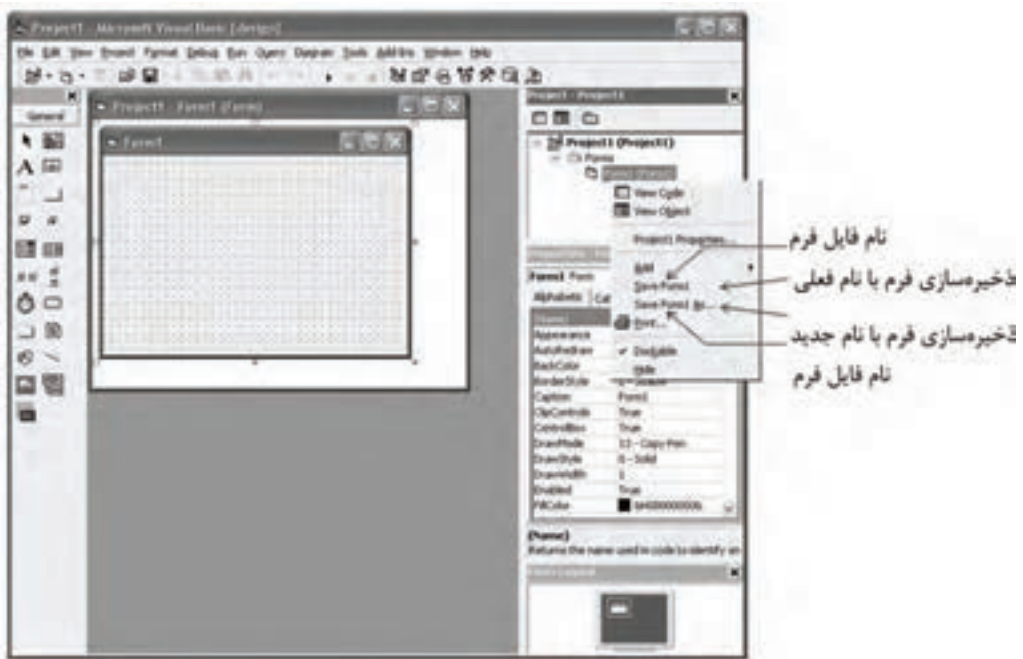
۳ - برای تنظیم عرض و ارتفاع فرم به پنجره خصوصیات بروید و در این پنجره خصوصیات Height را پیدا کنید و پس از کلیک روی نام خصوصیت مقدار ۳۶۰۰ را تایپ کنید، به همین صورت مقدار خصوصیت Width را روی ۵۲۵۰ تنظیم کنید (جدول ۳-۱). البته برای تنظیم عرض و ارتفاع فرم می‌توانید از عمل Drag روی گیره‌های مربعی شکل که در اطراف فرم قرار دارند، استفاده کنید و ابعاد فرم را در انتهای نوار ابزار استاندارد مشاهده کنید (شکل ۳-۶).



شکل ۳-۶

۴- برای ذخیره‌سازی فرم در نوار منوی پنجره ویژوال بیسیک ابتدا روی منوی File و سپس گزینه Save Form1 As... کلیک کنید. کادرمحاوره Save File As... برای ذخیره‌سازی فرم، نمایش داده می‌شود. همان‌طور که در این کادرمحاوره مشاهده می‌کنید، پسوند فایل فرم، frm است. فرم را با نام Main در پوشه‌ای با نام MyPrograms روی دسک‌تاپ (یا هر مسیر دلخواهی که مایل هستید) ذخیره کنید. البته می‌توانید فرم‌ها را با یکی از این روش‌ها نیز ذخیره کنید:

الف- در پنجره پروژه، آیکن فرم را انتخاب کرده و روی آن کلیک راست کنید، سپس گزینه Save Form1 As... را برگزینید (شکل ۷-۳).



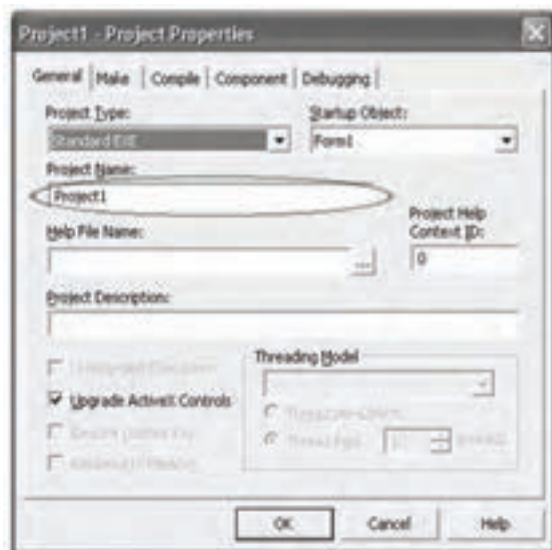
شکل ۷-۳

ب- گزینه Save Form1 را از منوی File انتخاب کنید یا از کلید ترکیبی Ctrl+S استفاده کنید.

۵- در این مرحله نامی برای پروژه تعیین کنید. برای تعیین نام پروژه در پنجره برنامه ویژوال بیسیک روی منوی Project کلیک کنید و سپس گزینه Project Properties... را برگزینید تا کادر محاوره Project Properties مطابق شکل ۸-۳ نمایش داده شود. هر پروژه یک خصوصیت Name دارد که به وسیله آن امکان دسترسی و شناسایی پروژه در زمان کدنویسی فراهم می‌شود. برای تعیین یک عبارت برای نام پروژه می‌توانید از حروف

الفبای انگلیسی، ارقام صفر تا ۹ و کاراکتر زیرخط استفاده کنید که باید با یک کاراکتر حرفی آغاز شود زیرا استفاده از کاراکترهای نقطه، فضای خالی و کاراکتر تفریق (-) مجاز نیست.

۶- در این کادر محاوره روی زبانه General کلیک کنید، سپس در کادر متن ProjectName عبارت FirstProgram را تایپ کنید و در پایان روی دکمه OK کلیک کنید.



شکل ۸-۳


۷- برای ذخیره‌سازی پروژه، از منوی File گزینه Save Project As... را برگزینید.

۸- با استفاده از کادر محاوره Save Project As فایل پروژه را با نام MyProject در پوشه MyPrograms روی دسک‌تاپ یا هر مسیر دلخواهی ذخیره کنید. همان‌طور که در این کادر محاوره می‌بینید، پسوند فایل‌های پروژه vbp است. فایل پروژه اطلاعات اجزای تشکیل‌دهنده پروژه را روی دیسک با نام فایلی که تعیین می‌شود، نگهداری می‌کند. زمانی که لازم باشد پروژه مورد استفاده قرار بگیرد با انتخاب و بازکردن فایل پروژه، فایل‌های مربوط به فرم‌ها و ماژول‌های تشکیل‌دهنده پروژه نیز به‌طور خودکار باز می‌شوند.

البته می‌توانید پروژه‌ها را با یکی از این روش‌ها نیز ذخیره کنید:

الف- در پنجره پروژه آیکن پروژه را انتخاب کرده و روی آن کلیک راست کنید، سپس گزینه Save Project را برگزینید.

ب- گزینه Save Project را از منوی File انتخاب کنید.

ج- روی دکمه Save  در نوار ابزار استاندارد کلیک کنید.

نکته در زمان ذخیره کردن پروژه و در صورتی که فرم را قبلاً ذخیره نکرده باشید، ابتدا کادر محاوره ذخیره‌سازی فرم نمایش داده می‌شود.

۹- تا این مرحله توانستید یک پروژه را به همراه یک فرم ساده ایجاد کنید. اکنون پنجره ویژوال بیسیک را با استفاده از دکمه Close ببندید. البته می‌توانید گزینه Exit را از منوی File نیز انتخاب کنید.

تمرین:

یک پروژه از نوع Standard EXE را که شامل یک فرم مطابق جدول ۲-۳ باشد، طراحی کنید سپس فرم و پروژه را به ترتیب با نام‌های Myform و Test ذخیره کنید. در انتها نام پروژه را First Project بگذارید.

مثال ۲: اکنون عبارت VISUAL BASIC 6.0 را روی فرمی که در مثال قبل ایجاد کردید، مطابق شکل ۹-۳ و جدول ۳-۳ نمایش دهید. به این منظور مراحل بعد را به ترتیب انجام دهید:

جدول ۳-۳

| کنترل خصوصیت | Label |
|-----------------|----------------|
| Name | lblmessage |
| Height | ۶۱۵ |
| Left | ۹۶۰ |
| Top | ۱۴۴۰ |
| Width | ۲۶۵۵ |
| Caption | VISUALBASIC6,0 |
| Alignment | 2-Center |

جدول ۳-۲ خصوصیات فرم

| نام خصوصیت | مقدار |
|------------|------------|
| Height | ۲۵۰۰ |
| Left | ۳۵۰۰ |
| Top | ۳۰۰۰ |
| Width | ۴۵۰۰ |
| Caption | First Form |



شکل ۹-۳

۱ - برای نمایش یک پیام روی فرم می‌توانید از کنترل برچسب (Label) استفاده کنید. برای قراردادن کنترل روی فرم، ابتدا روی آیکن **A** در جعبه ابزار کلیک کنید، سپس اشاره‌گر ماوس را روی فرم قرار داده و با استفاده از عمل درگ، کنترل را روی فرم قرار دهید. البته برای قرار دادن یک کنترل روی فرم می‌توانید در جعبه ابزار روی آیکن کنترل موردنظر دابل کلیک کنید، سپس در پنجره خصوصیات و با استفاده از خصوصیات Width و Height عرض و ارتفاع کنترل را روی مقادیر ۲۶۵۵ و ۶۱۵ تنظیم کنید (شکل ۱۰-۳). هنگام ایجاد کنترل می‌توانید عرض، ارتفاع و مختصات محل قرارگیری کنترل را در انتهای نوار ابزار استاندارد مشاهده کنید. در ضمن می‌توانید عرض و ارتفاع کنترل را با استفاده از عمل Drag روی گیره‌های مربعی اطراف کنترل نیز تنظیم کنید.


نکته برای قرار دادن کنترل‌ها روی فرم می‌توانید روی آیکن کنترل مورد نظر در جعبه ابزار ویژوال بیسیک دابل کلیک کنید.



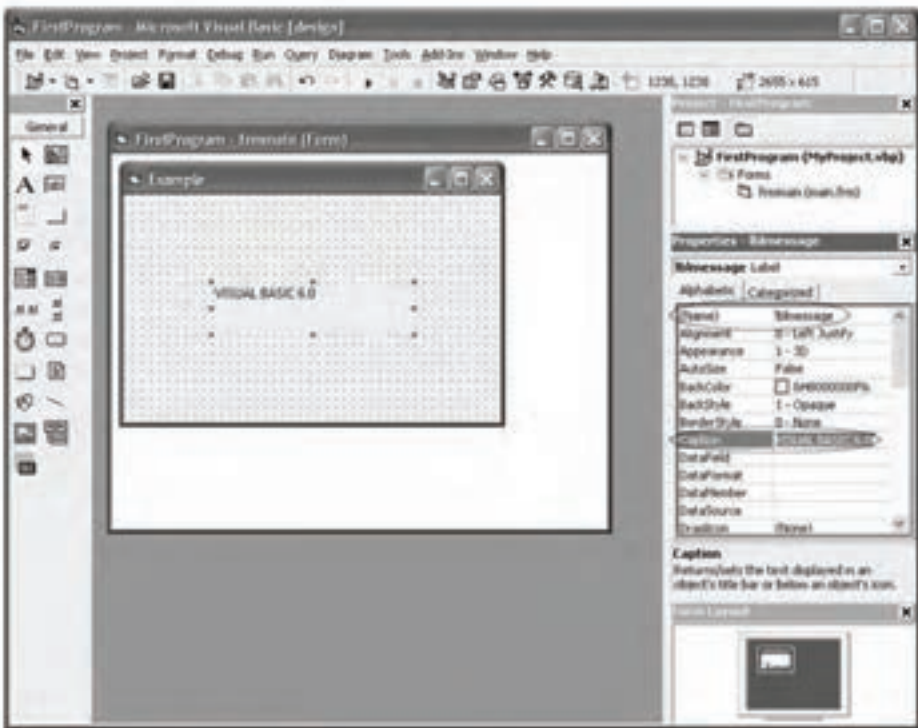
شکل ۱۰-۳

نکته برای تنظیم محل قرارگرفتن یک کنترل می‌توانید از خصوصیات Top و Left در پنجره خصوصیات استفاده کنید.

۲- در پنجره خصوصیات روی خصوصیت Name کنترل برچسب کلیک کنید و در کادر متن روبه‌روی آن عبارت lblmessage را به عنوان نام کنترل برچسب تایپ کنید و سپس کلید Enter را بفشارید (شکل ۱۱-۳).

 نکته توصیه می‌شود در نام‌گذاری کنترل برچسب از پیشوند lbl استفاده کنید.

۳- در پنجره خصوصیات روی خصوصیت Caption کنترل برچسب کلیک کرده و در کادر متن روبه‌روی آن عبارت VISUALBASIC 6.0 را تایپ کنید، همان‌طور که می‌بینید خصوصیت Caption این کنترل عبارتی را که نمایش داده می‌شود، نگهداری می‌کند (شکل ۱۱-۳).




شکل ۱۱-۳

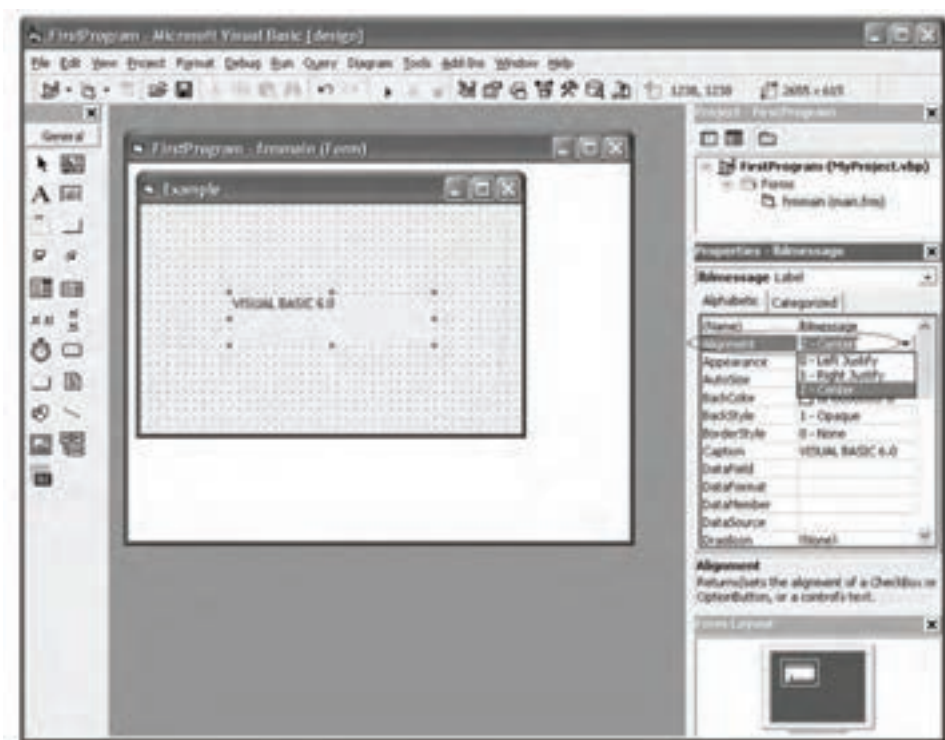
۴- پس از انجام مرحله قبل همان‌طور که می‌بینید عبارت به شکل مناسبی در داخل کنترل نمایش داده نمی‌شود، بنابراین برای آن که عبارت مربوط در وسط کنترل قرار گیرد از خصوصیت Alignment استفاده کنید. خصوصیت Alignment اجازه می‌دهد تا تراز عبارتی را که در خصوصیت Caption نگهداری می‌شود، تعیین کنید. عبارت نمایشی

جدول ۳-۴ مقادیری که خصوصیت Alignment کسب می‌کند.


| مقدار خصوصیت | توضیح |
|-----------------|----------------------|
| ۰-Left Justify | تراز متن از سمت چپ |
| ۱-Right Justify | تراز متن از سمت راست |
| ۲-Center | تراز متن از مرکز |

می‌تواند از سمت چپ یا راست کنترل یا در وسط آن نمایش داده شود، مقادیری که این خصوصیت می‌تواند کسب کند در جدول ۳-۴ ارایه شده است.



۵- در کادر لیست موجود در پنجره خصوصیات نام کنترل برچسب Iblmessage را انتخاب کنید و خصوصیت Alignment را پیدا کرده و روی آن کلیک کنید، سپس روی دکمه  روبروی این خصوصیت کلیک کنید و گزینه Center را برگزینید (شکل ۳-۱۲).



شکل ۳-۱۲

۶- ارتفاع کنترل را به وسیله گیره‌های اطراف آن روی مقدار ۲۵۵ تنظیم کنید.
 ۷- با استفاده از دکمه Start  در نوار ابزار استاندارد، برنامه را اجرا کرده و نتیجه را بررسی کنید.

برای اجرای برنامه می‌توانید از این روش‌ها نیز استفاده کنید:

- فشردن کلید F5
- گزینه Start را از منوی Run در نوار منوی ویژوال بیسیک انتخاب کنید.
- ۸ - با استفاده از دکمه  در پنجره برنامه اجرای برنامه را خاتمه دهید و به محیط طراحی برنامه بازگردید.
- برای پایان دادن به اجرای یک برنامه می‌توانید از این روش‌ها نیز استفاده کنید:
- در نوار ابزار استاندارد روی دکمه  top کلیک کنید.
- گزینه End را از منوی Run در نوار منوی ویژوال بیسیک انتخاب کنید.
- ۹ - تغییرات ایجاد شده در پروژه و فرم را ذخیره کنید و از برنامه ویژوال بیسیک خارج شوید.

تمرین:



یک پروژه از نوع Standard EXE را به همراه یک فرم طراحی کنید تا نام مدرسه خود را با تراز راست روی فرم، نمایش دهید.




مثال ۳: پروژه MyProject را که در مثال قبل ایجاد کردید به گونه‌ای تنظیم کنید که در هنگام نمایش فرم، رنگ عبارت نمایشی قرمز شود، به این منظور عملیات زیر را به ترتیب انجام دهید:

- ۱ - برنامه ویژوال بیسیک را اجرا کنید تا کادر محاوره New Project نمایش داده شود.
- ۲ - در کادر محاوره New Project روی زبانه Recent کلیک کنید. با انتخاب این زبانه لیستی از پروژه‌هایی را که اخیراً طراحی کرده یا تغییر داده‌اید، مشاهده خواهید کرد (شکل ۱۳-۳).

پروژه MyProject را از کادر لیست موجود در کادر محاوره انتخاب کرده سپس روی دکمه Open کلیک کنید.

۳ - پنجره طراحی فرم را فعال کنید، سپس در پنجره خصوصیات و از کادر لیست موجود در آن گزینه lblmessage Label را برگزینید.

۴ - برای تنظیم رنگ عبارت نمایشی در کنترل برچسب، خصوصیت ForeColor را از لیست اسامی خصوصیات کنترل انتخاب کنید. سپس روی دکمه  روبه‌روی نام این خصوصیت کلیک کنید (شکل ۱۴-۳).



شکل ۳-۱۴



شکل ۳-۱۳

۵ - در کادر رنگی که ظاهر می‌شود دو زبانه با عناوین System و Palette مشاهده می‌کنید که به وسیله این دو زبانه می‌توانید رنگ‌های متفاوتی را انتخاب کنید (شکل ۳-۱۴). هم‌چنین می‌توانید برای تعیین رنگ مورد نظر یک عدد مبنای شانزده را در کادر متنی که در روبه‌روی خصوصیت ForeColor قرار دارد، تایپ کنید. البته استفاده از این روش کمی دشوارتر از سایر روش‌هاست. به این دلیل استفاده از روش‌های دیگر متداول‌تر است.

۶ - روی زبانه Palette و سپس روی رنگ قرمز کلیک کنید. همان‌طور که مشاهده خواهید کرد در پنجره طراحی فرم عبارت VISUAL BASIC 6.0 به رنگ قرمز درمی‌آید.

تمرین:



پروژه‌ای طراحی کنید تا عبارت Microsoft Windows XP Home Edition را با رنگ زرد روی زمینه سیاه و روی یک فرم با عنوان Title نمایش تغییر یابد رنگ زمینه می‌توانید از خصوصیت BackColor استفاده کنید).

۴-۳ برنامه نویسی رویدادگرا

به هر یک از اتفاقاتی که در محیط ویندوز رخ می‌دهد، یک رویداد یا Event گفته می‌شود، به عنوان نمونه، هر یک از اتفاقات مانند بستن و بازکردن پنجره‌ها، کلیک، دابل کلیک، کلیک راست و فشردن کلیدهای صفحه کلید یک رویداد به شمار می‌روند؛ ویژوال بیسیک یک زبان برنامه‌نویسی رویدادگرا است که امکانات لازم را برای شناسایی و استفاده هر یک از رویدادها فراهم می‌کند.

یک رویداد، بخشی از برنامه است که به صورت دستورات عمل‌های مختلف توسط برنامه‌نویس تهیه و آماده می‌شود تا در زمانی که رویداد مربوطه اتفاق می‌افتد، این دستورات اجرا شوند. رویدادهای مربوط به فرم و کنترل‌های آن در بخش ماژول فرم قرار دارند که قسمتی از اطلاعات فرم به شمار می‌روند. در واقع یک فرم از دو بخش عمده تشکیل می‌شود بخش اول بخشی است که در پنجره طراحی فرم مشاهده می‌کنید و بخش دیگر ماژول فرم است و از رویدادهایی تشکیل می‌شود که می‌توانید از آن‌ها برای اجرای دستورات عمل‌های مورد نیاز خود استفاده کنید. فرم‌ها و کنترل‌ها رویدادهای مشابه و متفاوتی دارند که به مرور به توضیح آن‌ها می‌پردازیم.

شکل کلی یک رویداد به صورت زیر است:

() نام رویداد _ نام کنترل یا کلمه Private Sub Form

⋮
دستورات

End Sub

هر بلاک رویداد از دو بخش ابتدایی و انتهایی تشکیل می‌شود. بخش ابتدایی هر بلاک رویداد از کلمه کلیدی Private Sub و سپس نام کنترل یا کلمه Form (برای فرم‌ها) تشکیل می‌شود و در ادامه این بخش نام رویداد انتخاب شده همراه با یک خط زیر (Underline) و دو پرانتز باز و بسته قرار می‌گیرد. بخش پایانی در تمام رویدادها را کلمات کلیدی End Sub تشکیل می‌دهند و دستوراتی که در زمان رخ دادن رویداد اجرا می‌شوند در بین بخش آغازین و پایانی قرار می‌گیرند. به عنوان مثال رویداد کلیک برای یک فرم به این صورت است:

Private Sub Form_Click ()

⋮
دستورات

End Sub

دستوراتی که در این رویداد قرار می‌گیرند زمانی اجرا خواهند شد که کاربر در زمان اجرای برنامه روی فرم کلیک کند.

۵-۳ کنترل دکمه فرمان (Command Button)

این کنترل زمانی استفاده می‌شود که لازم باشد با کلیک کردن روی آن (یا فشردن کلید Enter) یک یا مجموعه‌ای از دستورات اجرا شود یا عملیاتی را متوقف کرده یا خاتمه دهد. احتمالاً با این نوع از کنترل‌ها آشنا هستید و از دکمه‌هایی مانند OK، Apply و Cancel در کادرهای محاوره (مانند Display Properties) استفاده کرده‌اید. این کنترل مانند کنترل برچسب دارای خاصیت‌های Name، Caption و Fore Color می‌باشد و رویداد Click آن از پرکاربردترین رویدادهای این کنترل است.




مثال ۴: یک پروژه از نوع Standard EXE به همراه یک فرم و یک دکمه فرمان با عنوان Show ایجاد کنید که با کلیک روی دکمه فرمان نام و نام خانوادگی شما نمایش داده شود. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE شامل یک فرم و یک کنترل دکمه فرمان و دو کنترل برچسب ایجاد کنید.

۲- اکنون باید رویداد مناسبی را برای این منظور انتخاب کنید، بنابراین در پنجره طراحی فرم روی کنترل دکمه فرمان دابل کلیک کنید تا بخش ماژول فرم فعال شود.

البته می‌توانید با استفاده یکی از این روش‌ها پنجره ماژول فرم را فعال کنید:

الف- برای فعال کردن ماژول فرم می‌توانید گزینه Code را از منوی View انتخاب کنید.

ب- برای فعال کردن ماژول فرم می‌توانید روی دکمه View Code  در پنجره پروژه کلیک کنید.

در قسمت بالای این پنجره دو کادر لیست را ملاحظه می‌کنید. در کادر لیست سمت چپ نام کنترل‌ها و فرم و در کادر لیست سمت راست نام رویدادها را مشاهده خواهید کرد. برای آماده‌سازی رویداد مورد نظر نام کنترل یا کلمه Form (برای فرم) را از کادر لیست سمت چپ و نام رویداد مورد نظر را از کادر لیست سمت راست انتخاب کنید. در این صورت بلاک‌های کد مانند شکل ۱۵-۳ نمایش داده می‌شود.

۳- در این مرحله رویداد Click دکمه فرمان در پنجره کد نمایش داده شده است. این رویداد زمانی اجرا می‌شود که عمل کلیک روی دکمه فرمان انجام شود.

۴- اکنون در رویداد Click دکمه فرمان، دستوری بنویسید که خصوصیت Caption کنترل

بر چسب را در هنگام اجرای برنامه تنظیم کند؛ بنابراین دستور بعد را در رویداد مزبور بنویسید
(شکل ۱۵-۳).

lblmessage.Caption=(«Kaveh Behrozi»)=vbRed



همان طور که ملاحظه می‌کنید برای تغییر خصوصیات فرم‌ها و کنترل‌ها با استفاده از کدنویسی می‌توانید به این شکل عمل کنید:

مقدار خصوصیت = نام خصوصیت . نام

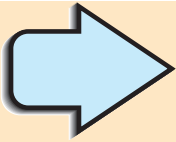
نکته در هنگام تایپ کاراکتر نقطه پس از نام کنترل، ویژوال بیسیک به طور خودکار لیستی از خصوصیات کنترل را در اختیار شما قرار می‌دهد که می‌توانید خصوصیت مورد نظر را انتخاب کرده و با فشردن کلید Spacebar آن را به پنجره کد کپی کنید (شکل ۱۵-۳).

- ۵ - پروژه را اجرا کرده و روی دکمه فرمان Show کلیک کنید.
- ۶ - اجرای پروژه را متوقف کنید و به پنجره طراحی فرم بازگردید. سپس پروژه و فرم را با نام Show name ذخیره کنید.



مثال ۵: می‌خواهیم پروژه Showname را به شکلی تنظیم کنیم تا بتواند نام و نام‌خانوادگی هر شخص دلخواهی را نمایش دهد. برای این کار عملیات زیر را به ترتیب انجام دهید:

- ۱ - برنامه ویژوال بیسیک را اجرا کرده و پروژه Showname را باز کنید.
- ۲ - یک کنترل کادر متن روی فرم قرار دهید.
- ۳ - به رویداد Click دکمه فرمان بروید و دستور آن را به این صورت تغییر دهید.
`lblmessage.Caption=txtname.Text`
- ۴ - تغییرات را ذخیره کرده و پروژه را اجرا کنید.
- ۵ - نام دلخواهی را در کادر متن تایپ کرده و روی دکمه فرمان Show کلیک کنید.
- ۶ - اجرای پروژه را متوقف کرده و به پنجره ویژوال بیسیک بازگردید.



Learn in English

Creating a Project

You begin creating this application by choosing New Project from the File menu, then selecting Standard EXE in the New Project dialog box (when you first start Visual Basic, the New Project dialog box is presented). Visual Basic creates a new project and displays a new form. To draw the interface, you use a label control, a textbox control or some of other controls.

واژه‌نامه

| | |
|------------------------------------|---|
| Alphabetic | الفبایی |
| Caption | عنوان |
| Categorized | موضوعی |
| Create | تشکیل دادن، ایجاد کردن |
| Development | توسعه |
| Dialog Box | کادر محاوره |
| Enterprise Edition | نگارش نهایی |
| Error Handling | رسیدگی به خطا |
| Event | رویداد |
| Event Handling | رسیدگی به اتفاق (رویدادگر) |
| High Level Languages | زبان‌های سطح بالا |
| Environment Integrated Development | محیط برنامه‌نویسی مجتمع |
| Label Control | کنترل برچسب |
| Learning Edition | نگارش آموزشی |
| Low Level Languages | زبان‌های سطح پایین |
| Module | ماژول (به عبارت دیگر قطعه‌ای که بخشی از یک پروژه بوده و فقط حاوی دستورات عمل‌هاست). |
| Object Oriented Feature | ویژگی شیء‌گرا |
| Object Oriented Programming | برنامه‌نویسی شیء‌گرا |
| Present | ارائه کردن، معرفی نمودن |
| Private | محلی |
| Project | پروژه |
| Property | خصوصیت |
| Provide | ارائه کردن |
| Rapid | سریع، تند |
| Rapid Application Development | طراحی سریع برنامه |
| Separate | جدا کردن، جداگانه، مجزا |
| Standard EXE | برنامه اجرایی استاندارد |
| TextBox Control | کنترل کادر متن |

خلاصه مطالب

- به مجموعه‌ای از قواعد و دستورالعمل‌های تعریف شده زبان برنامه‌نویسی می‌گویند.
- زبان برنامه‌نویسی و ویژوال بیسیک از زبان‌های برنامه‌نویسی سطح بالاست که از ویژگی‌های زیر برخوردار می‌باشد:
 - *رابط گرافیکی کاربر (GUI) و محیط توسعه یافته مجتمع (IDE)
 - *امکان طراحی سریع برنامه
 - *دسترسی به برنامه‌های کاربردی ویندوز با استفاده از توابع API
 - *توانایی مدیریت رویدادها و خطاها
- در مرورگر پروژه می‌توانید اجزای تشکیل دهنده برنامه مانند پروژه‌ها، فرم‌ها و ماژول‌ها را مشاهده کنید.
- در پنجره خصوصیات می‌توانید خصوصیات فرم‌ها و کنترل‌های موجود در آن‌ها را مشاهده و تنظیم کنید.
- جعبه ابزار و ویژوال بیسیک حاوی کنترل‌های متعددی است که از آن‌ها در طراحی برنامه استفاده می‌شود.
- از کنترل برچسب (Label) برای نمایش انواع خروجی‌های متنی مانند پیام‌ها یا مقدار خصوصیات اشیا استفاده می‌شود.
- به اتفاقاتی مانند کلیک، دابل کلیک، کلیک راست، بسته شدن یک پنجره، فشرده شدن یک کلید و غیره که در محیط ویندوز و برنامه‌ها رخ می‌دهد، یک رویداد گفته می‌شود.
- هر رویه رویداد یک مجموعه از دستورات است که به صورت جداگانه در بخش کد فرم نگهداری می‌شود و زمانی که رویداد رخ می‌دهد، این دستورات انجام می‌شوند.
- هر رویه رویداد با کلمات کلیدی Private Sub به همراه نام فرم یا کنترل آغاز شده و با End Sub خاتمه می‌یابد.
- مقدار خصوصیات کنترل‌ها را علاوه بر تنظیم در پنجره خصوصیات می‌توانید با استفاده از کدنویسی به شکل زیر تنظیم کنید:

مقدار خصوصیت = نام خصوصیت . نام

- از کنترل کادر متن (TextBox) برای دریافت داده‌ها از کاربر و ورود آن‌ها به برنامه استفاده می‌شود.

آزمون نظری

- ۱- کدام گزینه از اهداف طراحی زبان بیسیک بوده است؟
الف- برنامه‌نویسی سیستمی
ب- برنامه‌نویسی آموزشی
ج- برنامه‌نویسی تجاری
د- برنامه‌نویسی علمی
- ۲- کدام گزینه زیر یکی از انواع زبان‌های سطح پایین است؟
الف- فرترن
ب- اسمبلی
ج- ++C
د- پاسکال
- ۳- کدام ویژگی در ویژوال بیسیک برنامه‌نویسی رویدادگر را میسر می‌کند؟
الف- API
ب- EVENT HANDLING
ج- ERROR HANDLING
د- OOP
- ۴- کدام یک از زبان‌های برنامه‌نویسی برای برنامه‌نویسی علمی و تجاری مناسب است؟
الف- پاسکال
ب- فرترن
ج- زبان ماشین
د- اسمبلی
- ۵- ویژگی (RAD (Rapid Application Development عبارت است از:
الف- کشف و تصحیح آسان و سریع اشتباهات
ب- طراحی سریع و آسان برنامه‌ها
ج- طراحی هم‌زمان چند برنامه
د- برنامه‌نویسی به روش OOP
- ۶- در کدام نگارش ویژوال بیسیک ۶ امکانات ویژه‌ای به منظور برنامه‌نویسی برای محیط شبکه‌های محلی و اینترنت وجود دارد؟
الف- نگارش آموزشی
ب- نگارش اولیه
ج- نگارش نهایی
د- نگارش شبکه
- ۷- کدام نگارش از زبان Basic برای برنامه‌نویسی در محیط اینترنت ارائه شده است؟
الف- GWBasic
ب- QBasic
ج- Visual Basic
د- VB Script
- ۸- کدام نوع از انواع زبان‌های Basic برای برنامه‌نویسی در برنامه‌های کاربردی ارائه شده است؟
الف- QBasic
ب- VB
ج- VBA
د- Activex
- ۹- کدام یک از گزینه‌های زیر برای برنامه‌نویسی سیستمی مناسب هستند؟
الف- فرترن
ب- ماشین
ج- C و ++C
د- بیسیک

۱۰ - برای باز کردن پروژه‌هایی که اخیراً مورد استفاده قرار گرفته، انتخاب کدام زبانه

در کادر محاوره New Project مناسب است؟

الف - New ب - Recent ج - Existing د - Open

۱۱ - کدام نام برای خصوصیت Name یک فرم مناسب است؟

الف - frm-show ب - frm main ج - form ۱ د - frmtest

۱۲ - پسوند فایل‌های پروژه و فرم به ترتیب عبارتند از (راست به چپ):

الف - frm. ، vbp. ب - frm. ، vbp.

ج - vbf. ، vbp. د - frm. ، vbf.

۱۳ - کدام کنترل برای دریافت داده‌ها از کاربر مناسب است؟

الف - برچسب ب - کادر متن ج - کادر علامت د - فرم

۱۴ - با استفاده از کدام خصوصیت عنوان یک فرم تعیین می‌شود؟

الف - Caption ب - Visible ج - Name د - Enabled

۱۵ - از کدام بخش در پنجره ویژوال بیسیک برای ایجاد کنترل‌ها روی فرم استفاده می‌شود؟

الف - نوار ابزار ب - جعبه ابزار ج - نوار منو د - پنجره خصوصیات

16 - The VBA language can be used for programming in

a- Excel b- Access c- Applications d- all of above

17 - Which of the following options can be used for creating a project in new project dialog box?

a- Recent b- New c- Existing d- Open

۱۸ - زبان برنامه‌نویسی را تعریف کنید سپس انواع آن‌ها را با توجه به نزدیکی آن به

زبان ماشین بیان کنید.

۱۹ - ویژگی‌های زبان برنامه‌نویسی ویژوال بیسیک را شرح دهید.

۲۰ - اجزای اصلی در یک برنامه کاربردی را نام ببرید.

۲۱ - کاربرد پنجره پروژه، خصوصیات، ابزار و تعیین موقعیت فرم را توضیح دهید.

۲۲ - مفهوم رویداد را توضیح دهید و کاربرد رویه رویداد را بیان کنید.

۲۳ - نحوه تنظیم خصوصیت در کنترل‌ها را در زمان اجرای برنامه توضیح دهید.

۲۴ - خصوصیت‌هایی که به وسیله آن‌ها می‌توان موقعیت یک کنترل و فرم را در زمان

نمایش تعیین کرد، توضیح دهید.

آزمون عملی

۱ - یک برنامه از نوع استاندارد اجرایی ایجاد کنید که حاوی یک فرم، دو کنترل برچسب و دو کنترل کادر متن بوده و بتواند نام و نام‌خانوادگی کاربر را دریافت کند و نام وی را با رنگ قرمز روی زمینه سبز و نام خانوادگی را با رنگ آبی روی زمینه سفید روی کنترل‌های برچسب نمایش دهد. در ضمن در صورت انجام عمل کلیک روی کنترل‌های برچسب رنگ زمینه و قلم جابه‌جا شود.

۲ - برنامه آزمون عملی شماره یک را به گونه‌ای تغییر دهید تا با انجام عمل کلیک روی هر کادر متن محتویات آن پاک شود.

واحد کار چهارم

هدف جزئی

توانایی تعریف انواع متغیرها، ثابت‌ها و استفاده از عملگرهای ریاضی و رشته‌ای

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

- ۱- انواع متغیرها و ثابت‌ها را بیان کند و نحوه تعریف و استفاده از آنها را توضیح دهد.
- ۲- عملگرهای ریاضی و رشته‌ای را بیان کرده، نحوه استفاده و اولویت اجرای آنها را نسبت به یکدیگر شرح دهد.
- ۳- متغیرهای عمومی و محلی را توضیح دهد و نحوه تعریف و کار با آنها را بیان کند.

کلیات

در طراحی پروژه‌های واقعی لازم است تا داده‌ها و اطلاعات جمع‌آوری شده به کامپیوتر و نرم‌افزار داده شود و پس از انجام پردازش‌هایی روی آن‌ها، نتیجه یا نتایج حاصل به کاربر ارائه شود. بنابراین در تمام مراحل ورود، محاسبه و ارائه نتایج نیاز به ذخیره‌سازی داده و اطلاعات در حافظه کامپیوتر احساس می‌شود که در این رابطه می‌توانید از حافظه اصلی و حافظه جانبی استفاده کنید. در این واحد کار نحوه استفاده از حافظه اصلی یا RAM را برای ذخیره‌سازی داده‌ها و اطلاعات فرا می‌گیرید، به علاوه برای انجام عملیات ریاضی، نحوه استفاده از انواع عملگرها را در ویژوال بیسیک می‌آموزید.

۴-۱ نحوه تعریف و استفاده از انواع متغیرها در ویژوال بیسیک

یکی از عملیاتی که در برنامه‌ها انجام می‌گیرد، ذخیره‌سازی داده و اطلاعات در حافظه کامپیوتر است. ابتدا نحوه معرفی انواع متغیرها را در ویژوال بیسیک توضیح می‌دهیم، سپس نحوه انجام این کار را همراه با یک مثال آموزش می‌دهیم.

۴-۱-۱ نحوه معرفی متغیرها در ویژوال بیسیک

متغیر، مکانی از حافظه است که برای نگهداری داده‌ها و اطلاعات حاصل از پردازش داده‌ها استفاده می‌شود، به عبارت دیگر متغیر ظرف نگهداری داده‌ها و اطلاعات است و نوع آن با توجه به داده‌ای که نگهداری می‌کنند، تعیین می‌شود. به علاوه هر متغیر یک نام ویژه دارد که به وسیله آن می‌توان داده‌ها را در حافظه ذخیره کرد یا مقدار آن را مورد دستیابی قرار داد. برای تعریف متغیرها می‌توانید از دستور Dim به این صورت استفاده کنید:

نوع داده As نام متغیر Dim

نام متغیر دلخواه بوده و می‌توانید از کاراکترهای حرفی، رقمی یا ترکیبی از آن‌ها استفاده کنید، اما بهتر است نام متغیر متناسب با مقداری که در آن ذخیره می‌شود، تعیین شده و از اسامی طولانی و نامأنوس استفاده نشود. برای نام‌گذاری یک متغیر این موارد را رعایت کنید:

- الف- نام متغیر با یک کاراکتر حرفی آغاز شود و حداکثر ۲۵۵ کاراکتر باشد.
- ب- استفاده از فاصله خالی، نقطه، کاما و نظایر آن‌ها مجاز نیست، اما استفاده از کاراکتر زیرخط (_) امکان‌پذیر است.

ج- در نام‌گذاری متغیرها از علایم و اسامی رزرو شده در ویژوال بیسیک استفاده نکنید. علایم و اسامی رزرو شده در ویژوال بیسیک علایم و عباراتی هستند که از آن‌ها برای انجام اعمال و دستورالعمل‌های خاصی استفاده می‌شود مانند علایم + و - و عباراتی مانند Form، Control، و غیره.

د- ویژوال بیسیک تفاوتی بین کاراکترهای حرفی کوچک و بزرگ قائل نمی‌شود. نوع داده یک متغیر، تعیین‌کننده نوع مقداری است که در آن ذخیره می‌شود و میزان حافظه‌ای را که برای آن اختصاص داده می‌شود و نحوه ذخیره‌شدن آن را در حافظه معین می‌کند. میزان حافظه اشغالی برای ذخیره‌سازی یک اسم با میزان حافظه اشغالی توسط یک عدد صحیح یا اعشاری متفاوت است.

به عنوان مثال یک متغیر از نوع عدد صحیح (Integer) ۲ بایت فضا اشغال می‌کند، در صورتی که یک متغیر از نوع عدد اعشاری (Single) ۴ بایت فضا نیاز دارد. بنابراین در زمان تعریف یک متغیر لازم است تا نوع داده‌ای که در آن ذخیره خواهد شد، معین شود تا ویژوال بیسیک با توجه به آن، مقدار حافظه لازم را در اختیار متغیر قرار دهد.

انواع داده‌هایی که در ویژوال بیسیک ۶ قابل استفاده هستند به همراه میزان فضای مورد نیاز و محدوده مقادیر در جدول ۱-۴ ارایه شده‌اند.

جدول ۱-۴ انواع داده‌ها در ویژوال بیسیک

| نوع داده | نوع مقدار | میزان حافظه اشغالی | محدوده مقادیر | پیشوند مناسب |
|--------------------------|--------------------|--------------------------------|---|--------------|
| Byte | اعداد صحیح | ۱ بایت | صفر تا ۲۵۵ | byt |
| Integer | اعداد صحیح | ۲ بایت | ۳۲۷۶۸- تا ۳۲۷۶۷+ | int |
| Long | اعداد صحیح | ۴ بایت | $\pm 2 \times 10^9$ | lng |
| Single | اعداد اعشاری | ۴ بایت | $\pm 3 \times 10^{38}$ تا $\pm 1 \times 10^{-45}$ | sng |
| Double | اعداد اعشاری | ۸ بایت | $\pm 5 \times 10^{-324}$ تا $\pm 1 / 8 \times 10^{308}$ | dbl |
| Currency | اعداد اعشاری | ۸ بایت | $\pm 9 \times 10^{14}$ | cur |
| (با طول ثابت) String | متن (رشته) | یک بایت برای هر کاراکتر | کاراکتر ۶۵۴۰۰ | str |
| (با طول متغیر) String | متن (رشته) | یک بایت برای هر کاراکتر | 2×10^9 کاراکتر | str |
| Boolean | منطقی | ۲ بایت | True یا False | bln |
| Date | تاریخ و ساعت | ۸ بایت | ۱/۱۰۰ / ۱ / ۱۱ الی ۱۲/۳۱/۹۹۹۹ | dem |
| Variant | هر یک از انواع فوق | با توجه به نوع داده متغیر است. | _____ | vnt |


ویژوال بیسیک از انواع داده‌های مختلفی پشتیبانی می‌کند که می‌توانند نیازهای مختلف شما را در زمان برنامه‌نویسی پوشش دهند و با استفاده از انواع داده‌های ارایه شده در جدول ۱-۵ انواع متفاوتی از متغیرهای مورد نیاز را ایجاد کنند. در این جا به توضیح انواع آن‌ها می‌پردازیم.

متغیرهای عددی

ویژوال بیسیک از چند نوع داده عددی مختلف پشتیبانی می‌کند؛ داده صحیح، اعشاری و پولی. داده صحیح شامل انواع Integer، Long و Byte می‌باشد، با انواع Integer و Long می‌توانید متغیرهایی ایجاد کنید که در محدوده اعداد صحیح باشند، مقادیری مانند ۲۲۰، ۲۰۰۵، ۷۸- و غیره. اما نوع Byte در مواردی به کار می‌رود که فقط از اعداد صحیح مثبت در محدوده صفر تا ۲۵۵ استفاده می‌کنیم. انواع داده اعشاری نیز شامل انواع Single، Double و Currency هستند و توانایی ایجاد متغیرهایی را دارند که می‌توانند هرگونه عدد اعم از صحیح، اعشاری، مثبت و منفی مانند ۱۴۲-، ۱۷/۷۵- یا ۱/۰۰۲۵ / ۴۸۹۰ را در خود جای دهند. در صورتی که نیاز به نگهداری داده‌هایی از نوع پولی داشته باشید می‌توانید از نوع داده Currency استفاده کنید. به عنوان مثال به این تعاریف توجه کنید:

| | | | |
|-----|-------------|----|-----------------------------|
| Dim | intcounter | As | Integer |
| Dim | dbltotal | As | Double |
| Dim | sngdistance | As | Single, intgrade As Integer |
| Dim | bytage | As | Byte, cursalary As Currency |

همان‌طور که مشاهده می‌کنید متغیر intcounter از نوع اعداد صحیح و dbltotal از نوع اعشاری تعریف شده‌اند. به علاوه متغیرهای sngdistance و intgrade به‌طور هم‌زمان با استفاده از یک دستور Dim تعریف شده‌اند. همین‌طور متغیرهای bytage و cursalary نیز به ترتیب از نوع Byte و پولی معرفی شده‌اند.

 **نکته** استفاده از پیشنهادهای مناسب با نوع داده در نام متغیر، شناسایی آسان متغیرها و در نتیجه خوانایی کدها را در پی خواهد داشت.

متغیرهای منطقی

این نوع متغیرها توانایی دریافت مقادیر درست (True) یا نادرست (False)، بله (Yes) یا خیر (No) را دارا می‌باشند، از این نوع متغیرها معمولاً زمانی استفاده می‌شود که داده‌های دو حالتی مانند زن یا مرد، مردود یا قبول، بیمار یا سالم و نظایر آن‌ها مورد نظر باشند به عنوان مثال برای ذخیره‌سازی وضعیت قبولی یک دانش‌آموز در مدرسه می‌توان متغیر `blnresult` را به این شکل تعریف کرد:

```
Dim blnresult As Boolean
```

متغیرهای تاریخ و زمان (Date & Time)

با استفاده از این نوع متغیرها می‌توانید مقادیر ساعت و تاریخ یا ترکیبی از این دو را ذخیره کنید. به عنوان مثال برای ذخیره‌سازی تاریخ تولد افراد می‌توانید متغیر `dtmb_Date` را به این صورت تعریف کنید:

```
Dim dtmb_Date As Date
```

متغیرهای رشته‌ای

اگر بخواهید مقادیر غیر عددی مانند متون و کاراکترها را ذخیره کنید، می‌توانید متغیرهای خود را از نوع داده رشته‌ای تعریف کنید. به عنوان مثال این نوع داده را می‌توان برای ذخیره‌سازی نام و نام خانوادگی اشخاص، نام شهر، آدرس یک محل و غیره به کار برد. متغیر رشته‌ای با طول ثابت، توانایی ذخیره‌سازی داده‌های رشته‌ای را با توجه به تعداد کاراکترهای تعیین شده دارد و برای تعریف آن از این الگو استفاده می‌شود:

```
Dim نام متغیر رشته‌ای As String*n
```

`n` عدد صحیح و مثبتی است که تعداد کاراکترها را در متغیر رشته‌ای تعیین می‌کند. به عنوان مثال برای ذخیره‌سازی نام افراد با حداکثر طول نام ۲۵ کاراکتر، از این دستور استفاده می‌شود:

```
Dim strname As String*25
```

متغیر رشته‌ای با طول متغیر، توانایی نگهداری یک تا ۲ میلیارد کاراکتر را دارد و در هنگام ذخیره‌سازی اطلاعات، فضای لازم را با توجه به طول رشته در اختیار متغیر قرار می‌دهد. برای تعریف این گونه متغیرها از این الگو استفاده می‌شود:

```
Dim نام متغیر رشته‌ای As String
```

به عنوان مثال برای ذخیره‌سازی اسامی افراد که حداکثر طول نام آن‌ها مشخص نیست، از این دستور استفاده می‌شود:

Dim strname As String

متغیرهای ترکیبی (Variant)

اگر به تغییری احتیاج دارید که نوع داده‌هایی که در آن ذخیره می‌شوند قابل پیش‌بینی نبوده و انواع متفاوتی را دربرمی‌گیرد، از این نوع متغیر استفاده کنید. این نوع متغیر تمام انواع داده‌های قبلی را دربرمی‌گیرد و می‌تواند مقادیر عددی، رشته‌ای، منطقی و تاریخی را بدون آن‌که نیازی به تبدیل آن‌ها باشد در خود جای دهد.

نکته در صورتی که نوع یک متغیر تعیین نشود، نوع متغیر به طور خودکار از نوع Variant در نظر گرفته خواهد شد.

تمرین:

کدام یک از این تعاریف اشتباه است:

Dim 1Age As Integer

Dim snggrade 1 As Single,n-no As Double

Dim sngaverage As Single,intno As Integer

جدول ۲-۴

| کاراکتر | نوع داده |
|---------|----------|
| % | Integer |
| & | Long |
| ! | Single |
| # | Double |
| @ | Currency |
| \$ | String |

علاوه بر روش‌هایی که تاکنون برای تعریف انواع متغیرها گفته شد، می‌توانید با استفاده از بعضی کاراکترها نوع متغیر را به طور ضمنی تعریف کنید. به عبارت دیگر با اضافه کردن کاراکترهای رایج شده در جدول ۲-۴ به انتهای نام متغیر، نوع آن را معین کنید. از این روش فقط می‌توانید برای داده‌های عددی و رشته‌ای استفاده کنید.

به عنوان مثال به موارد ارایه شده در جدول ۳-۴ توجه کنید:

جدول ۳-۴

| تعریف متغیرها با صورت ضمنی | تعریف متغیرها با دستور Dim |
|----------------------------|----------------------------|
| Intage % | Dim intage As Integer |
| Lngspeed& | Dim lngspeed As Long |
| sngaverage! | Dim sngaverage As Single |
| dblgrade# | Dim dblgrade As Double |
| cursalary@ | Dim cursalary As Currency |
| strname\$ | Dim strname As string |

۲-۱-۴ نحوه مقداردهی به متغیرها

پس از تعریف یک متغیر بهتر است مقداری را در آن ذخیره کنیم. به این کار انتساب مقادیر یا مقداردهی به متغیر گفته می‌شود. برای انجام این کار می‌توانید از این الگو استفاده کنید:

Let نام متغیر = expression

البته ذکر کلمه کلیدی Let اختیاری است و معمولاً از آن استفاده نمی‌شود. Expression عبارتی است که می‌تواند یکی از انواع داده‌های گفته شده، فرمول و عبارات محاسباتی، خصوصیت یک شیء مانند فرم یا کنترل و غیره باشد. مشاهده می‌کنید که دستور انتساب شبیه به مقداردهی به خصوصیات اشیا مانند فرم‌ها و کنترل است. به عنوان مثال به مواردی که در جدول ۴-۴ ارایه شده است، توجه کنید.

جدول ۴-۴

| دستور انتساب | نوع عبارت |
|---------------------------------|--------------|
| Strname="Ali" | رشته‌ای |
| int count=-4 | عددی |
| Sngave=snglength/3 | عبارت ریاضی |
| dtmb_Date=#Dec-4-90# | تاریخ |
| blntest=True | منطقی |
| dblsum=inta+intb+intc | عبارت ریاضی |
| dtmtime= # 18:05:00 PM# | ساعت |
| dtmzone= #8/10/2004 5:12:17 AM# | تاریخ و ساعت |



نکته در مقداردهی به متغیرهای منطقی می‌توانید از مقادیر یک و صفر به جای True و False استفاده کنید.

متغیرهای منطقی در هنگام تعریف به طور خودکار مقدار False، متغیرهای عددی مقدار صفر و متغیرهای رشته‌ای به صورت یک رشته خالی ("") مقداردهی اولیه می‌شوند. مقداردهی متغیرهای تاریخ و ساعت در هنگام تعریف با توجه به تنظیمات تاریخ و ساعت که به وسیله برنامه Regional and Language Options در بخش Control Panel ویندوز انجام می‌شود به دو روش صورت می‌گیرد. اگر ساعت روی حالت ۲۴ ساعته تنظیم شود مقدار اولیه به صورت ۰۰:۰۰:۰۰ یا ۰:۰۰:۰۰ خواهد بود اما وقتی ساعت روی حالت ۱۲ ساعته تنظیم شود مقدار اولیه به صورت ۱۲:۰۰:۰۰ AM می‌باشد.



مثال ۱: یک پروژه مطابق شکل ۴-۱ طراحی کنید که اطلاعات مربوط به یک کارمند شامل نام، نام خانوادگی و میزان حقوق وی را دریافت کرده، سپس با کلیک روی دکمه Calculate میزان حقوق خالص وی را پس از کسر مالیات و بیمه بر اساس فرمول‌های زیر محاسبه کرده و نمایش دهد؛ همچنین با کلیک روی دکمه New، برنامه برای دریافت اطلاعات کارمند جدید آماده شود.

۳ درصد حقوق = بیمه

۵ درصد حقوق = مالیات

شکل ۴-۱

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE شامل یک فرم و کنترل‌های کادرمتن و برچسب را مطابق شکل ۴-۱ و جدول‌های ۴-۵، ۴-۶ و ۴-۷ طراحی کنید.

جدول ۴-۵ خصوصیات فرم


| مقدار | خصوصیت |
|--------|---------|
| frmsal | Name |
| Salary | Caption |

جدول ۴-۶ خصوصیات کنترل‌ها

| کنترل خصوصیت | Label | Label | Label | Label | Label |
|-----------------|---------|---------|---------|----------|--------|
| Name | lblname | lblfam | lblsal | lblp | lblpay |
| Caption | Name: | Family: | Salary: | Payment: | o |

جدول ۴-۷ خصوصیات کنترل‌ها

| کنترل خصوصیت | TextBox | TextBox | TextBox | Command Button | Command Button | Command Button |
|-----------------|---------|---------|---------|-------------------|-------------------|-------------------|
| Name | txtname | txtfam | txtsal | cmdcalculate | cmdnew | cmdexit |
| Caption | -- | -- | -- | &Calculate | &New | &Exit |

۲- در این مرحله لازم است کنترل‌های دکمه فرمان را روی فرم قرار دهید. به این منظور در جعبه ابزار روی آیکن  (Command Button) دابل کلیک کنید پس از ایجاد دکمه‌های فرمان خصوصیت‌های آن‌ها را مطابق جدول ۴-۷ تنظیم نمایید.

۳- برای تعیین کلید ایجاد دسترسی سریع (Hot Key) در کنترل‌ها می‌توان در خصوصیت Caption و قبل از حرف موردنظر از کاراکتر & استفاده کرد بنابراین لازم است قبل از حروف C, N و E در سه دکمه فرمان ایجاد شده کاراکتر & را تایپ کنید. پس از تعیین کلید دسترسی سریع، بلافاصله در پنجره طراحی فرم زیر حروف مربوط یک خط مشاهده می‌شود.

۴- به پنجره ماژول فرم بروید و رویداد Click دکمه فرمان Calculate, New و Exit را به این صورت تنظیم کنید:

```
Private Sub cmdcalculate_Click()
```

```
Dim sngsal As Single
Dim sngins As Single
Dim sngtax As Single
Dim sngpay As Single
sngsal = Val(txtsal.Text)
sngins=sngsal *0.03
sngtax=sngsal * 0.05
sngpay= sngsal-sngins-sngtax
lblpay.Caption=sngpay
End Sub
```

در این رویداد ابتدا با استفاده از دستور Dim متغیرهای sngsal برای حقوق، sngins برای بیمه، sngtax برای مالیات و sngpay برای حقوق خالص از نوع اعشاری تعریف شده‌اند سپس مقدار حقوق که در خصوصیت Text کادر متن txtsal ذخیره شده است با استفاده از تابع Val به عدد تبدیل و در متغیر اعشاری sngsal ذخیره می‌شود تا بعداً برای محاسبه بیمه، مالیات و حقوق پرداختی از آن استفاده شود. تابع Val می‌تواند یک عبارت رشته‌ای را که شامل کاراکترهای رقمی باشد دریافت کرده و به نوع داده عددی تبدیل کند. در مرحله آخر نیز حقوق خالص وی در خصوصیت Caption کنترل برچسب lblpay ذخیره می‌شود تا روی فرم نمایش داده شود.

۵ - رویداد Click دکمه فرمان New را به این صورت تنظیم کنید:

```
Private Sub cmdnew_Click()
    txtname.Text=""
    txtfam.Text=""
    txtsal.Text=""
    lblpay.Caption = 0
    txtname.setFocus
End Sub
```

در این رویداد برای آن‌که بتوان داده‌های قبلی را در کادرهای متن پاک کرده و امکان ورود داده‌های جدید به وجود آید ابتدا محتویات کادرهای متن پاک شده و سپس مقدار

صفر در کنترل برچسب نمایش داده می‌شود سپس با استفاده از متد SetFocus، فوکوس به اولین کادر متن منتقل می‌شود تا وقتی کاربر روی دکمه New کلیک کند بلافاصله بتواند نام کارمند را وارد نماید.

متد، یک یا مجموعه عملیاتی است که روی یک فرم یا کنترل انجام می‌شود و روی آن تأثیر می‌گذارد. برای استفاده از متدها می‌توانید از شکل کلی زیر استفاده کنید:

نام متد . نام کنترل



نکته: برای دسترسی به متدهای یک فرم در ماژول فرم مربوط به آن می‌توان از نام متد به تنهایی یا همراه با عبارت me. استفاده کرد.

فوکوس امکانی است که با استفاده از ماوس یا صفحه کلید به کاربر اجازه می‌دهد به فرم یا کنترل موردنظر خود دسترسی پیدا کند. در ویندوز چندین برنامه می‌توانند به طور هم‌زمان اجرا شوند، اما کاربر فقط می‌تواند با یک برنامه کار کند یعنی برنامه‌ای که نوار عنوان پنجره آن فعال است یا به عبارت دیگر فوکوس را در اختیار دارد. در یک پنجره و فرم با چند کنترل نیز کاربر می‌تواند در هر لحظه با یک کنترل کار کند در این حالت کنترلی که فوکوس را در اختیار دارد، قابل استفاده است. برای انتقال فوکوس بین فرم‌ها می‌توانید از کلیک روی فرم مورد نظر یا کلید ترکیبی Alt+Tab و برای انتقال فوکوس از یک کنترل به کنترل دیگر از عمل کلیک یا کلید Tab استفاده کنید.

متد SetFocus یکی از متدهای مشترک بین کنترل‌ها و فرم‌هاست و اجرای آن فوکوس را به فرم یا کنترل مربوطه منتقل می‌کند.

رویداد Click دکمه فرمان Exit را به این صورت تنظیم کنید:

```
Private Sub cmdexit_Click()
```

```
Unload Me
```

```
End Sub
```

در این رویداد با استفاده از دستور Unload Me فرم برنامه از حافظه خارج می‌شود در نتیجه برنامه خاتمه می‌یابد.

۶ - پروژه و فرم را با نام salary ذخیره کنید، سپس پروژه را اجرا کرده و اطلاعات

دلخواهی را در کادرهای متن فرم تایپ کنید.

۷ - روی دکمه Calculate کلیک کنید تا نتیجه محاسبات یا به عبارت دیگر میزان حقوق خالص کارمند مربوطه نمایش داده شود.

۸ - روی دکمه New کلیک کنید و این بار مشخصات فرد دیگری را تایپ کرده و میزان حقوق وی را محاسبه نمایید.

۹ - با کلیک روی دکمه Exit به محیط طراحی بازگردید.

تمرین:



پروژه salary را به شکلی تغییر دهید تا هزینه بیمه، مالیات و حقوق دریافتی هر کارمند به صورت جداگانه، روی فرم قابل مشاهده باشد.

۲-۴ نحوه انجام عملیات ریاضی در ویژوال بیسیک

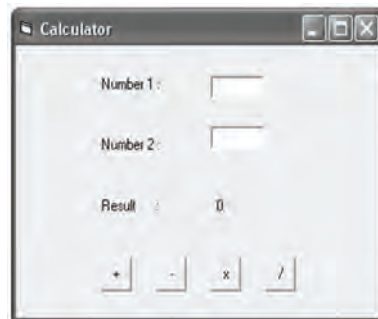
در تمام زبان‌های برنامه‌نویسی برای تبدیل انواع فرمول‌ها و انجام محاسبات، امکاناتی در نظر گرفته می‌شود. در زبان برنامه‌نویسی ویژوال بیسیک نیز برای این کار، از عملگرهای ریاضی برای انجام اعمالی چون جمع، تفریق، به توان‌رسانی و غیره استفاده می‌شود. با استفاده از این عملگرها انجام بخش عمده‌ای از عملیات محاسباتی امکان‌پذیر می‌شود.

مثال ۲: یک ماشین حساب ساده مطابق شکل ۲-۴ و جداول ۸-۴ الی ۱۰-۴ طراحی کنید که توانایی انجام عملیات ریاضی ساده را داشته باشد. به این منظور مراحل بعد را به ترتیب انجام دهید:



جدول ۸-۴ خصوصیات فرم

| مقدار | خصوصیت |
|------------|---------|
| frmcalc | Name |
| ۳۷۲۰ | Height |
| ۳۰۰ | Left |
| ۳۰۰ | Top |
| ۴۹۰۵ | Width |
| Calculator | Caption |



شکل ۲-۴

جدول ۹-۴ خصوصیات کنترل‌ها

| کنترل خصوصیت | Label | Label | Label | Label | TextBox |
|-----------------|------------|------------|----------|-----------|---------|
| Name | lblnum۱ | lblnum۲ | lblr | lblresult | txtnum۱ |
| Caption | Number ۱ : | Number ۲ : | Result : | ° | -- |

جدول ۱۰-۴ خصوصیات کنترل‌ها

| کنترل خصوصیت | TextBox | Command Button | Command Button | Command Button | Command Button |
|-----------------|---------|-------------------|-------------------|-------------------|-------------------|
| Name | txtnum۲ | cmdadd | cmddiv | cmdminus | Cmdmulti |
| Caption | -- | + | / | - | × |

۱- برنامه ویژوال بیسیک را اجرا کرده و کنترل‌های لازم را مطابق شکل ۵-۴ روی فرم ایجاد کنید.

۲- رویداد Click دکمه جمع را به این صورت تنظیم کنید:

```
Private Sub cmdadd_Click()
```

```
    lblresult.Caption=Val(txtnum1.Text) + Val(txtnum2.Text)
```

```
End Sub
```

۳- فرم و پروژه را با نام Calculator ذخیره کنید، سپس پروژه را اجرا کرده و دو عدد در کادرهای متن بنویسید؛ سپس روی دکمه جمع کلیک کنید و نتیجه را بررسی کنید، همان‌طور که می‌بینید حاصل جمع اعداد تایپ شده در بخش Result نمایش داده می‌شود.

همان‌طور که در رویداد Click دکمه جمع مشاهده کردید برای جمع دو عدد که در کادرهای متن تایپ می‌شوند از کاراکتر + استفاده می‌شود. در زبان ویژوال بیسیک برای انجام عملیات جمع از این کاراکتر استفاده می‌شود. به علاوه مقادیری که در کادرهای متن تایپ می‌شوند ماهیت غیر عددی دارند و جمع آن‌ها به صورت عادی امکان‌پذیر نیست؛

بنابراین با استفاده از تابع Val رشته عددی تایپ شده در کادرهای متن را به نوع عددی تبدیل کرده و سپس آن‌ها را با هم جمع و در خصوصیت Caption کنترل برچسب ذخیره می‌کنید تا نتیجه محاسبه نمایش داده شود.

۴ - در این مرحله رویدادهای Click دکمه‌های تفریق، ضرب و تقسیم را به این صورت و مشابه دکمه جمع تنظیم کنید:

```
Private Sub cmdminus_Click()
```

```
    lblresult.Caption=Val(txtnum1.Text)-Val(txtnum2.Text)
```

```
End Sub
```

```
Private Sub cmdmulti_Click()
```

```
    lblresult.Caption=Val(txtnum1.Text)*Val(txtnum2.Text)
```

```
End Sub
```

```
Private Sub cmddiv_Click()
```

```
    lblresult.Caption=Val(txtnum1.Text)/Val(txtnum2.Text)
```

```
End Sub
```

۵ - پروژه را اجرا کرده و عملکرد برنامه را در رابطه با چهار عمل اصلی بررسی کنید.

۶ - پنجره برنامه را ببندید و سپس تغییرات اعمال شده را ذخیره کنید.

۳-۴ نحوه تعریف و استفاده از ثابت‌ها در ویژوال بیسیک

قبلاً روش‌های مختلفی را برای ذخیره‌سازی اطلاعات فراگرفته‌اید. روش‌های دیگری نیز وجود دارد که یکی از آن‌ها استفاده از ثابت‌هاست (Constant). علاوه بر ثابت‌هایی که به طور آماده در ویژوال بیسیک وجود دارند (برای مثال، می‌توان به ثابت‌هایی که برای تعیین رنگ‌ها استفاده می‌شوند، اشاره کرد). می‌توانید ثابت‌های مورد نیاز خود را نیز در برنامه تعریف کنید. تفاوتی که بین ثابت‌ها و متغیرها وجود دارد این است که مقدار یک ثابت پس از تعریف قابل تغییر نیست، در صورتی که مقدار متغیرها را می‌توانید بارها در هنگام اجرای برنامه تغییر دهید.

برای استفاده از ثابت‌ها دلایلی نیز وجود دارد به عنوان نمونه گاهی اوقات مقادیر عددی و

رشته‌ای خاصی به طور مکرر در برنامه استفاده می‌شوند که ذکر آن‌ها توأم با صرف زمان و ایجاد اشتباه در سطح برنامه خواهد شد، اما با استفاده از یک ثابت می‌توان مقدار مورد نظر را در آن ذخیره کرد و در هر جایی از برنامه نام ثابت را به جای مقدار مربوطه به کار برد. در این صورت برنامه خواناتر شده و ایجاد تغییرات در برنامه در آینده سریع‌تر و با اشتباه کمتری همراه خواهد بود.

برای تعریف یک ثابت به این صورت عمل کنید:

مقدار = نوع ثابت As نام ثابت Const
 تعریف ثابت‌ها و قوانین نام‌گذاری آن‌ها شبیه به متغیرهاست. نوع ثابت می‌تواند از انواع داده‌ها در ویژوال بیسیک باشد که در جدول ۱-۴ ارایه شده‌اند. پس از تعیین نوع داده برای ثابت و قراردادن علامت مساوی، مقدار موردنظر برای ذخیره‌سازی در ثابت قرار می‌گیرد. مقادیر رشته‌ای را داخل علامت (") و مقادیر تاریخ و ساعت را داخل علامت (#) قرار دهید. به عنوان مثال به این نمونه‌ها توجه کنید:

```
Const Conlength As Integer = 20
Const Conname As String = "Ali"
Const ConcreateDate As Date = #12/5/98#
```



مثال ۳: یک پروژه طراحی کنید که کاربر توانایی محاسبه حجم یک استوانه دلخواه را داشته باشد.

۱ - یک پروژه از نوع Standard EXE با یک فرم و کنترل‌های مربوطه مطابق شکل ۳-۴ و جداول ۱۱-۴ الی ۱۳-۴ ایجاد کنید.

جدول ۱۱-۴ خصوصیات فرم

| خصوصیت | مقدار |
|---------|----------|
| Name | frm cyl |
| Caption | Cylinder |



شکل ۳-۴

جدول ۱۲-۴ خصوصیات کنترل‌ها

| کنترل خصوصیت | Label | Label | Label | Label |
|-----------------|----------|----------|------------|-----------|
| Name | lblr | lblh | lblc | lblresult |
| Caption | Radius : | Height : | Capacity : | o |

جدول ۱۳-۴ خصوصیات کنترل‌ها

| کنترل خصوصیت | TextBox | TextBox | Command Button |
|-----------------|---------|---------|----------------|
| Name | txtr | txth | cmdcalc |
| Caption | -- | -- | Calculate& |

۲ - خصوصیت Border Style فرم را روی مقدار Fixed Dialog 3- تنظیم کنید. به وسیله این خصوصیت می‌توان کادر دور فرم، دکمه‌های کنترلی منوی کنترل و قابلیت تغییر اندازه را در فرم تعیین کرد. این خصوصیت می‌تواند یکی از موارد ارائه شده در جدول ۱۴-۴ را کسب نماید.

جدول ۱۴-۴ خصوصیت Border Style

| مقدار خصوصیت | توضیح |
|------------------------|---|
| 0-None | فرم بدون کادر، نوارعنوان، منوی کنترل، دکمه‌های کنترلی و با اندازه ثابت |
| 1- Fixed Single | فرم با کادر، نوارعنوان، دکمه‌های کنترلی و قابلیت تغییر اندازه با دکمه‌های Maximize و Minimize |
| 2- Sizable | فرم با کادر، نوارعنوان، دکمه‌های کنترلی و قابلیت تغییر اندازه |
| 3- Fixed Dialog | فرم با کادر، منوی کنترل، نوارعنوان، بدون دکمه‌های کنترلی Minimize و Maximize و با اندازه ثابت. در ضمن در هنگام اجرای برنامه آیکن فرم در Taskbar نمایش داده نمی‌شود. |
| 4-Fixed Tool Window | فرم با کادر، نوارعنوان و دکمه Close (با اندازه کوچک) و بدون قابلیت تغییر اندازه، به علاوه در هنگام اجرای برنامه آیکن فرم در Taskbar نمایش داده نمی‌شود. |
| 5- Sizable Tool Window | فرم با کادر، نوارعنوان و دکمه Close (با اندازه کوچک) و قابلیت تغییر اندازه، به علاوه در هنگام اجرای برنامه آیکن فرم در Taskbar نمایش داده نمی‌شود. |



نکته منوی کنترل در نوارعنوان فرم‌ها در سمت چپ عنوان فرم به صورت یک آیکن کوچک نمایش داده می‌شوند و شامل گزینه‌هایی مانند Move، Size، Minimize، Maximize، Restore و Close است که برای مدیریت فرم‌ها استفاده می‌شود.

۳- رویداد Click کنترل دکمه فرمان را به این صورت تنظیم کنید:

```
Private Sub cmdcalc_Click()
```

```
Const pi As Single=3.14159265358979
```

```
lblresult.Caption=pi*Val(txtx.Text)^2*Val(txth.Text)
```

```
End Sub
```

۴- پروژه و فرم را با نام cylinder ذخیره کنید، سپس پروژه را اجرا کرده و برنامه را به ازای چند مقدار آزمایش کنید.

۵- به اجرای پروژه خاتمه دهید و به پنجره ویژوال بیسیک بازگردید.
برای استفاده از مقدار عدد π از ثابت pi استفاده شده است که موجب دقیق‌تر شدن محاسبات و تایپ راحت‌تر دستورات خواهد شد.

۴-۴ متغیرهای ایستا، محلی و عمومی در ویژوال بیسیک

در این جا لازم است انواع متغیرها را از نظر میدان دید (Scope) و شناسایی آن‌ها در بخش‌های مختلف برنامه مورد بررسی قرار دهید. میدان دید یک متغیر معین می‌کند که بعد از تعریف یک متغیر چه بخش‌هایی از برنامه می‌توانند به آن دسترسی داشته باشند و از آن استفاده کنند.

وقتی متغیری در یک رویه رویداد تعریف می‌شود فقط دستورات داخل آن رویه رویداد می‌توانند به آن دسترسی داشته باشند چنین متغیرهایی را متغیرهای محلی (Local) در سطح رویه می‌گویند. در این صورت با خروج از رویه رویداد، متغیر در حافظه پاک شده و مقدار ذخیره شده در آن از بین می‌رود. برای تعریف این نوع از متغیرها می‌توانید از دستور Dim استفاده کنید.

نوع دیگری از متغیرهای محلی نیز وجود دارند که با استفاده از کلمه کلیدی Static به جای Dim تعریف می‌شوند، این نوع متغیرها تمام ویژگی‌های متغیرهای محلی را دارند

با این تفاوت که پس از خاتمه اجرای رویه‌ای که در آن تعریف شده‌اند، مقدار آن‌ها در حافظه از بین نمی‌رود و در اجرای بعدی رویه مربوطه می‌توانید از مقادیر قبلی متغیر استفاده کنید.

به عنوان مثال به این نمونه‌ها توجه کنید:

Dim intno As Integer

Static sngk As Single

اما گاهی اوقات لازم است تا یک متغیر در تمام رویه‌های یک ماژول فرم قابل استفاده باشد، برای این کار می‌توانید متغیر مورد نظر را در بخش تعاریف ماژول فرم و با استفاده از کلمه کلیدی Private تعریف کنید. در صورت تعریف یک متغیر به صورت فوق تمام رویه‌های موجود در همان ماژول می‌توانند از متغیر مربوطه استفاده کنند و تا زمانی که فرم از حافظه خارج نشده است، مقدار خود را حفظ نمایند. شکل کلی نحوه استفاده از این نوع متغیرها به این صورت است:

نوع داده As نام متغیر Private

به عنوان مثال در صورتی که متغیرهای زیر در بخش تعاریف ماژول فرم تعریف شوند در همه سطح ماژول فرم قابل شناسایی و استفاده می‌باشد.

Private strname As String

Dim snggrasde As Single

نکته به جای استفاده از کلمه کلیدی Private می‌توانید از کلمه کلیدی Dim نیز استفاده کنید.



۴-۵ عملگرهای رشته‌ای در ویژوال بیسیک

علاوه بر عملگرهای ریاضی، مقایسه‌ای و منطقی، عملگرهای دیگری نیز وجود دارند که توانایی انجام عملیات را روی رشته‌ها فراهم می‌کنند. یکی از این عملگرها، عملگر جمع (+) است این عملگر می‌تواند دو یا چند مقدار رشته‌ای را به یکدیگر اضافه کند به عنوان مثال عبارت "Basic" + "Visual"، رشته‌ای به صورت "VisualBasic" را به وجود می‌آورد. البته این عملگر را می‌توانید برای متغیرها و

خصوصیات از نوع رشته‌ای به کار ببرید.



نکته به جای عملگر + می‌توانید از عملگر & نیز استفاده کنید.



مثال ۴: پروژه‌ای طراحی کنید که هزینه رنگ‌آمیزی یک سالن یا اتاق را با دریافت ابعاد آن‌ها و با فرض این‌که هزینه هر مترمربع ۴۰۰۰۰ ریال باشد، محاسبه نماید. برای این کار عملیات زیر را به ترتیب انجام دهید:
یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم و کنترل‌های مربوطه مطابق شکل ۴-۴ و جداول ۴-۱۵ الی ۴-۱۸ باشد؛ سپس خصوصیت BorderStyle فرم را روی مقدار 3-Fixed Dialog تنظیم کنید.
سپس رویداد Click دکمه فرمان Calculate را به این صورت تنظیم کنید:

```
Private Sub cmdcalc_Click()  
    Dim dblarea As Double  
    Dim curpay As Currency  
    dblarea=2*Val(txtl.Text)*Val(txtrh.Text)+ 2*Val(txtrw.Text)*Val(txtrh.Text)+ _  
    Val(txtl.Text)*Val(txtrw.Text)  
    curpay = dblarea * 40000  
    lblar.Caption=Str(dblarea)+ "M2"  
    lblpay.Caption=Str(curpay)+"Rials"  
End Sub
```

در این رویداد پس از محاسبه مساحت سطح و هزینه رنگ‌آمیزی شده، مقادیر آن‌ها به ترتیب در متغیرهای dblarea و curpay ذخیره می‌شوند، سپس با استفاده از کنترل‌های برچسب مساحت سطح رنگ شده و کل هزینه برای پرداخت نمایش داده می‌شود. در دستوراتی که برای نمایش نتیجه محاسبات استفاده شده‌اند، ابتدا با استفاده از تابع Str مقادیر عددی به مقادیر رشته‌ای تبدیل می‌شوند، سپس با استفاده از عملگر + آن‌ها را به هم متصل کرده و در خصوصیت Caption کنترل‌های برچسب ذخیره می‌شوند تا در روی فرم نمایش داده شوند.



شکل ۴-۴

جدول ۴-۱۵ خصوصیات فرم

| مقدار | خصوصیت |
|---------|----------|
| Name | frmpaint |
| Caption | Paint |

جدول ۴-۱۶ خصوصیات کنترل‌ها

| کنترل / خصوصیت | Label | Label | Label |
|----------------|---------------|--------------|---------------|
| Name | lblrl | lblrw | lblrh |
| Caption | Room Length : | Room Width : | Room Height : |

جدول ۴-۱۷ خصوصیات کنترل‌ها

| کنترل / خصوصیت | Label | Label | Label | Label | Command Button |
|----------------|-------|-------|-----------|---------|----------------|
| Name | lbla | lblar | lblpay | lblpayr | cmdcalc |
| Caption | Area: | • | Payment : | • | &Calculate |

جدول ۴-۱۸ خصوصیات کنترل‌ها

| کنترل / خصوصیت | TextBox | TextBox | TextBox |
|----------------|---------|---------|---------|
| Name | txtrl | txtrw | txtrh |

نکته در صورتی که خطوط مربوط به دستورات بیش از حد طولانی باشند می‌توانید آن‌ها را در چند خط تایپ کنید و آن‌ها را با کاراکتر زیرخط به هم مرتبط کنید.

۳ - پروژه و فرم را با نام Paint ذخیره و اجرا کنید، سپس اعداد ۴، ۲/۵ و ۳ متر را به ترتیب برای طول، عرض و ارتفاع وارد کنید و روی دکمه Calculate کلیک کرده و نتیجه محاسبات را بررسی کنید.

۴ - برنامه را برای مقادیر دیگری نیز آزمایش کنید، سپس از برنامه خارج شوید.

تمرین:



پروژه‌ای طراحی کنید که نام و نام خانوادگی هر فرد دلخواهی را دریافت کند سپس ترتیب آن دو را در کنار هم با یک کنترل برچسب نمایش دهد.

علاوه بر عملگر جمع، عملگر دیگری به نام Like نیز وجود دارد که توانایی انجام عملیات روی رشته‌ها را دارد. این عملگر یک رشته را با رشته دیگری مقایسه کرده و نتیجه را مشخص می‌کند. می‌توانید از این عملگر به صورت زیر استفاده کنید:

عبارت رشته‌ای دوم Like عبارت رشته‌ای اول

اگر عبارت رشته‌ای اول با عبارت رشته‌ای دوم مشابه باشد، مقدار True و اگر عبارت رشته‌ای اول با عبارت رشته‌ای دوم مشابه نباشد، مقدار False به دست می‌آید، به عنوان مثال مقایسه "ali" Like "alireza" نتیجه True و مقایسه "ali" Like "ali" نتیجه False در پی خواهد داشت. به علاوه با استفاده از کاراکترهای *، ؟، # و [] می‌توانید ترکیب‌های متنوعی از مقایسه‌ها را ایجاد کنید.

جدول ۱۹-۴

| عبارت | |
|-------------------------------|-------|
| "aBBBa" Like "a#a" | True |
| "F" Like "[A-Z]" | True |
| "F" Like "[!A-Z]" | False |
| "a۲a" Like "a#a" | True |
| "*BAT۱۲۳khg" Like "B?T" | True |
| "*CAT۱۲۳khg" Like "B?T" | False |
| "aM۵b" Like "a[L - P]#[!c-e]" | True |

کاراکتر * بیانگر گروهی از کاراکترها، کاراکتر ? بیانگر هر کاراکتر دلخواه، کاراکتر # بیانگر هر کاراکتر رقمی و کاراکترهای [] برای ایجاد بازه‌ای از کاراکترها در مقایسه به کار می‌روند. برای مشخص شدن بهتر مطلب به مثال‌هایی که در جدول ۱۹-۴ ارائه شده‌اند، توجه کنید.

همان‌طور که ملاحظه می‌کنید اولین مقایسه نتیجه True در بردارد، زیرا * می‌تواند جانشین گروهی از کاراکترها شود. در عبارت دوم حرف F در بازه حروف A تا Z قرار دارد، پس نتیجه True به دست می‌آید؛ اما در عبارت سوم از کاراکتر ! استفاده شده است و به این معنی است که F جزء حروف A تا Z نیست و نادرست است. در عبارت بعدی وجود کاراکتر # در بین حروف، پذیرش کاراکتر رقمی ۲ را در رشته "a۲a" توجیه می‌کند. عبارت‌های بعدی نیز به همین صورت قابل بررسی هستند و ترکیبی از حالت‌های گفته شده در آن‌ها به کار رفته است.

تمرین:



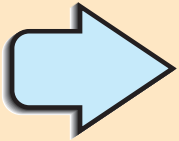
نتیجه این مقایسه‌ها را به دست آورید:

"Vb6" Like "?b#"

"Microsoft" Like "M*"

"Windows XP" Like "?i[a-z]*"

"Visual C++" Like "Visual ?[!A-Z]++"



Learn in English

Variables

In Visual Basic, you use variables to temporarily store values during the execution of an application. Variables have a name (the word you use to refer to the value the variable contains) and a data type (which determines the kind of data the variable can store).

Declaring Variables

To declare a variable is to tell the program about it in advance. You declare a variable with the Dim statement, supplying a name for the variable:

Dim variablename [**As** type]

واژه‌نامه

| | |
|------------|-----------------------|
| Constant | ثابت |
| Currency | مقادیر پولی |
| Declare | تعریف کردن |
| Determine | تعیین کردن |
| Execution | اجرا |
| Precedence | اولویت |
| Private | خصوصی |
| Public | عمومی |
| Store | ذخیره کردن |
| Supply | تأمین کردن، عرضه کردن |
| Temporary | موقت |
| Variable | متغیر |
| Variant | ترکیبی |

خلاصه مطالب

- برای انجام عملیات ریاضی از عملگرهای ریاضی استفاده می‌شود.
- انواع داده‌ها در ویژوال بیسیک عبارتند از: داده‌های عددی، رشته‌ای، منطقی، پولی، تاریخ و ساعت و داده‌های ترکیبی.
- برای تعریف متغیر از دستور Dim استفاده می‌شود.
- در صورت استفاده از دستور Option Explicit، باید متغیر را قبل از استفاده از آن تعریف کرد.
- متغیرهایی که در یک رویداد تعریف می‌شوند فقط در همان رویداد قابل شناسایی و استفاده هستند به این متغیرها، متغیرهای محلی می‌گویند.
- متغیرهای خصوصی در تمام رویدادهای ماژول فرم قابل شناسایی و استفاده هستند و برای تعریف آن‌ها از بخش تعاریف استفاده می‌شود.
- برای محاسبه زمان و اجرای دستورات مورد نظر به‌طور خودکار و مکرر در فاصله‌های زمانی معین از کنترل زمان‌سنج استفاده می‌شود.
- عملگرهای رشته‌ای در ویژوال بیسیک عبارتند از: +، & و Like.
- شکل کلی نحوه استفاده از متدها به این صورت است:

نام متد. نام شیء

آزمون نظری

- ۱ - کدام نوع از انواع داده برای تعریف یک متغیر از نوع پولی مناسب‌تر است؟
 الف - Single ب - Double ج - Currency د - Boolean
- ۲ - کدام عملگر ریاضی نسبت به سایرین از اولویت پایین‌تری برخوردار است؟
 الف - جمع ب - Mod ج - توان د - ضرب
- ۳ - کدام گزینه برای تعریف یک متغیر از نوع اعداد صحیح بلند درست است؟
 الف - % ب - & ج - ! د - #
- ۴ - کدام عملگر رشته‌ای مشابه عملگر جمع در رشته‌هاست؟
 الف - % ب - & ج - Like د - #
- ۵ - اگر متغیری در بخش تعاریف مازول فرم با دستور Dim معرفی شود متغیر
 نامیده می‌شود.
- الف - عمومی ب - محلی ج - خصوصی د - ایستا
- ۶ - کدام نوع داده برای تعریف اعداد مثبت کوچک‌تر از ۲۵۵ مناسب است؟
 الف - String ب - Integer ج - Byte د - Long
- ۷ - حاصل عبارت "Computer2004" Like "C?m?u*####" چیست؟
 الف - True ب - False
- ج - عبارت اشتباه است. د - در هنگام اجرا پیام خطا نمایش داده می‌شود.
- ۸ - متغیرهای خصوصی یا Private در چه محدوده‌ای قابل شناسایی هستند؟
 الف - فرم‌ها
 ب - مازول فرمی که در آن تعریف شده‌اند.
 ج - تمام پروژه
 د - رویدادی که در آن تعریف شده‌اند.
- ۹ - The variable can be declared with statement in within a procedure.
 a- private b- public c- dim d- as
- ۱۰ - مفهوم متغیر و نوع داده را توضیح دهید و نحوه تعریف متغیرها را بیان کنید.
- ۱۱ - تفاوت بین متغیرهای عمومی و محلی را شرح دهید.
- ۱۲ - انواع عملگرهای ریاضی و عملگرهای رشته‌ای را توضیح دهید.

آزمون عملی

۱ - پروژه‌ای طراحی کنید که با توجه به نوع سپرده‌گذاری در یک بانک، میزان بهره سالیانه را مطابق این جدول محاسبه کرده و نمایش دهد.

| نوع سپرده | میزان بهره |
|-----------------|------------|
| قرض الحسنه | — |
| کوتاه مدت | ۸ درصد |
| بلند مدت ۳ ساله | ۱۲ درصد |
| بلند مدت ۶ ساله | ۱۵ درصد |

۲ - یک پروژه از نوع Standard EXE طراحی کنید که شامل فرم به صورتی که در شکل نشان داده شده و با این شرایط باشد.

الف- نام کاربر با اندازه ۲۴ و با رنگ سیاه و ضخیم در روی فرم نمایش داده شود.

ب- کاربر بتواند با کلیک روی دکمه‌های Red، Green، و Blue رنگ قلم را در نام کاربر روی رنگ قرمز، سبز یا آبی تنظیم کند.

ج- در صورت کلیک روی دکمه Hide نام کاربر مخفی شود؛ به علاوه دکمه Hide غیرفعال و دکمه Show فعال شود.

د- در صورت کلیک روی دکمه Show نام کاربر نمایش داده شود؛ به علاوه دکمه

Show غیرفعال و دکمه Hide فعال شود.

هـ - فرم و پروژه را با نام Show ذخیره کنید.



۳- عبارات زیر را به زبان ویژوال بیسیک تبدیل کنید.

الف- $\frac{\frac{a-c}{b-d}}{5-\frac{e}{f}}$

ب- $(3x + y^2) \times (6z - 12)$

ج- $4x^2 - 5x + 3$

د- $\frac{\left(\frac{15}{7y}\right)}{9z - 18x}$

۴- عبارات زیر را به صورت ریاضی بنویسید.

الف- $1/2 * x + 14 * z^3$

ب- $x \setminus y * z / w^u$

ج- $243 - a / b^2 + 3 * z$

۵- پس از انجام دستورات زیر حاصل متغیر x چیست؟

Dim str1 As String , str2 As String

Dim x As Integer

str 1 ="34"

str2="Ali"

x=Val (str1)+Val (str2)

۶- با توجه به تمرین ۵، حاصل عبارات $str1=str1+str2$ و $str1=Str(x)+str2$ چیست؟

توانایی استفاده از دستور شرطی IF و عملگرهای مقایسه‌ای و منطقی

هدف‌های رفتاری

- پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:
- ۱- بتواند از دستور If با حالت یک دستوری و چند دستوری استفاده کند.
 - ۲- عملگرهای مقایسه‌ای و منطقی و نحوه استفاده از آنها را توضیح دهد.
 - ۳- اولویت اجرای عملگرهای مقایسه‌ای و منطقی را بیان کند.
 - ۴- یک فرم جدید را به پروژه اضافه کرده، از کنترل دکمه فرمان استفاده کند و خصوصیات مربوط به کنترل دکمه فرمان را توضیح دهد.
 - ۵- بتواند با متدهای Show و Hide فرم کار کند.
 - ۶- به کمک کنترل‌های کادر تصویر و Image تصاویر مورد نظر خود را نمایش دهد.
 - ۷- بتواند از کادرهای پیام و کادرهای ورود داده استفاده کند.

کلیات

در طراحی پروژه‌های برنامه‌نویسی معمولاً نیاز به کنترل روند برنامه خواهید داشت، چرا که یک برنامه باید در شرایط مختلف عملکرد مناسبی را از خود نشان دهد و توانایی تصمیم‌گیری را با توجه به حالات و رویدادهای متفاوتی که در هنگام اجرای برنامه رخ می‌دهند، داشته باشد. در چنین شرایطی استفاده از ساختارهای تصمیم‌اجتناب‌ناپذیر است. زبان برنامه‌نویسی ویژوال بیسیک امکانات لازم را برای کنترل روند اجرای برنامه در اختیار برنامه‌نویسان قرار می‌دهد.

۱-۵ نحوه استفاده از ساختار IF و عملگرهای مقایسه‌ای در برنامه‌ها

ساختارهای تصمیم به شما اجازه می‌دهند که نحوه اجرای برنامه را به میل خود تغییر داده و تنظیم کنید. یکی از انواع دستورات کنترلی در ویژوال بیسیک دستور If است. این دستور با بررسی شرط یا شرط‌های تعیین شده و با توجه به درست یا نادرست بودن نتیجه بررسی‌ها، روند اجرای برنامه را معین می‌کند. به‌طور کلی می‌توانید از دستور If به دو صورت یک دستوری و چند دستوری استفاده کنید. شکل کلی دستور If در حالت یک دستوری به این صورت است:

یک دستور Else یک دستور Then شرط‌ها If

در این دستور، ابتدا شرط و مقایسه مورد نظری که پس از دستور If قرار می‌گیرد، بررسی می‌شود و در صورتی که نتیجه بررسی شرط درست باشد دستوری که پس از Then قرار گرفته است، اجرا می‌شود سپس دستوری که پس از If قرار گرفته اجرا می‌شود. اما در صورتی که نتیجه بررسی شرط نادرست باشد از اجرای دستوری که بعد از Then قرار گرفته است صرف نظر کرده و دستوری که پس از Else قرار گرفته است، اجرا می‌شود. سپس دستور خطی که پس از If قرار گرفته، اجرا می‌شود. البته در صورت تمایل می‌توانید از بخش Else صرف نظر کرده و از آن استفاده نکنید، در نتیجه دستور پس از Then فقط در زمان درست بودن شرط اجرا می‌شود و در غیر این صورت بدون اجرای آن، کنترل برنامه به دستوری که پس از If قرار گرفته، منتقل می‌شود. با این حال گاهی لازم است از دستور If با حالت چند دستوری استفاده کنید. شکل کلی دستور If در این حالت در

ادامه آمده است:

If (شرط(ها)) Then

⋮

دستور(ات)

Else

⋮

دستور(ات)

End If

در این حالت نیز مانند حالت یک دستوری اگر نتیجه شرطی که پس از دستور If قرار گرفته است، درست باشد دستورات بین Then و Else اجرا می‌شوند و سپس دستوری که پس از End If قرار گرفته است، اجرا می‌شود. اما در صورتی که نتیجه بررسی شرط نادرست باشد از اجرای دستورات بین Then و Else صرف نظر می‌شود و دستورات بین بخش Else و End If اجرا می‌شوند.

اکنون دستوری که پس از End If قرار دارد، اجرا می‌شود. البته در این حالت نیز استفاده از بخش Else اختیاری است و می‌توانید از این دستور به صورت زیر نیز استفاده کنید:

If شرط‌ها Then

⋮

دستور(ات)

End If

در این حالت اگر نتیجه بررسی شرط درست باشد، دستورات بین Then و End If اجرا می‌شوند و سپس دستوری که پس از End If قرار دارد، اجرا می‌شود و اگر نتیجه این بررسی نادرست باشد از اجرای دستورات موجود بین Then و End If صرف نظر شده و اجرای برنامه به دستور پس از End If منتقل می‌شود.



مثال ۱: پروژه‌ای طراحی کنید که یک عدد دلخواه را دریافت کرده و زوج یا فرد

بودن آن را مشخص کند. برای این منظور عملیات زیر را به ترتیب انجام دهید:

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه جدید از نوع Standard EXE به همراه یک فرم و کنترل‌های آن را مطابق شکل ۱-۵ ایجاد کنید و خصوصیت‌های آن‌ها را مطابق جدول‌های

۵-۱ و ۵-۲ تنظیم نمایید:



شکل ۵-۱

جدول ۵-۱ خصوصیات فرم

| مقدار | خصوصیت |
|----------------|---------|
| Frmeven or odd | Name |
| Even or Odd | Caption |

جدول ۵-۲ خصوصیات کنترل‌ها

| کنترل خصوصیت | Label | Label | Text Box | Command Button |
|-----------------|-----------|-----------|----------|----------------|
| Name | lblno | lblresult | txtno | cmdevenodd |
| Caption | Enter No: | --- | --- | &Even or Odd |

۲- در این مرحله رویداد Click دکمه فرمان cmdevenodd را به صورت زیر تنظیم کنید:

```
Private Sub cmdevenodd_Click()
    Dim intno As Integer
    Dim R As Integer
    intno = Val (txtno. Text)
    R= intno Mod 2
    If R= 0 Then
        lblresult. Caption = " Your Number Is Even."
    Else
        lblresult. Caption = "Your Number Is Odd."
    End Sub
```

در این رویداد ابتدا رشته عددی که در کنترل کادر متن txtno تایپ می شود با استفاده از تابع Val به عدد تبدیل شده و در متغیر ذخیره می شود سپس با استفاده از عملگر Mod باقیمانده تقسیم متغیر intno را بر عدد ۲ محاسبه می کند و در متغیر R قرار می دهد.

در این مرحله دستور If شرط $R=0$ را بررسی می‌کند اگر مقدار R برابر با صفر باشد عدد زوج است و نتیجه بررسی شرط درست (True) می‌شود بنابراین پیام "Your Number Is Even." نمایش داده خواهد شد اما اگر عدد فرد باشد مقدار R برابر با ۱ خواهد شد بنابراین نتیجه بررسی شرط نادرست (False) می‌شود و پیام "Your Number Is Odd" نمایش داده می‌شود.

۳ - فرم و پروژه را با نام even or odd ذخیره کنید.

۴ - پروژه را اجرا کنید و برای اعداد زوج و فرد آن را آزمایش کنید.

۵ - اجرای پروژه را متوقف کرده و به پنجره ویژوال بیسیک بازگردید.


تمرین:



پروژه مثال ۱ را به‌گونه‌ای تنظیم کنید تا برای اعداد منفی و صفر با نمایش یک پیام خطا کاربر را مطلع نماید.

۲-۵ کنترل کادر تصویر (Picture Box)

یکی دیگر از ویژگی‌های زبان ویژوال بیسیک امکان استفاده آسان از تصاویر، فایل‌های گرافیکی و عکس‌ها در برنامه‌هاست. برای نمایش تصاویر می‌توانید از کنترل کادر تصویر استفاده کنید. این کنترل می‌تواند انواع فایل‌های گرافیکی مانند bmp, gif, jpg, ico و cur را نمایش دهد.

مهم‌ترین خصوصیت کنترل کادر تصویر Picture است. این خصوصیت مسیر و نام فایل گرافیکی یا تصویر را مشخص می‌کند. برای این کار کافی است روی دکمه  در روبه‌روی خصوصیت مزبور کلیک کنید و به‌وسیله کادر محاوره Load Picture تصویر موردنظرتان را انتخاب کنید.

کنترل کادر تصویر دارای خصوصیتی به نام Align می‌باشد. این خصوصیت می‌تواند محل نمایش تصویر را تعیین کند. مقادیری که این خصوصیت می‌تواند کسب کند در جدول ۳-۵ ارائه شده‌اند.

جدول ۳-۵ مقادیر خصوصیت Align

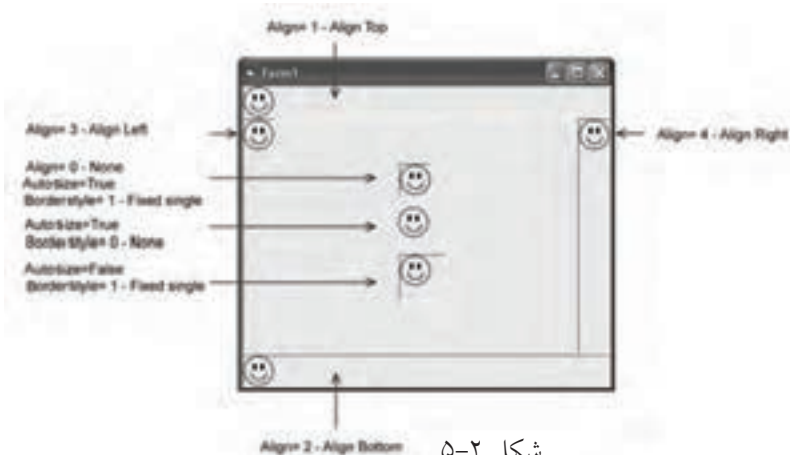
| مقدار خصوصیت | توضیح |
|----------------|--|
| 0 - None | نمایش کنترل با استفاده از مقادیر خصوصیت‌های Top و Left |
| 1- Align Top | نمایش کنترل در بالای فرم |
| 2-Align Bottom | نمایش کنترل در پایین فرم |
| 3 -Align Left | نمایش کنترل در سمت چپ فرم |
| 4 -Align Right | نمایش کنترل در سمت راست فرم |

یکی دیگر از خصوصیت‌های کادر تصویر خصوصیت AutoSize است و می‌تواند مقدار True یا False را کسب کند. در صورتی که مقدار این خصوصیت را روی True تنظیم کنید اندازه کنترل کادر تصویر با اندازه تصویر به صورت خودکار یکسان می‌شود اما اگر مقدار این خصوصیت روی False تنظیم شود اندازه کنترل کادر تصویر با توجه به مقدار خصوصیت‌های Height و Width تنظیم می‌شود. به عبارت دیگر در این حالت تصویر با اندازه خود و کنترل کادر تصویر نیز با اندازه خود نمایش داده می‌شوند.

کنترل کادر تصویر دارای خصوصیت دیگری به نام BorderStyle است. با استفاده از این خصوصیت می‌توانید کنترل کادر تصویر را با یک کادر در اطراف آن یا بدون کادر نمایش دهید. خصوصیت BorderStyle می‌تواند مقادیر ارائه شده در جدول ۴-۵ را کسب کند.

جدول ۴-۵

| مقدار خصوصیت | توضیح |
|-----------------|---|
| 0 -None | کنترل کادر تصویر بدون کادر نمایش داده می‌شود. |
| 1 -Fixed Single | کنترل کادر تصویر با کادر نمایش داده می‌شود. |



شکل ۲-۵

کنترل کادر تصویر علاوه بر رویدادهای مشترکی که با سایر کنترل‌ها دارد، دارای دو رویداد Change و Resize می‌باشد. رویداد Change وقتی رخ می‌دهد که تصویر نمایش داده شده به وسیله کنترل کادر تصویر تغییر کند و رویداد Resize نیز زمانی اجرا می‌شود که اندازه کنترل تغییر می‌کند.



مثال ۲: پروژه‌ای طراحی کنید که یک تصویر را روی یک فرم نمایش دهد. برای این منظور ابتدا فرم اصلی را طراحی کرده، آن را کامل کنید، سپس عملیات بعد را به ترتیب انجام دهید:


۱ - برنامه ویژوال بیسیک را اجرا کنید و یک پروژه جدید از نوع Standard EXE به همراه یک فرم مطابق شکل ۳-۵ ایجاد کنید که مشخصات فرم آن مطابق جدول ۵-۵ باشد.



شکل ۳-۵

جدول ۵-۵ خصوصیات فرم

| مقدار | خصوصیت |
|--------------|---------|
| frmshow | Name |
| Show Picture | Caption |

۲ - برای نمایش تصویر می‌توانید از کنترل کادر تصویر (PictureBox) استفاده کنید. برای استفاده از این کنترل ابتدا به پنجره طراحی Form1 بروید و در جعبه ابزار روی آیکن PictureBox  دابل کلیک کنید تا کنترل کادر تصویر روی فرم قرار گیرد، سپس خصوصیات کنترل کادر تصویر را مطابق جدول ۶-۵ تنظیم کنید.

جدول ۶-۵

| مقدار | خصوصیت |
|---------|----------|
| picshow | Name |
| True | AutoSize |

۳ - اکنون باید تصویر نمایشی را برای کنترل کادر تصویر انتخاب کنید، بنابراین روی کنترل کادر تصویر کلیک نمایید و در پنجره خصوصیات، خصوصیت Picture این کنترل را پیدا کنید.


۴ - روی دکمه **...** در روبه روی این خصوصیت کلیک کنید و با استفاده از کادرمحاوره Load Picture فایل EARTH.ICO را از مسیر زیر انتخاب کنید، همان طور که مشاهده خواهید کرد تصویر انتخاب شده روی فرم قابل مشاهده است.

D:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Elements

۵ - یک کنترل دکمه فرمان روی فرم قرار داده سپس ویژگی های آن را با توجه به جدول ۷-۵ تنظیم کنید. از این کنترل برای خروج از برنامه استفاده می شود.

جدول ۷-۵

| مقدار | خصوصیت |
|---------|---------|
| cmdexit | Name |
| &Exit | Caption |

 **نکته** برای آن که یک کلید دسترسی سریع برای یک کنترل ایجاد کنید، کافی است که در خصوصیت Caption آن و قبل از حرف مورد نظر یک کاراکتر & تایپ کنید، در این صورت فشردن کلید ترکیبی Alt به همراه حرفی که کاراکتر & قبل از آن تایپ شده است، معادل کلیک کردن روی کنترل دکمه فرمان خواهد بود.

۶ - اکنون باید رویداد مناسبی را انتخاب کنید تا با استفاده از آن و در زمان کلیک روی دکمه فرمان Exit، برنامه خاتمه یابد. بنابراین در پنجره طراحی فرم روی دکمه فرمان Exit دابل کلیک کنید تا پنجره کد فرم فعال شود و رویداد () cmdexit_Click در اختیار شما قرار گیرد. سپس دستور زیر را در این رویداد تایپ کنید:

Unload frmshow

در زمان اجرا وقتی کاربر روی دکمه Exit کلیک کند رویداد () cmdexit_Click به دنبال آن دستور Unload اجرا خواهد شد، این دستور اطلاعات فرم frmshow را از حافظه خارج می کند، به عبارت دیگر برنامه خاتمه می پذیرد.

۷- برنامه را اجرا کنید و پس از مشاهده فرم و تصویر نمایشی روی دکمه Exit کلیک کنید تا برنامه خاتمه یابد.

۸- پروژه و فرم را با نام showpicture ذخیره کنید.



مثال ۳: اکنون پروژه showpicture را کامل کنید تا در هنگام اجرای برنامه، ابتدا یک کلمه رمز سؤال شود و در صورت صحیح بودن کلمه رمز، اجازه ورود کاربر به برنامه داده شود و فرم اصلی برنامه با تصویر مورد نظر نمایش داده شود. برای این منظور عملیات زیر را انجام دهید:

۱- به پنجره ویژوال بیسیک و پروژه showpicture بروید.

۲- برای اضافه کردن یک فرم جدید، در مکان خالی از پنجره پروژه کلیک راست کنید و گزینه Add را انتخاب کنید، سپس روی گزینه Form کلیک کنید (شکل ۵-۵).



شکل ۴-۵



شکل ۵-۵


کادر محاوره Add Form مطابق شکل ۵-۶ نمایش داده می‌شود که دو زبانه New و Existing در آن دیده می‌شود. برای ایجاد یک فرم جدید می‌توانید از زبانه New و برای اضافه کردن یک فرم که قبلاً ایجاد و در روی دیسک ذخیره شده است، از زبانه Existing استفاده کنید.



شکل ۵-۶

البته می‌توانید فرم‌ها را با یکی از این روش‌ها نیز اضافه کنید:

الف- برای دسترسی به کادر محاوره Add Form می‌توانید گزینه Add Form را از منوی Project در نوار منوی ویژوال بیسیک انتخاب کنید.

ب- برای دسترسی به کادر محاوره Add Form می‌توانید روی علامت مثلثی شکل دکمه  در نوار ابزار استاندارد کلیک کنید و سپس گزینه Form را انتخاب کنید.

۳- در کادر محاوره Add Form ابتدا زبانه New را انتخاب کنید، سپس از کادر لیست موجود در زیرزبانه‌ها، روی آیکن Form و بعد روی دکمه Open کلیک کنید. همان‌طور که مشاهده می‌کنید یک پنجره طراحی دیگر در پنجره ویژوال بیسیک نمایش داده می‌شود به علاوه در پنجره‌های پروژه، خصوصیات و تعیین موقعیت نیز، فرم اضافه شده قابل مشاهده است.

۴- ویژگی‌های فرم جدید را تنظیم کنید و سپس آن‌را با نام Log On در روی دسک‌تاپ ذخیره کنید.

۵- اکنون باید یک کنترل کادر متن، برچسب و دکمه فرمان با مشخصات ارائه شده در جدول ۶-۹ به پنجره frmlogon اضافه کنید.

۶- حال باید عملکرد برنامه را طوری تنظیم کنید تا اگر عبارتی که در کادر متن تایپ می‌شود با کلمه رمز برابر باشد، امکان ادامه اجرای برنامه فراهم شود. برای این کار از رویداد کلیک کنترل دکمه فرمان استفاده می‌شود. این رویداد زمانی اجرا می‌شود که کاربر روی این دکمه کلیک کند یا کلید Enter را بفشارد. بنابراین کنترل cmdok را در پنجره طراحی

فرم frmlogon انتخاب کنید، سپس روی این کنترل دابل کلیک کنید. همان‌طور که مشاهده می‌کنید پنجره کد فرم نمایش داده می‌شود و رویداد cmdok_Click در دسترس شما قرار می‌گیرد. با فرض این که کلمه رمز BASIC باشد، رویداد Click این کنترل را به صورت زیر تنظیم کنید:

```
Private Sub cmdok_Click()
```

```
    If txtpass.Text = "BASIC" Then frmshow.Show
```

```
End Sub
```

در رویداد cmdok_Click از دستور If استفاده شده است. در این دستور کلمه رمزی که کاربر در کنترل کادر متن تایپ کرده و در خصوصیت Text کادر متن نگهداری می‌شود با استفاده از علامت تساوی (=) با عبارت "BASIC" که به عنوان کلمه رمز، در نظر گرفته شده است، مقایسه می‌شود. اگر کلمه رمز تایپ شده با عبارت "BASIC" برابر باشد، دستوری که بعد از کلمه کلیدی Then قرار دارد، اجرا می‌شود و در غیر این صورت یعنی اشتباه بودن کلمه رمز، دستوری که بعد از Then قرار گرفته است، اجرا نمی‌شود و دستور خط بعد از If اجرا خواهد شد.

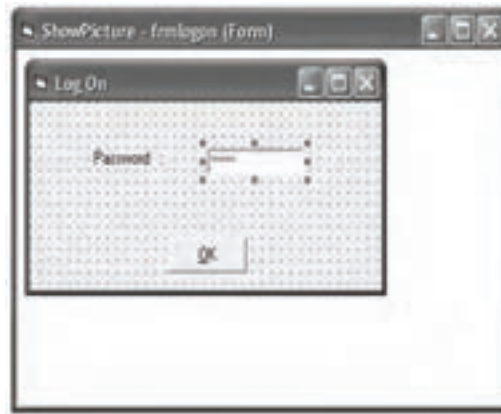
اکنون به بررسی دستور frmshow.Show که بعد از کلمه کلیدی Then قرار گرفته است، می‌پردازیم. Show متدی است که می‌تواند یک فرم را که اطلاعات آن در حافظه بارگذاری شده است، نمایش دهد. البته در صورتی که فرم در حافظه بارگذاری نشده باشد آن را بارگذاری کرده و سپس نمایش می‌دهد.



نکته متد Hide نیز از انواع متدهای فرم است و عملکرد آن برعکس متد Show می‌باشد و از نمایش فرم جلوگیری کرده و آن را مخفی می‌کند، اما اطلاعات فرم را از حافظه خارج نمی‌کند.

۷- می‌دانید که در زمان تایپ کلمه رمز در مکان مربوطه، برای جلوگیری از مشاهده کلمه رمز توسط سایر کاربران، به جای کاراکترهایی که تایپ می‌شوند کاراکتر دیگری نمایش داده می‌شود. به این منظور می‌توانید از خصوصیت PasswordChar کادر متن استفاده کنید. این خصوصیت کاراکتر تعیین شده را به جای کاراکترهایی که در کادر متن تایپ می‌شوند، نمایش می‌دهند، اما محتویات تایپ شده در کادر متن به همان صورت در

خصوصیت Text کادر متن نگهداری می‌شود. بنابراین خصوصیت فوق را پیدا کرده و در کادر متن روبه‌روی این خصوصیت کاراکتر ستاره (*) را تایپ کنید. همان‌طور که مشاهده می‌کنید پنج کاراکتر ستاره در پنجره طراحی فرم و در کادر متن نمایش داده می‌شود. به عبارت دیگر به تعداد کاراکترهای عبارت موجود در خصوصیت Text کادر متن، کاراکتر ستاره نمایش داده می‌شود (شکل ۷-۵).



شکل ۷-۵

۸- در این مرحله برای تعداد کاراکترهای کلمه رمز، محدودیت ایجاد کنید. خصوصیت MaxLength کادر متن، حداکثر تعداد کاراکترهایی را که در این کنترل می‌توان تایپ کرد، معین می‌کند. مقدار پیش فرض این خصوصیت صفر است که تعداد کاراکترهای ورودی نامحدود را پشتیبانی می‌کند. مقدار این خصوصیت را برای کنترل txtpass روی مقدار ۵ تنظیم کنید. در نتیجه هنگام اجرای برنامه می‌توانید حداکثر ۵ کاراکتر را در کادر متن تایپ کنید.

۹- دستور زیر را در رویداد Load فرم frmlogon تایپ کنید تا در هنگام نمایش این فرم کاربر مجبور نباشد مقدار پیش فرض Text_۱ را در کادر متن پاک کند و کلمه رمز را تایپ نماید.

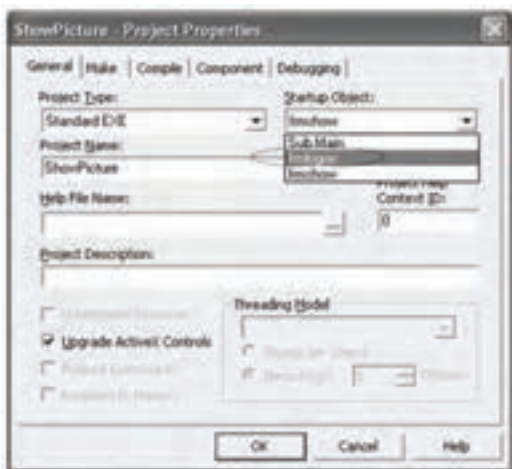
```
txtpass.Text = ""
```

۱۰- در این مرحله تغییرات ایجاد شده در فرم‌ها و پروژه را ذخیره کرده و پروژه را اجرا کنید.


همان‌طور که می‌بینید پنجره اصلی برنامه یعنی فرم frmshow نمایش داده شده و فرم frmlogon مشاهده نمی‌شود. در واقع در هر برنامه می‌توان یک فرم را به عنوان فرمی که

در ابتدای اجرای برنامه نمایش داده می‌شود، معرفی کرد. اگر از چند فرم در پروژه خود استفاده می‌کنید باید فرم مورد نظر را برای نمایش در ابتدای اجرای برنامه معین کنید.

برای این کار باید از کادر محاوره Project Properties استفاده کنید.




شکل ۵-۸

۱۱ - اجرای برنامه را متوقف کنید و از منوی Project گزینه ShowPicture Properties را انتخاب کنید تا کادر محاوره Project Properties نمایش داده شود. در زبانه General و در کادر لیست Startup Object روی دکمه  کلیک کنید (شکل ۵-۸). سپس از کادر لیستی که نمایش داده می‌شود، فرم frmlogon را برگزیده و روی دکمه OK کلیک کنید.

۱۲ - برنامه را اجرا کنید و عبارت BASIC را در کادر متن تایپ کنید، سپس روی دکمه OK کلیک کنید. همان‌طور که می‌بینید فرم frmshow نمایش داده می‌شود و هر دو فرم قابل مشاهده خواهند بود.

۱۳ - پنجره‌های برنامه را ببندید و بار دیگر برنامه را اجرا کنید و این بار کلمه رمز را به صورت basic تایپ کنید و روی دکمه OK کلیک کنید. همان‌طور که مشاهده خواهید کرد فرم برنامه نمایش داده نخواهد شد.

۱۴ - اجرای برنامه را متوقف کنید و به پنجره ویژوال بیسیک بازگردید.

 **نکته** در صورتی که بخواهید یک متغیر در تمام فرم‌های پروژه قابل دسترسی و شناسایی باشد می‌توان آن را با استفاده از کلمه کلیدی Public در بخش تعاریف ماژول فرم تعریف نمود. به این نوع از متغیرها، متغیر عمومی می‌گویند. شکل کلی نحوه تعریف متغیرهای عمومی به صورت زیر است:

نوع داده As نام متغیر Public

به عنوان مثال به این نمونه توجه کنید:

Public curpament As Currency

برای دسترسی به متغیرهای عمومی در سایر فرم‌ها می‌توان نام فرمی را که متغیر در آن تعریف شده است را قبل از نام متغیر قرار داد. به عنوان مثال اگر متغیر curpament در بخش تعاریف فرمی با نام frmmain تعریف شده باشد برای مقداردهی به آن در فرم frmmain در همان پروژه می‌توان به این صورت عمل کرد:

frmmain. Curpament = 2000

تمرین:



۱ - برنامه را به شکلی تنظیم کنید که با کلیک روی دکمه Exit تمام پنجره‌ها به‌طور هم‌زمان بسته شوند.

۲ - برنامه را به گونه‌ای تنظیم کنید که در هنگام خروج از برنامه، پنجره کوچکی به همراه دو دکمه Yes و No مبنی بر تأیید برای خروج از نرم‌افزار نمایش داده شود. در صورتی که کاربر روی دکمه Yes کلیک کند، برنامه خاتمه یابد و در صورتی که روی دکمه No کلیک کند، به برنامه بازگردد.

در این جا لازم است با انواع عملگرهای مقایسه‌ای آشنا شوید. برای انجام هرگونه مقایسه می‌توانید از این عملگرها مطابق جدول ۸-۵ استفاده کنید.

جدول ۸-۵ عملگرهای مقایسه‌ای ویژوال بیسیک

| عملگر | مفهوم عملگر | مثال | نتیجه مقایسه |
|-------|----------------------|------------------|--------------|
| = | تساوی | "Ali"="BASIC" | نادرست |
| <= | کوچک‌تر یا مساوی | "BASIC"<="basic" | درست |
| >= | بزرگ‌تر یا مساوی | "BASIC">="Basic" | نادرست |
| > | کوچک‌تر | ۱۰ > ۵ | نادرست |
| < | بزرگ‌تر | ۱۴/۵ > ۲/۷۵ | درست |
| <> | عدم تساوی یا نامساوی | -۲ <> -۲ | نادرست |



نکته در مقایسه کاراکترها و عبارات رشته‌ای، از کد اسکی کاراکترها برای انجام مقایسه استفاده می‌شود.

عملگرهای مقایسه‌ای را می‌توانید برای مقایسه هر نوع داده‌ای استفاده کنید، با توجه به این‌که مقادیری که مقایسه می‌شوند از یک نوع باشند.

گاهی اوقات ممکن است از چند عملگر مقایسه‌ای در یک عبارت استفاده شود. در این صورت این سؤال مطرح می‌شود که کدام یک از عملگرها زودتر پردازش می‌شوند. در واقع در یک عبارت مقایسه‌ای با چند عملگر، عملیات با توجه به ترتیب قرار گرفتن عملگرها از چپ به راست عبارت انجام می‌شود و اگر از پرانتز استفاده شود، اولویت و حق تقدم عملیات با عملگرهای موجود در داخل پرانتز خواهد بود. به عنوان مثال این عبارت را در نظر بگیرید:

در عبارت مذکور ابتدا مقایسه $2 > 10$ بررسی می‌شود که نتیجه نادرست یا False را دربر خواهد داشت، سپس مقایسه $0 = 0$ بررسی می‌شود که نتیجه آن درست یا True خواهد بود.

تمرین:



- ۱ - پروژه showpicture را به شکلی تنظیم کنید تا در صورتی که کلمه رمز وارد شده صحیح باشد، فرم frmlogon از روی دسک‌تاپ برداشته شده و سپس فرم frmshow مشاهده شود، هم‌چنین در صورت نادرست بودن کلمه رمز رنگ کاراکترهای تایپ شده قرمز شود و کاربر توانایی ورود کلمه رمز دیگری را نداشته باشد. در ضمن در صورت ورود کلمه رمز به صورت اشتباه دکمه OK غیرفعال شود و رنگ زمینه کادر متن نیز به رنگ آبی درآید.
- ۲ - پروژه‌ای طراحی کنید که کاربر با تایپ اسامی کشورها بتواند سایر مشخصات مانند وسعت، جمعیت، پرچم، مذهب و واحد پول آن‌ها را نمایش دهد و هم‌چنین بتواند برای ورود به نرم‌افزار یک کلمه کاربر و یک کلمه رمز را وارد کند.

۳-۵ نحوه استفاده از عملگرهای منطقی برای ترکیب شرطها

گاهی اوقات ممکن است لازم باشد در دستورات شرطی بیش از یک شرط مورد بررسی قرار گیرد و براساس نتیجه حاصل از بررسی تمام شرطها اجرای برنامه صورت گیرد در این حالت می توان از عملگرهای منطقی مانند AND، OR یا NOT استفاده کرد. این عملگرها امکان ترکیب دو یا چند شرط را با یکدیگر فراهم می کنند.



مثال ۴: یکی دیگر از موارد قابل توجه در مثال قبل، این است که ویژوال بیسیک بین حروف کوچک و بزرگ تفاوت قائل می شود، به عنوان مثال عبارت BASIC را با عبارت basic مساوی در نظر نمی گیرد. البته اگر بخواهید می توانید این حالت را تغییر دهید و به ویژوال بیسیک بگویید که بین حروف کوچک و بزرگ تفاوت قائل نشود. به این منظور باید دستور Option Compare Text را در بخش تعاریف (Declarations Section) در پنجره کد فرم frmlogon بنویسید (شکل ۹-۵).

```
Option Compare Text
Private Sub cmdok_Click()
    If txtpass.Text = "BASIC" Then
        Valued frmlogon
        frmshow.Show
    Else
        txtpass.Locked = True
    End If
End Sub

Private Sub Form_Load()
    txtpass.Text = ""
    cmdok.Enabled = False
End Sub
```

شکل ۹-۵

برنامه را اجرا کنید و کلمه رمز را به صورت basic تایپ کنید و روی دکمه OK کلیک کنید. همان طور که مشاهده خواهید کرد این بار کلمه رمز پذیرفته می شود. روی دکمه Exit کلیک کنید تا برنامه خاتمه یابد، سپس تغییرات را ذخیره نمایید. البته این کار را می توانید به روش دیگری نیز انجام دهید که در این جا به ذکر آن می پردازیم. با استفاده از

عملگرهای منطقی در ویژوال بیسیک می‌توانید ترکیب‌های مختلفی از چند شرط را ایجاد کنید، به عنوان مثال در پروژه قبل برای آن‌که برنامه هر دو نوع عبارت BASIC یا basic را به عنوان کلمه رمز بپذیرد، می‌توانید خط اول از دستور If موجود در رویداد cmdok_Click را به این صورت تنظیم کنید:

```
If txtpass.Text = "BASIC" Or txtpass.Text = "basic" Then
```

سپس دستور Option Compare Text را از بخش تعاریف حذف کنید و پروژه را با دو حالت مختلف کلمه رمز آزمایش کنید و نتیجه را بررسی نمایید. همان‌طور که ملاحظه کردید در هر دو بار کلمه رمز پذیرفته می‌شود.

عملگر Or ("یا" منطقی) یکی از عملگرهای منطقی در ویژوال بیسیک است. از عملگرهای منطقی برای ایجاد ترکیب‌های مورد نظر از چند شرط مختلف، استفاده می‌شود. در صورتی که دو یا چند شرط با استفاده از عملگر منطقی Or با هم ترکیب شوند، نتیجه بررسی تمام ترکیب شرطی، زمانی درست است که حداقل یکی از شرط‌ها درست باشد یا به عبارت دیگر زمانی ترکیب شرطی نتیجه نادرست خواهد داشت که تمام شرط‌ها نادرست باشند. نتیجه ترکیب شرطی که با عملگر Or به وجود می‌آید در

جدول ۵-۹

| A | B | A Or B |
|--------|--------|--------|
| نادرست | نادرست | نادرست |
| درست | نادرست | درست |
| نادرست | درست | درست |
| درست | درست | درست |

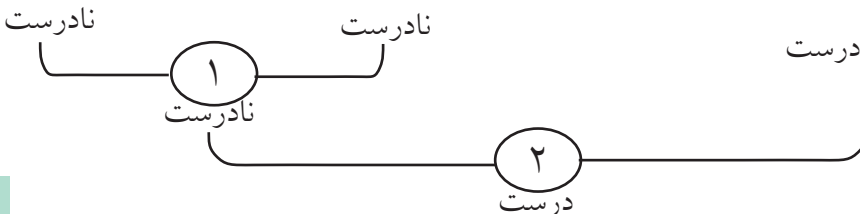
حالت‌های مختلف در جدول ۵-۹ ارایه شده است. منظور از A و B در این جدول عبارت‌های مقایسه‌ای است که با عملگر Or ترکیب شرطی را به وجود آورده و می‌تواند ارزش درست و یا نادرست داشته باشند.

به عنوان مثال این عبارت را در نظر بگیرید:

```
"ALI" = "ali" Or "d" < "B" Or "Tehran" > = "TABRIZ"
```

در عبارت فوق از دو عبارت مقایسه‌ای استفاده شده است که با عملگر منطقی Or با یکدیگر ترکیب شده‌اند، پس برای به دست آوردن نتیجه این ترکیب به صورت زیر عمل می‌کنید:

```
"ALI" = "ali" Or "d" < "B" Or "Tehran" > = "TABRIZ"
```



علاوه بر عملگر منطقی Or دو عملگر منطقی دیگر که معمولاً مورد استفاده برنامه‌نویسان قرار می‌گیرند عبارتند از: عملگر منطقی And ("و" منطقی) و عملگر منطقی Not (نقیض).

نتیجه عبارت شرطی که با And ایجاد می‌شود در جدول ۵-۱۰ ارایه شده است. همان‌طور که ملاحظه می‌کنید فقط زمانی ترکیب شرطی نتیجه درست خواهد داشت که نتیجه هر دو مقایسه و شرط به طور هم‌زمان درست باشد، در غیر این صورت ترکیب شرطی، نتیجه نادرست خواهد داشت.

نتیجه ترکیب شرطی که عملگر Not روی آن انجام می‌شود، مطابق جدول ۵-۱۱ است. همان‌طور که مشاهده می‌کنید عملگر Not می‌تواند ارزش یک شرط یا ترکیب شرطی را معکوس کند.

جدول ۵-۱۰

| A | B | A And B |
|--------|--------|---------|
| درست | درست | درست |
| نادرست | درست | نادرست |
| درست | نادرست | نادرست |
| نادرست | نادرست | نادرست |

جدول ۵-۱۱

| A | Not A |
|--------|--------|
| درست | نادرست |
| نادرست | درست |

اکنون ارزش درستی این عبارت را به‌دست آورید:

Not ("ali" <= "ALI") Or ("Reza" > "R") And ("Computer" <> "computer")

قبل از این‌که ارزش درستی عبارت قبل را به‌دست آورید، ذکر این نکته ضروری است که عملگرهای منطقی نیز مانند عملگرهای مقایسه‌ای، نسبت به هم اولویت اجرایی دارند و اگر از عملگرهای مشابه در یک عبارت استفاده شود، حق تقدم و اولویت اجرای آن‌ها از چپ به راست خواهد بود، اما اگر از عملگرهای متفاوت استفاده شود، حق تقدم اجرای آن‌ها با توجه به جدول ۵-۱۲ و از بالا به پایین خواهد بود.

جدول ۵-۱۲

| |
|-----|
| Not |
| And |
| Or |

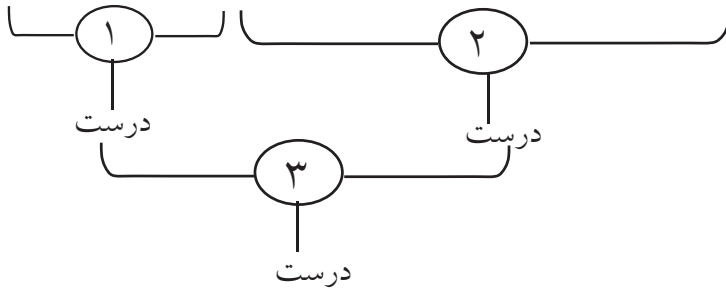
اولویت اجرای عملگرهای مقایسه‌ای از عملگرهای منطقی بالاتر است.



با توجه به مطالب گفته شده نحوه ارزیابی عبارت مورد نظر به این صورت خواهد بود:

Not ("ali" <= "ALI") Or ("Reza" > "R") And ("Computer" <> "computer")

Not(نادرست) Or (درست) And (درست)



تمرین:

در صورتی که مقدار $A=2$ ، $B=-14$ و $C=0$ باشد، ارزش درستی این عبارات را به دست

آورید:

$A < C$ Or Not $B <> C$ AND "A" = "Q" Or $C <= 2$

"6" > "9" AND $A > B$ Or Not $C > B$

۴-۵ کنترل تصویر

ویژوال بیسیک کنترل دیگری به نام تصویر Image دارد که مانند کادر تصویر برای نمایش تصاویر به کار می‌رود. خصوصیات این کنترل مشابه کنترل کادر تصویر است، اما کنترل تصویر (Image) یک خصوصیت به نام Stretch دارد که در کنترل کادر تصویر وجود ندارد. این خصوصیت از نوع منطقی بوده و در صورتی که روی مقدار False تنظیم شود، اندازه کنترل با اندازه تصویر هماهنگ می‌شود. اما اگر این خصوصیت روی مقدار True تنظیم شود اندازه تصویر با توجه به اندازه کنترل تنظیم می‌شود. می‌توانید از این خصوصیت برای بزرگ و کوچک کردن تصاویر استفاده کنید.



مثال ۵: پروژه‌ای طراحی کنید که کاربر بتواند هر فایل گرافیکی مورد نظر خود را

در یک پنجره مطابق شکل ۱۰-۵ نمایش دهد.



شکل ۱۰-۵

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE ایجاد کنید، سپس کنترل‌های مربوطه را مطابق جدول ۱۳-۵ ایجاد کنید و فرم و پروژه را با نام Image ذخیره نمایید.

جدول ۱۳-۵

| کنترل / خصوصیت | Form | TextBox | Label | Command Button | Image | Command Button |
|----------------|----------|-------------|---------------------|----------------|---------|----------------|
| Name | frmimage | txtfilename | lblfilename | cmdok | imgshow | cmdexit |
| Caption | Image | — | Path and File Name: | &OK | — | Exit |
| Stretch | — | — | — | — | True | — |
| AutoSize | — | — | True | — | — | — |

۲- به پنجره کد فرم بروید و رویداد Load فرم و Change کادر متن را به صورت زیر تنظیم کنید. این دستورات باعث می‌شوند تا زمانی که کاربر، نام و مسیر فایل را در کادر متن تایپ نکند، دکمه OK فعال و قابل استفاده نباشد.

```
Private Sub Form_Load()  
    cmdok.Enabled = False  
End Sub  
Private Sub txtfilename_change()  
    cmdok.Enabled = True  
End Sub
```

۳ - این دستور را در رویداد Click دکمه Exit تایپ کنید:

```
Unload frmimage
```


ذکر این نکته ضروری است که برای خروج اطلاعات فرم فعال از حافظه و پایان دادن به برنامه می‌توانید از دستور Unload Me استفاده کنید، کلمه کلیدی Me به نام فرم فعال اشاره می‌کند.

۴ - برای آن‌که پس از تایپ نام، مسیر فایل و کلیک روی دکمه OK، تصویر مورد نظر نمایش داده شود، رویداد Click دکمه فرمان OK را به این صورت تنظیم کنید:

```
Private Sub cmdok_Click()  
    imgshow.Picture= LoadPicture (txtfilename.Text)  
End Sub
```

با کلیک روی دکمه OK این رویداد اجرا شده و تصویر مورد نظر به وسیله تابع LoadPicture نمایش داده می‌شود. تابع LoadPicture می‌تواند نام و مسیر یک فایل گرافیکی را دریافت کند و آنرا در یک کنترل کادر تصویر (PictureBox) یا تصویر (Image) نمایش دهد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

(نام و مسیر فایل گرافیکی) "Picture = LoadPicture نام کنترل تصویر یا کادر تصویر"

 **نکته** باید نام و مسیر فایل را داخل کاراکتر " قرار دهید. اگر از تابع فوق به صورت ("") Load Picture استفاده کنید تصویر نمایش داده شده در کنترل کادر تصویر یا کنترل تصویر، برداشته شده و حذف می‌شود.

۵ - پروژه را اجرا کرده و نام و مسیر یک فایل گرافیکی را در کادر متن بنویسید و روی دکمه OK کلیک کنید.

۶ - اجرای پروژه را خاتمه دهید و به پنجره ویژوال بیسیک بازگردید.

۵-۵ اولویت اجرای عملگرها نسبت به یکدیگر

تاکنون اولویت اجرای عملگرها در گروه خود را فرا گرفتید. عملگرهای موجود در گروه‌های مختلف نیز در صورتی که در یک عبارت قرار بگیرند، دارای اولویت اجرایی نسبت به هم هستند، اولویت عملگرها در تمام گروه‌ها در جدول ۵-۱۴-۱۴ ارایه شده‌اند. بالاترین اولویت مربوط به عملگرهای ریاضی و پایین‌ترین اولویت مربوط به عملگرهای منطقی می‌شود.

جدول ۵-۱۴

| |
|--|
| عملگرهای ریاضی براساس اولویت‌های ذکر شده |
| عملگر اتصال رشته‌ها (&) |
| عملگرهای مقایسه‌ای براساس اولویت‌های ذکر شده |
| عملگرهای منطقی براساس اولویت‌های ذکر شده |

۵-۶ نحوه استفاده از کادرهای پیام در ویژوال بیسیک

احتمالاً تاکنون با کادرهای پیام در ویندوز برخورد کرده‌اید، کادرهای پیام معمولاً برای تأیید یا لغو عملیات یا مطلع کردن کاربران از یک اتفاق یا خطا استفاده می‌شوند و معمولاً پیامی را به همراه یک یا چند دکمه به کاربر نمایش می‌دهند تا با توجه به نیاز خود تصمیم‌گیری مناسب را انجام دهد. به عنوان مثال می‌توان به کادرهای پیامی که در شکل‌های ۵-۱۱ و ۵-۱۲-۱۲ نمایش داده شده‌اند، اشاره کرد.



شکل ۵-۱۲



شکل ۵-۱۱

ویژوال بیسیک توانایی ایجاد انواع مختلفی از کادرهای پیام را برای شما فراهم می‌کند. برای نمایش کادرهای پیام در ویژوال بیسیک از تابع MsgBox استفاده می‌شود. نحوه استفاده از این تابع به یکی از این روش‌ها امکان‌پذیر است:

(عنوان, نوع و تعداد دکمه‌ها و آیکن کادر پیام, پیام‌نمایشی) MsgBox = نام متغیر از نوع صحیح

عنوان, نوع و تعداد دکمه‌ها و آیکن کادر پیغام, پیام‌نمایشی MsgBox

پیام‌نمایشی و عنوان می‌توانند یک عبارت یا متغیر رشته‌ای باشند که به ترتیب، پیام‌نمایشی داخل کادر پیام و عنوان کادر پیام را تعیین می‌کنند. بخش دوم در تابع MsgBox می‌تواند نوع و تعداد دکمه‌ها، نوع آیکن کادر پیام و دکمه پیش فرض را در کادر پیام تعیین کند. برای این کار می‌توانید مقادیر عددی یا رشته‌ای ارائه شده در جداول ۱۵-۵ تا ۱۷-۵ را با استفاده از کاراکتر + با هم ترکیب کنید.

جدول ۱۵-۵ مقادیر مربوط به نوع و تعداد دکمه‌ها در کادر پیام

| ثابت رشته‌ای | ثابت عددی | توضیح |
|--------------------|-----------|--|
| vbOKOnly | ۰ | کادر پیام با دکمه OK |
| vbOKCancel | ۱ | کادر پیام با دکمه OK و Cancel |
| vbAbortRetryIgnore | ۲ | کادر پیام با دکمه Abort، Retry، Ignore |
| vbYesNoCancel | ۳ | کادر پیام با دکمه Yes، No، Cancel |
| vbYesNo | ۴ | کادر پیام با دکمه Yes و No |
| vbRetryCancel | ۵ | کادر پیام با دکمه Retry و Cancel |

جدول ۱۶-۵ مقادیر مربوط به نوع آیکن در کادر پیام

| ثابت رشته‌ای | ثابت عددی | توضیح |
|---------------|-----------|---|
| vbCritical | ۱۶ | آیکن پیام خطای بحرانی  |
| vbQuestion | ۳۲ | آیکن پرسش  |
| vbExclamation | ۴۸ | آیکن پیام هشدار  |
| vbInformation | ۶۴ | آیکن پیام اطلاعات  |

جدول ۱۷-۵ مقادیر مربوط به دکمه پیش فرض

| ثابت رشته‌ای | ثابت عددی | توضیح |
|------------------|-----------|-------------------------------|
| vbDefaultButton1 | صفر | اولین دکمه، دکمه پیش فرض است. |
| vbDefaultButton2 | ۲۵۶ | دومین دکمه، دکمه پیش فرض است. |
| vbDefaultButton3 | ۵۱۲ | سومین دکمه، دکمه پیش فرض است. |

نکته دکمه پیش فرض دکمه‌ای است که در زمان نمایش کادر پیام، فوکوس را در اختیار می‌گیرد.



وقتی که کاربر روی دکمه‌ای در کادر پیام کلیک کند، یکی از مقادیر ارائه شده در جدول ۱۸-۵ بازگشت داده می‌شود. با ذخیره‌سازی این مقدار در یک متغیر از نوع Integer می‌توان با توجه به پاسخ کاربر اجرای دستورالعمل‌ها را کنترل و مدیریت کرد.

جدول ۱۸-۵ مقادیر مربوط به دکمه‌ای که کلیک می‌شود.

| ثابت رشته‌ای | ثابت عددی | توضیح |
|--------------|-----------|-------------------------------|
| vbOK | ۱ | اگر روی دکمه OK کلیک شود. |
| vbCancel | ۲ | اگر روی دکمه Cancel کلیک شود. |
| vbAbort | ۳ | اگر روی دکمه Abort کلیک شود. |
| vbRetry | ۴ | اگر روی دکمه Retry کلیک شود. |
| vbIgnore | ۵ | اگر روی دکمه Ignore کلیک شود. |
| vbYes | ۶ | اگر روی دکمه Yes کلیک شود. |
| vbNo | ۷ | اگر روی دکمه No کلیک شود. |

به عنوان مثال به دستور زیر توجه کنید:

Intanswer=MsgBox("Data Entry Is Invalid!" ,

vbRetryCancel+vbCritical+vbDefaultButton2 , "Error")



شکل ۱۳-۵

در صورت اجرای این دستور کادر پیامی مطابق شکل ۱۳-۵ نمایش داده می‌شود. همان‌طور که ملاحظه می‌کنید در بخش دوم با استفاده از کاراکتر + مقادیر مربوط به نوع آیکن، تعداد و نوع دکمه‌ها و دکمه پیش فرض تعیین شده‌اند.

اگر کاربر روی دکمه Retry کلیک کند مقدار VbRetry (معادل عدد ۴) در متغیر Intanswer ذخیره می‌شود و اگر روی دکمه Cancel کلیک کند مقدار VbCancel (معادل عدد ۲) در متغیر Intanswer ذخیره می‌شود.

نکته اگر نوع و تعداد دکمه‌ها تعیین نشده باشند کادر پیام با دکمه OK نمایش داده می‌شود. در صورت عدم انتخاب آیکن، هیچ آیکنی نمایش داده نخواهد شد. به علاوه اگر دکمه پیش فرض انتخاب نشده باشد، اولین دکمه در کادر پیام به عنوان دکمه پیش فرض انتخاب خواهد شد.



به عنوان مثال این دستور یک کادر محاوره مطابق شکل ۱۴-۵ ایجاد می‌کند.

Intanswer=MsgBox("Password Is Not Correct!?", vbOKCancel)



شکل ۱۴-۵

مثال ۶: می‌خواهیم پروژه Image را به شکلی تنظیم کنیم تا هنگام خروج از برنامه پیام تأییدی به کاربر نمایش داده شود.



- ۱ - پروژه Image را باز کنید و به پنجره طراحی فرم بروید.
- ۲ - روی دکمه فرمان Exit دوبار کلیک کنید تا به رویداد Click دکمه فرمان Exit بروید.
- ۳ - دستورات رویداد Click دکمه فرمان Exit را به این صورت تغییر دهید.

```
Private Sub cmdexit_Click()
```

```
Dim intanswer As Integerintanswer=MsgBox("Do You Want To Exit?",
```

```
VbYesNo + vbQuestion,"Exit")
```

```
If intanswer =vbYes Then UnLoad me
```

```
End Sub
```

در این رویداد با استفاده از تابع MsgBox کادر پیامی طراحی شده است که شامل عبارت Do You Want To Exit? و عنوان Exit به همراه دو دکمه Yes و No می‌باشد. در صورت اجرای برنامه و کلیک روی دکمه Exit این رویداد اجرا می‌شود و با نمایش کادر پیام سبب

توقف اجرای برنامه می‌شود و تا زمانی که کاربر پاسخی به کادر پیام ندهد اجرای برنامه متوقف خواهد ماند؛ اما در صورتی که کاربر روی یکی از دکمه‌های کادر پیام کلیک کند تابع MsgBox مقداری مناسب با دکمه مربوطه در متغیر intanswer ذخیره می‌کند. اگر کاربر روی دکمه Yes کلیک کند، با بسته شدن فرم مقدار vbYes در متغیر intanswer ذخیره می‌شود و با استفاده از یک دستور If و بررسی مقدار intanswer اجرای برنامه خاتمه می‌یابد، اما در صورتی که کاربر روی دکمه No یا دکمه Close کلیک کند، مقدار vbNo در متغیر intanswer ذخیره می‌شود و برنامه بدون هیچ گونه تغییری در مقدار زمان ادامه می‌یابد.

۴ - تغییرات را ذخیره کرده و پروژه را اجرا نمایید. روی دکمه Exit کلیک کنید همان‌طور که ملاحظه می‌کنید کادر پیامی مطابق شکل ۱۵-۵ نمایش داده می‌شود.



شکل ۱۵-۵

۵ - روی دکمه No کلیک کنید تا به برنامه بازگردید.

۶ - روی دکمه Exit کلیک کنید. این بار روی

دکمه Yes در کادر پیام کلیک کنید و تفاوت را با حالت قبل بررسی کنید.

۷ - اجرای برنامه را متوقف کنید و به محیط

ویژوال بیسیک بازگردید.

۷-۵ نحوه استفاده از کادرهای ورود داده در ویژوال بیسیک

تاکنون برای ورود داده‌های خود به برنامه از کنترل کادر متن استفاده کرده‌اید. در ویژوال بیسیک روش راحت‌تری برای انجام این کار وجود دارد و آن استفاده از کادر ورود داده یا InputBox است. کادر ورود داده در واقع شبیه به یک کادر پیام است، با این تفاوت که دارای یک کادر متن برای ورود داده و دو دکمه OK و Cancel است (شکل ۱۶-۵). برای ایجاد کادرهای ورود داده در ویژوال بیسیک از تابع InputBox استفاده می‌شود.



شکل ۱۶-۵

این تابع می‌تواند یک کادر ورود داده ایجاد کند و در صورتی که کاربر روی OK کلیک کند داده‌های وارد شده در کادر متن را در یک متغیر رشته‌ای ذخیره می‌کند. اما اگر کاربر روی دکمه Cancel کلیک کند، یک رشته خالی ("") را در متغیر رشته‌ای ذخیره می‌کند. شکل کلی نحوه استفاده از این تابع به یکی از روش‌های زیر است:

(x,y, مقدارپیش‌فرض در کادرمتن, عنوان کادر ورود داده
, پیام نمایشی در کادر ورود داده) = InputBox = نام متغیر رشته‌ای

x, y, عنوان کادر ورود داده, پیام نمایشی در کادر ورود داده InputBox

مقادیر مربوط به پیام نمایشی و عنوان کادر ورود داده از نوع رشته‌ای است و مقدار پیش‌فرض مقداری است که در زمان نمایش کادر ورود داده در کادر متن نمایش داده می‌شود. x و y مقادیر عددی هستند که به ترتیب فاصله کادر ورود داده را از سمت چپ و بالای دسک‌تاپ تعیین می‌کنند.

نکته در صورتی که عنوان کادر ورود داده و کادر پیام تعیین نشود نام فایل پروژه در نوار عنوان آن‌ها نمایش داده می‌شود

مثال ۷: مطابق شکل ۱۷-۵ برنامه‌ای طراحی کنید که کاربر توانایی محاسبه محیط و مساحت هر دایره به شعاع دلخواه را داشته باشد و برای دریافت داده‌ها از کادر ورود داده که به وسیله دکمه Enter Radius فعال می‌شود، استفاده کند.
۱ - یک پروژه با یک فرم، یک کنترل دکمه فرمان و کنترل‌های برچسب مطابق شکل ۱۷-۵ ایجاد کنید.



شکل ۱۷-۵

۲ - رویداد Click کنترل دکمه فرمان را به این صورت تنظیم کنید:

```
Private Sub cmdr_Click()
```

```
Dim str As String
```

```
Dim sngarea As Single, sngper As Single
```

```
str = InputBox("Enter Radius:", "InputData:", 0)
```

```
sngper = 2 * 3.14 * Val(str)
```

```
sngarea = 3.14 * Val(str)^2
```

```
lblper.Caption = sngper
```

```
lblarea.Caption = sngarea
```

```
End Sub
```

۳ - پروژه و فرم را با نام Circle و Circles ذخیره کنید، سپس پروژه را اجرا کنید.

۴ - روی دکمه Enter Radius کلیک کنید تا کادر ورود داده نمایش داده شود.

۵ - عدد ۳ را در کادر متن، کادر ورود داده تایپ کنید و روی دکمه OK در کادر

ورود داده کلیک کنید. در این صورت مقدار تایپ شده در کادر متن در متغیر رشته‌ای str ذخیره می‌شود، سپس با استفاده از تابع Val مقدار متغیر str به عدد تبدیل می‌شود و با استفاده از فرمول‌های مربوطه محیط و مساحت دایره محاسبه می‌شود.

۶ - مجدداً روی دکمه Enter Radius کلیک کنید و عدد ۴ را در کادر متن، کادر ورود

داده تایپ کنید و سپس روی دکمه Cancel کلیک کنید. در این صورت یک رشته خالی در متغیر str ذخیره می‌شود و در تبدیل متغیر str با تابع Val مقدار صفر تولید می‌شود که نتایج محاسبات نیز خواهد بود.

۷ - به اجرای پروژه خاتمه دهید و به پنجره ویژوال بیسیک بازگردید.

۸-۵ کنترل کادر علامت CheckBox

این کنترل امکان ایجاد انتخاب‌های دو حالتی را به وجود می‌آورد به شکلی که در صورت انتخاب کنترل (قرار دادن علامت چک مارک در داخل مربع کنترل) تصمیمات خاصی در برنامه اتخاذ شود و براساس آن پردازش و محاسبات انجام شود یا با برداشتن علامت

چک‌مارک از داخل مربع کنترل کادر علامت از تصمیمات مورد نظر صرف نظر شود. بدین ترتیب کاربر می‌تواند با استفاده از این ویژگی‌ها شرایط خود را به برنامه اطلاع دهد.

جدول ۱۹-۵

| خصوصیت | توضیح |
|-----------|---|
| Alignment | در صورتی که مقدار این خصوصیت روی 0-Left Justify تنظیم شود، کادر علامت در سمت چپ عنوان کنترل و در صورتی که روی 1-Right Justify تنظیم شود کادر علامت در سمت راست عنوان آن نمایش داده می‌شود. |
| Style | در صورتی که مقدار این خصوصیت روی 0-Standard تنظیم شود شکل کنترل به صورت یک کادر علامت معمولی و وقتی روی مقدار 1-Graphical تنظیم شود به شکل یک دکمه فرمان دیده می‌شود. |
| Value | در صورتی که مقدار این خصوصیت برابر 0-Unchecked باشد نشان دهنده این است که کنترل انتخاب نشده است و اگر برابر 1-Checked باشد نشان دهنده این است که کنترل انتخاب شده است و اگر مقدار آن برابر 2-Grayed باشد به معنی این است که کنترل دارای علامت چک مارک بوده و زمینه مربع کادر علامت نیز خاکستری رنگ است. |

۹-۵ معرفی مهم‌ترین ویژگی‌های فرم‌ها و کنترل‌های کادر متن، دکمه فرمان و غیره

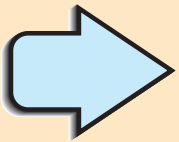
در این واحد کار با اشیایی نظیر فرم و کنترل‌های متعدد آشنا شدید، در پایان لازم است تا شما را با بعضی از خصوصیات و رویدادهای دیگر فرم و کنترل ذکر شده بیشتر آشنا کنیم.

جدول ۲۰-۵ خصوصیات فرم‌ها و کنترل‌ها

| خصوصیت | توضیح |
|--------------|---|
| Appearance | اگر مقدار این خصوصیت روی Flat- تنظیم شود، کنترل یا فرم به صورت مسطح و اگر روی مقدار ۱-۳D تنظیم شود، کنترل یا فرم به صورت سه بعدی نمایش داده می‌شود. |
| BackColor | رنگ زمینه را در فرم و بعضی از کنترل‌ها مانند کادر متن تعیین می‌کند. |
| Enabled | اگر مقدار این خصوصیت روی True تنظیم شود، کنترل یا فرم فعال و قابل استفاده بوده و اگر روی False تنظیم شود، کنترل و فرم غیرفعال و غیر قابل استفاده می‌شود. |
| ForeColor | رنگ قلم را در فرم و بعضی از کنترل‌ها تعیین می‌کند. |
| Font | تنظیمات قلم را در فرم و بعضی از کنترل‌ها تعیین می‌کند. |
| Picture | یک تصویر را روی فرم و بعضی از کنترل‌ها تعیین می‌کند. |
| MousePointer | اشاره‌گر ماوس را برای فرم و بعضی از کنترل‌ها تعیین می‌کند. |
| MouseIcon | یک آیکن دلخواه را به عنوان اشاره‌گر برای فرم و بعضی کنترل‌ها تعیین می‌کند. |
| Visible | اگر مقدار این خصوصیت روی True تنظیم شود کنترل یا فرم نمایش داده شده و اگر مقدار این خصوصیت روی False تنظیم شود کنترل یا فرم نمایش داده نخواهد شد. |
| TabIndex | این خصوصیت مربوط به کنترل‌ها می‌شود و ترتیب حرکت روی کنترل‌ها را با استفاده از کلید Tab تعیین می‌کند و مقادیر مساوی یا بزرگ‌تر از صفر را کسب می‌کند. |
| TabStop | این خصوصیت از نوع منطقی بوده و مربوط به کنترل‌ها می‌شود و اگر روی مقدار False تنظیم شود در هنگام استفاده از کلید Tab، کنترل مربوط، فوکوس را دریافت نمی‌کند. |
| BorderStyle | اگر این خصوصیت در کنترل‌های برجسته و کادر متن روی مقدار None- ۰ تنظیم شود، کنترل بدون حاشیه و کادر دور و اگر روی مقدار 1-Fixed Single تنظیم شود، کنترل با حاشیه و کادر دور مشاهده می‌شود. |
| MaxButton | اگر مقدار این خصوصیت روی True تنظیم شود، دکمه Maximize در فرم فعال و در صورتی که مقدار آن روی False تنظیم شود، دکمه Maximize غیرفعال می‌شود. |
| MinButton | اگر مقدار این خصوصیت روی True تنظیم شود، دکمه Minimize در فرم فعال و در صورتی که مقدار آن روی False تنظیم شود دکمه Minimize غیرفعال می‌شود. |
| Icon | یک آیکن دلخواه را به عنوان آیکن منوی کنترل در فرم تعیین می‌کند. |

ادامه جدول ۲۰-۵

| خصوصیت | توضیح |
|-------------|---|
| Alignment | این خصوصیت در کنترل‌هایی مانند برجسب و کادر متن وجود دارد و تراز عبارت را در کنترل تعیین می‌کند، مقدار 0-Left Justify تراز از چپ، 1-Right Justify تراز از راست و 2-Center تراز از وسط را به وجود می‌آورند. |
| ToolTipText | این خصوصیت در بیشتر کنترل‌ها وجود دارد و زمانی که اشاره‌گر ماوس روی کنترل نگه داشته شود عبارت موجود در این خصوصیت نمایش داده می‌شود. از این خصوصیت برای راهنمایی کاربران استفاده می‌شود. |
| WordWrap | این خصوصیت مربوط به کنترل برجسب است. اگر محتویات خصوصیت Caption بیش از اندازه کنترل باشد و بخواهید عبارت موجود در خصوصیت Caption در چند خط نمایش داده شود، مقدار این خصوصیت و سپس مقدار خصوصیت AutoSize را روی True تنظیم کنید. |
| MultiLine | این خصوصیت مربوط به کنترل کادر متن است و در صورتی که مقدار آن روی True تنظیم شود، می‌توانید اطلاعات خود را در خطوط جداگانه وارد کنید. البته بهتر است که ارتفاع کنترل را نیز طوری تنظیم نمایید تا بتوان چند خط را به طور هم‌زمان مشاهده کرد. |
| Moveable | اگر مقدار این خصوصیت روی True تنظیم شود امکان جابه‌جا کردن فرم روی دسک‌تاپ وجود دارد، اما اگر مقدار آن روی False تنظیم شود این امکان وجود ندارد. |
| WindowState | این خصوصیت مربوط به فرم‌ها بوده و سه مقدار مختلف را کسب می‌کند. در صورتی که مقدار آن روی 0-Normal تنظیم شود موقعیت فرم روی دسک‌تاپ به وسیله خصوصیات Top و Left تنظیم می‌شود و اگر مقدار آن روی 1-Minimized تنظیم شود پنجره برنامه در زمان اجرا با اندازه حداقل و در صورتی که مقدار آن روی 2-Maximized تنظیم شود، پنجره برنامه با اندازه حداکثر نمایش داده خواهد شد. |



Learn in English

If...Then...Else

Use an If...Then...Else block to define several blocks of statements, one of which will execute

If *condition1* Then

[*statementblock*]

[Else

[*statementblock*]]

End If

Visual Basic first tests *condition1*. If it's False, Visual Basic executes the corresponding statement Else statement block

For example, your application could perform different actions with depending on the value of grade variable:

```
If grade < 10 Then
```

```
    result ="failed"
```

```
Else
```

```
    result ="passed"
```

```
End If
```


واژه‌نامه

| | |
|---------------|----------------|
| Border | کادر، حاشیه |
| Compare | مقایسه کردن |
| Condition | شرط |
| CorresPonding | متناظر، متناسب |
| Decision | تصمیم |
| Execute | اجرا کردن |
| Image | تصویر |
| Input Box | کادر ورود داده |
| Logical | منطقی |
| Message Box | کادر پیام |
| Operation | عملگر |
| Perform | انجام دادن |
| Structure | ساختار |
| Style | حالت |

خلاصه مطالب

• برای کنترل روند اجرای برنامه در شرایط مختلف از دستورات شرطی مانند If استفاده می‌شود.

شکل کلی دستور If به این صورت است:

```
If (شرط(ها)) Then
    .....
    دستورات
Else
    .....
    دستورات
End If
```

• برای اضافه کردن یک فرم جدید به پروژه می‌توانید از گزینه Add Form در منوی Project، کلیک راست در پنجره پروژه و انتخاب گزینه Add Form (یا از دکمه Add Form در نوار ابزار استاندارد) استفاده کنید.

- برای مقایسه مقادیر رشته‌ای، عددی و سایر انواع داده‌ها از عملگرهای مقایسه‌ای استفاده می‌شود.
- متد Show می‌تواند یک فرم را در حافظه بارگذاری کرده و نمایش دهد.
- متد Hide می‌تواند یک فرم را بدون آن که از حافظه خارج نماید، مخفی کند.
- برای ایجاد ترکیب چند شرط با یکدیگر از عملگرهای منطقی And، Or و Not استفاده می‌شود.
- با استفاده از دستور Unload می‌توان یک فرم را از حافظه خارج کرد.
- ترکیب شرطی که با استفاده از عملگر And ایجاد می‌شود، وقتی نتیجه درست (True) خواهد داشت که تمام مقایسه‌ها به طور هم‌زمان درست باشند و در سایر حالت‌ها نتیجه نادرست خواهد داشت.
- ترکیب شرطی که با استفاده از عملگر Or ایجاد می‌شود، وقتی نتیجه نادرست (False) خواهد داشت که تمام مقایسه‌ها نادرست باشند و در سایر حالت‌ها نتیجه درست خواهد داشت.
- از کنترل کادر تصویر و کنترل تصویر می‌توان برای نمایش فایل‌های گرافیکی یا تصویر استفاده کرد.
- عملگر Not نتیجه هر عبارت مقایسه‌ای را که درست باشد به نادرست و اگر نادرست باشد به درست تبدیل می‌کند.
- از کادرهای پیام، برای نمایش پیام‌های مناسب در مواقع نیاز و ایجاد حالت‌های مختلف انتخاب برای کاربر استفاده می‌شود.
- از کادرهای ورود داده برای دریافت داده‌ها از کاربر استفاده می‌شود.
- کادرهای پیام با استفاده از تابع MsgBox و کادرهای ورود داده با استفاده از تابع InputBox ایجاد می‌شوند.
- تابع RTrim کاراکترهای فاصله در انتهای یک عبارت متنی (رشته‌ای) را حذف می‌کند.
- تابع LTrim کاراکترهای فاصله در ابتدای یک عبارت متنی (رشته‌ای) را حذف می‌کند.
- تابع Trim کاراکترهای فاصله در ابتدا و انتهای یک عبارت متنی (رشته‌ای) را حذف می‌کند.
- با استفاده از تابع Load Picture می‌توان یک فایل گرافیکی یا تصویر را در کنترل کادر تصویر یا کنترل تصویر نمایش داد.

آزمون نظری

۱ - کدام خصوصیت در کادر متن حداکثر تعداد کاراکترهای ورودی را تعیین می‌کند؟

الف - BorderStyle ب - PasswordChar ج - MaxLength د - TabIndex

۲ - کدام عملگر منطقی از اولویت پایین‌تری نسبت به سایر عملگرهای منطقی

برخوردار است؟

الف - And ب - Or ج - Not د - And و Or

۳ - کدام تابع برای ایجاد یک کادر ورود داده مناسب است؟

الف - MsgBox ب - InputBox ج - InBox د - msg

۴ - در صورتی که نوع و تعداد دکمه‌ها در کادر پیام تعیین نشود، به‌طور پیش‌فرض از

چه دکمه‌هایی استفاده می‌شود؟

الف - OK ب - OK و Cancel ج - Yes و No د - Cancel

۵ - کدام گزینه در رابطه با خصوصیت TabIndex درست است؟

الف - ترتیب نمایش کنترل‌ها را روی فرم تعیین می‌کند.

ب - ترتیب حرکت بین کنترل‌ها را روی یک فرم معین می‌کند.

ج - ترتیب بارگذاری کنترل‌ها را در حافظه تعیین می‌کند.

د - وضعیت کلید Tab را در صفحه کلید بررسی می‌کند.

۶ - حاصل عبارت $3 < 5 * 3 > 12$ چیست؟

الف - True ب - False ج - ۲ د - ۲

۷ - کدام خصوصیت مشترک بین فرم و کنترل‌ها وجود دارد که توانایی مخفی کردن

یا نمایش آن‌ها را دارد؟

الف - Caption ب - Enabled ج - Visible د - Appearance

۸ - حاصل ترکیب شرطی زیر چیست؟

(با توجه به عدم استفاده از دستور Option Compare Text)

"Cpu" = "Cpu" Or 50 <= -70 And Not True

الف - True

ب - False

ج - استفاده از عملگر Not و And در کنار هم اشتباه است.

د - استفاده از عملگر مقادیر منطقی، عددی و رشته‌ای در یک ترکیب اشتباه است.

۹- در صورت عدم انتخاب عنوان یک کادر پیام به طور پیش فرض از استفاده می شود.
۱۰- الف- نام فرم ب- نام پروژه ج- نام فایل فرم د- نام فایل پروژه
با استفاده از کدام خصوصیت در کادر متن می توان از ویرایش داده ها توسط کاربر
جلوگیری کرد؟

الف- Wordwrap ب- Lock ج- Locked د- MultiLine

۱۱- In the If statement while condition is false, Visual Basic executes the statements that exist within the part.

a- then b- else c- if else d- end if

۱۲- عملگرهای مقایسه ای را بیان کنید سپس کاربرد هر یک را توضیح دهید.
۱۳- دستور شرطی If را توضیح دهید و حالت های مختلف نحوه استفاده از آن ها را بیان کنید.

۱۴- اصطلاحات زیر را توضیح دهید:

الف- متد Show

ب- رویداد Unload

ج- متد Hide

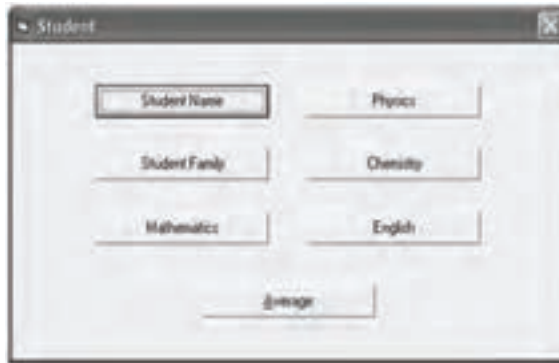
د- رویداد Change

۱۵- تفاوت بین کادر پیغام و کادر ورود داده را بیان کنید.
۱۶- عملگرهای منطقی را نام ببرید و کاربرد هر یک را بیان کنید.

آزمون عملی

- ۱ - حاصل عبارت‌های زیر به ازای مقادیر $a = 15$ ، $b = 20$ و $c = 100$ چه خواهد بود؟
- الف - $a > 40$ AND $b >= 10$ OR $c < 0$
- ب - $b <= 10$ OR $c <> 100$
- ج - $a <= 35$ OR $c > 100$ AND $b >= 20$

۲ - پروژه‌ای طراحی کنید که مطابق شکل ۱ نام، نام خانوادگی و نمره ۴ درس یک دانش‌آموز را دریافت کند، سپس معدل وی را محاسبه نماید. در صورتی که معدل وی کوچک‌تر از ۱۲ باشد یک کادر پیام مطابق شکل ۲ و در صورتی که معدل نمره وی بزرگ‌تر یا مساوی ۱۲ باشد کادر پیامی مطابق شکل ۳ نمایش داده شود، به علاوه برای دریافت داده از کادر ورود داده استفاده شود. لازم به ذکر است که کلیک روی دکمه‌های فرمان، کادر ورود داده‌ای را در اختیار کاربر قرار می‌دهد.



The image shows a window titled 'Student' with a close button in the top right corner. Inside the window, there are six input fields arranged in two columns. The left column contains 'Student Name', 'Student Family', and 'Mathematics'. The right column contains 'Physics', 'Chemistry', and 'English'. Below these fields is a single input field labeled 'Average'.

شکل ۱



شکل ۳



شکل ۲

توانایی استفاده از انواع حلقه‌ها و ساختار Case Select و کنترل دکمه انتخاب

هدف‌های رفتاری

- پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:
- ۱- بتواند از دستور Select Case استفاده کند.
 - ۲- بتواند از کنترل دکمه انتخاب استفاده کرده، رویدادها و ویژگی‌های آن را به کاربرد.
 - ۳- نحوه کار با حلقه‌های Do While...Loop.While...Wend.For...Next و غیره را توضیح دهد.
 - ۴- بتواند از دستورات خروج از حلقه Exit For و Exit Do استفاده کند.

کلیات

یکی از مسایلی که در پروژه‌های واقعی با آن برخورد خواهید کرد تکراری بودن بعضی از دستورات و عملیات است. این دستورات تکراری ممکن است به تعداد دفعات معین تا رسیدن به شرایط خاصی انجام شوند. ساختارهای تکرار یا به عبارت دیگر حلقه‌ها، در این زمینه به شما کمک می‌کنند تا این‌گونه عملیات را راحت‌تر و مناسب‌تر انجام دهید. به علاوه معمولاً هنگام طراحی برنامه‌ها لازم است تا روند اجرای دستورات را تعیین کنید. ساختارهای تصمیم، امکانات اجرایی دستورات را با توجه به شرایط مورد نظر فراهم می‌کنند. از این گروه نحوه استفاده از دستور If را فرا گرفته‌اید. نوع دیگری از ساختارهای تصمیم نیز به نام Select Case وجود دارند که در این واحدکار به توضیح آن می‌پردازیم.

۱-۶ دستور Select Case

این دستور مانند دستور If به برنامه‌نویس اجازه می‌دهد تا در شرایط مختلف دستورات مورد نظر خود را اجرا کند و عملیات مناسب را با داده‌ها و اطلاعات موجود انجام دهد. استفاده از دستور If زمانی که نیاز به بررسی شرط‌ها و انجام مقایسه‌های متعدد باشد سبب شلوغ شدن برنامه می‌شود. دستور Select Case زمانی استفاده می‌شود که یک متغیر یا حاصل یک عبارت، دارای مقادیر مختلفی باشد و بخواهیم براساس مقدار خاصی عملیات ویژه‌ای انجام دهیم. به عبارت دیگر Select Case جایگزین If‌هایی خواهد بود که می‌توانند مقادیر مختلف یک متغیر یا عبارت را بررسی کنند. در صورت به کارگیری دستور Select Case نوشتن کدها آسان‌تر و برنامه از خوانایی و دقت بیشتری برخوردار می‌شود. شکل کلی این دستور به این صورت است:

عبارت مورد مقایسه Select Case

Case مقدار اول :
دستور(ات)

Case مقدار دوم :
دستور(ات)

Case مقدار سوم :
دستور(ات)

Case Else :
دستور(ات)

End Select :
دستور(ات)

در این دستور ابتدا مقدار عبارت مورد مقایسه با مقادیری که در مقابل هر Case قرار داده شده است، مقایسه می‌شود. اگر مقدار ذخیره شده در عبارت مورد نظر با مقدار ذکر شده در مقابل اولین Case برابر باشد، دستورات موجود در این Case اجرا می‌شوند و از بررسی سایر Case‌ها صرف نظر خواهد شد؛ اما اگر اولین مقایسه، نتیجه نادرست در پی داشته باشد، Case دوم بررسی می‌شود و در صورت درست بودن نتیجه دستورات، این Case اجرا می‌شود و در غیر این صورت Case بعدی بررسی می‌شود و به همین شکل تمام Case‌ها به ترتیب بررسی می‌شوند و اگر نتیجه بررسی تمام آن‌ها نادرست باشد، دستورات موجود در بخش Case Else اجرا خواهند شد و سپس برنامه ادامه می‌یابد.



نکته استفاده از بخش Case Else اختیاری است.

برای ایجاد یک بازه از مقادیر عددی در Case‌ها، می‌توانید از کلمه کلیدی To استفاده کنید. به عنوان مثال برای مقادیر عددی بین ۱۲ تا ۱۴ می‌توانید از Case 12 To 14 استفاده کنید. برای استفاده از چند مقدار مختلف در یک Case می‌توانید از کاراکتر کاما (،)، کلمه کلیدی To، Is یا ترکیبی از آن‌ها استفاده کنید.



مثال ۱: پروژه‌ای مطابق شکل ۶-۱ و جداول ۶-۱ و ۶-۲ طراحی کنید که سه نمره یک دانش‌آموز را دریافت کند و رتبه وی را مشخص کند. اگر معدل وی بین ۱۸ تا ۲۰ باشد، رتبه A و اگر بین ۱۶ الی ۱۸ باشد رتبه B و بین ۱۴ الی ۱۶ رتبه C و کوچک‌تر از ۱۴ رتبه D برای او نمایش داده شود. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱ - یک پروژه جدید به همراه یک فرم و کنترل‌های آن مطابق شکل ۶-۱ ایجاد کنید.

جدول ۶-۱ خصوصیات فرم

| خصوصیت | مقدار |
|---------|---------|
| Name | frmave |
| Caption | Average |



شکل ۶-۱

جدول ۲-۶ خصوصیات کنترل‌ها

| کنترل خصوصیت | Label | Label | Label | TextBox | TextBox | TextBox | Command Button |
|-----------------|-------|-----------|-------|---------|---------|---------|-------------------|
| Name | lblit | lblwin | lblvb | txtit | txtwin | txtvb | cmdave |
| Caption | IT : | WINDOWS : | VB6: | — | — | — | &Average |

۲ - سپس رویداد Click را به این صورت تنظیم کنید:

```
Private Sub cmdave_Click()
```

```
    Dim sngit As Single
```

```
    Dim sngwin As Single
```

```
    Dim sngvb As Single
```

```
    Dim sngav As Single ,strrank As String*1
```

```
    sngit = Val(txtit.Text)
```

```
    sngwin = Val(txtwin.Text)
```

```
    sngvb = Val(txtvb.Text)
```

```
    sngav = (sngit+sngwin + sngvb)/3
```

```
    Select Case sngav
```

```
        Case 18 To 20
```

```
            strrank = "A"
```

```
        Case 16 To 18
```

```
            strrank = "B"
```

```
        Case 14 To 16
```

```
            strrank = "C"
```

```
        Case Is < 14
```

```
            strrank = "D"
```

```
    End Select
```

```
    MsgBox «RANK IS : «+ strrank,»RANKING»
```

```
End Sub
```

در این رویداد پس از تعریف متغیرها ابتدا مقادیر تایپ شده در کادرهای متن با استفاده از تابع Val به عدد تبدیل شده و در متغیرهای مربوطه ذخیره می‌شوند سپس متوسط سه نمره محاسبه شده و در ادامه با استفاده از یک دستور Select Case مقدار معدل (sngav) مورد ارزیابی قرار می‌گیرد.

اگر مقدار آن بین ۱۸ تا ۲۰ باشد دستور موجود در شرط Case 18 To 20 را اجرا می‌شود و رتبه A را برای وی در متغیر رشته‌ای strrank ذخیره می‌کند، سپس بدون آن که سایر شرط‌ها بررسی شوند دستور Select Case خاتمه می‌یابد؛ اما اگر مقدار متغیر sngav بین ۱۶ تا ۱۸ باشد شرط موجود در اولین Case نادرست خواهد بود و شرط Case دوم یعنی Case 16 To 18 بررسی می‌شود و در نتیجه دستور مربوط به این Case اجرا می‌شود و بدون بررسی سایر Case‌ها دستور Select Case خاتمه می‌یابد و به همین شکل در صورت نادرست بودن شرط در Case دوم، سایر Case‌ها یکی یکی بررسی می‌شوند و در Case آخر اگر مقدار متغیر sngav کوچک‌تر از ۱۴ باشد شرط Case Is < 14 درست خواهد بود و رتبه مناسب برای وی در نظر گرفته می‌شود و پس از خاتمه دستور Select Case رتبه دانش‌آموز به وسیله یک کادر پیام نمایش داده می‌شود.

۳ - پروژه و فرم را با نام Average ذخیره کنید سپس آن را اجرا نمایید.

۴ - اعداد ۱۷، ۱۹ و ۲۰ را در کادرهای متن موجود تایپ کنید، سپس روی دکمه Average کلیک کرده و نتیجه را بررسی کنید.

۵ - مجدداً اعداد ۱۴، ۵ و ۸ را در کادرهای متن موجود تایپ کرده و روی دکمه Average کلیک و نتیجه را بررسی کنید.

۶ - اجرای برنامه را متوقف کرده و به پنجره ویژوال بیسیک بازگردید.



مثال ۲: پروژه‌ای طراحی کنید که هزینه حمل کالا به وسیله وسایل نقلیه مختلف را محاسبه کند. به این منظور کاربر وزن کالا و مسافت حمل را به همراه روش حمل آن معین می‌کند و سپس هزینه حمل کالا محاسبه شده، در اختیار وی قرار می‌گیرد. روش‌های حمل کالا و هزینه هر یک به ازای هر کیلومتر جابه‌جایی در جدول ۳-۶ ارائه شده است و برای محاسبه هزینه حمل کالا از فرمول زیر استفاده می‌شود.

هزینه به ازای یک کیلومتر جابه‌جایی مسافت جابه‌جایی وزن کالا = هزینه حمل کالا


جدول ۳-۶

| روش حمل کالا | هزینه حمل به ریال |
|--------------|-------------------|
| اتوبوس | ۱۰۰۰۰ |
| قطار | ۱۵۰۰۰ |
| کشتی | ۱۴۰۰۰ |
| هواپیما | ۲۰۰۰۰ |



شکل ۲-۶

اکنون این عملیات را به ترتیب انجام دهید:
 ۱ - یک پروژه از نوع Standard EXE به همراه یک فرم، دو کادر متن و برچسب و یک دکمه فرمان مطابق شکل ۲-۶ ایجاد کنید.
 کادر متن اول برای ورود وزن کالا و کادر متن دوم برای ورود مسافت حمل کالا استفاده می‌شود.

۲ - برای آن‌که کاربر توانایی انتخاب نوع روش حمل کالا را داشته باشد، از کنترل دکمه انتخاب (RadioButton) استفاده کنید. این کنترل اجازه می‌دهد تا امکان انتخاب یک گزینه از بین چند انتخاب برای کاربران فراهم شود. این کنترل معمولاً به صورت گروه‌هایی دوتایی یا بیشتر استفاده می‌شود و هر لحظه می‌توان یکی از کنترل‌ها را از مجموعه آن‌ها انتخاب کرد. برای استفاده از این کنترل در جعبه ابزار روی کنترل دکمه انتخاب  کلیک کنید و چهار کنترل از آن را مطابق شکل ۲-۶ روی فرم قرار دهید و خصوصیات آن‌ها را تنظیم کنید.

۳ - در پنجره خصوصیات، optbus را انتخاب کنید و خصوصیت Value آن را روی مقدار True تنظیم کنید. کنترل‌های دکمه انتخاب دارای خصوصیتی به نام Value هستند که یک خصوصیت منطقی است و می‌تواند True یا False باشد. اگر مقدار این خصوصیت روی True تنظیم شود نشان دهنده این است که دکمه انتخاب مربوطه انتخاب شده است و در غیر این صورت نشان‌دهنده عدم انتخاب کنترل است. همواره در یک گروه از دکمه‌های انتخاب، خصوصیت Value یکی از کنترل‌های دکمه انتخاب، True و برای سایر کنترل‌ها False می‌باشد.

۴ - اکنون باید دستورات مورد نظر را در رویداد Click دکمه فرمان Compute بنویسید.

بنابراین رویداد مزبور را به این صورت تنظیم کنید:

```
Private Sub cmdcom_Click()
    Dim intrans As Integer, sngprice As Currency
    Dim sngw As Single, sngl As Single
    If optbus.Value = True Then intrans=0
    If opttrain.Value = True Then intrans =1
    If optship.Value = True Then intrans =2
    If optplane.Value = True Then intrans =3
    sngw = Val(txtw.Text)
    sngl = Val(txtl.Text)
    Select Case intrans
        Case 0
            sngprice =sngw * sngl *10000
        Case 1
            sngprice=sngw*sngl*15000
        Case 2
            sngprice = sngw * sngl *14000
        Case 3
            sngprice = sngw * sngl *20000
    End Select
    MsgBox "Total Price =" + Str(sngprice)+ "Rials", , " Payment"
End Sub
```

در این رویداد با استفاده از دستور If مقدار خصوصیت Value هر یک از کنترل‌های دکمه انتخاب بررسی می‌شود و در صورت انتخاب هر یک از آن‌ها متغیر intrans متناسب با روش حمل انتخاب شده مقداردهی می‌شود؛ بنابراین در صورت انتخاب روش حمل با اتوبوس مقدار صفر، حمل با قطار مقدار ۱، حمل با کشتی مقدار ۲ و حمل با هواپیما مقدار ۳ در آن ذخیره می‌شود. سپس با استفاده از دستور Select Case مقدار متغیر intrans بررسی می‌شود تا با توجه به روش حمل انتخاب شده، هزینه حمل کالا محاسبه و در متغیر sngprice ذخیره شود، در نهایت این مقدار به وسیله یک کادر پیام نمایش داده خواهد شد.

۵- برای آن که ترتیب دریافت فوکوس در کنترل به صورت مناسب انجام شود،

خصوصیت TabIndex دکمه فرمان Compute را روی عدد ۶ تنظیم کنید. در این صورت

دکمه فرمان Compute آخرین کنترلی است که در زمان فشردن کلید Tab، فوکوس را به دست می آورد.

۶ - پروژه و فرم را با نام transport ذخیره کنید.


۷ - پروژه را اجرا کنید و وزن کالا را ۲۰۰ کیلوگرم و مسافت جابه جایی را ۱۰۰ کیلومتر تایپ کنید، سپس روی دکمه انتخاب Plane کلیک کنید تا روش حمل کالا نیز انتخاب شود.

۸ - روی دکمه Compute کلیک کنید و نتیجه محاسبه را بررسی کنید.

۹ - مجدداً مقادیر ۳۵۰ و ۲۰۰ را برای وزن کالا و مسافت جابه جایی تایپ کنید و

این بار روش حمل با قطار را انتخاب کرده، روی دکمه Compute کلیک کنید و نتیجه را بررسی نمایید.

۱۰ - اجرای برنامه را خاتمه داده و به پنجره ویژوال بیسیک بازگردید.

 **نکته:** اگر بخواهید از کنترل های دکمه انتخاب به گونه ای استفاده کنید که به صورت گروه های جداگانه عمل کنند باید از کنترل دیگری به نام کنترل قاب یا Frame استفاده کنید، در غیر این صورت تمام کنترل های دکمه انتخاب که در یک فرم قرار دارند به صورت یک گروه واحد در نظر گرفته خواهند شد. برای ایجاد گروه های مستقل از کنترل های دکمه انتخاب، باید ابتدا کنترل قاب را روی فرم قرار دهید و سپس با روش درگ، کنترل های دکمه انتخاب را روی کنترل قاب بگذارید. کنترل قاب، رویداد و خصوصیت ویژه ای ندارد و مهم ترین خصوصیت آن خصوصیت Caption است که عنوان کنترل قاب را تعیین می کند.

۲-۶ ساختارهای تکرار در ویژوال بیسیک

بعضی اوقات لازم است تا عملیاتی را به صورت تکراری انجام دهید. در ویژوال بیسیک دستورات متعددی برای انجام عملیات تکراری قرار داده شده اند که بعضی از آنها توانایی اجرای عملیات تکراری را با تعداد دفعات معین دارند و بعضی دیگر عملیات مورد نظر را تا رسیدن به شرایط خاصی فراهم می کنند. در این واحد کار به توضیح بعضی از آنها می پردازیم.


۱-۲-۶ حلقه For ... Next

از این حلقه زمانی استفاده می‌شود که لازم باشد دستورات را به تعداد دفعات معین و محدودی اجرا کنید. نحوه استفاده از این حلقه به این صورت است:

مقدار خاتمه To مقدار شروع = شمارنده حلقه For

 دستورات
 Next شمارنده حلقه

برای استفاده از این گونه حلقه‌ها از یک متغیر به عنوان شمارنده حلقه استفاده می‌شود که به وسیله آن تعداد دفعات تکرار حلقه کنترل می‌شود. مقادیر شروع و خاتمه با توجه به نیاز برنامه‌نویس تنظیم می‌شود و معمولاً از مقادیر عددی صحیح استفاده می‌شود. اما می‌توان از هر نوع متغیر عددی یا عبارات ریاضی نیز استفاده کرد. این حلقه با کلمه کلیدی For آغاز شده و با کلمه کلیدی Next که به همراه نام متغیر شمارنده ذکر می‌شود، خاتمه می‌یابد و دستورات مورد نظر برای اجرا در بین آن‌ها قرار می‌گیرند. زمانی که اجرای برنامه به اولین خط از حلقه یعنی For می‌رسد، مقدار شروع در متغیر شمارنده ذخیره می‌شود و سپس این مقدار با مقدار خاتمه مقایسه می‌شود و در صورتی که کوچک‌تر یا مساوی با مقدار خاتمه باشد، دستورات بین For و Next اجرا می‌شود. با رسیدن به کلمه کلیدی Next، اجرای برنامه مجدداً به بخش For منتقل می‌شود، سپس یک واحد به مقدار قبلی متغیر شمارنده اضافه شده و در صورتی که این مقدار کوچک‌تر یا مساوی با مقدار خاتمه باشد، دستورات موجود در حلقه اجرا می‌شود و این عملیات تا زمانی که مقدار متغیر شمارنده از مقدار خاتمه بزرگ‌تر شود، ادامه می‌یابد. پس از خاتمه اجرای حلقه، اجرای دستورات بعد از کلمه کلیدی Next انجام می‌شود.

 **مثال ۳:** پروژه‌ای طراحی کنید که مجموع اعداد ۱ تا ۱۰ را محاسبه کرده و روی فرم نمایش دهد. به این منظور عملیات زیر را به ترتیب انجام دهید:



شکل ۳-۶

۱ - برنامه ویژوال بیسیک را اجرا کنید
 سپس یک پروژه از نوع Standard EXE به همراه یک فرم و یک کنترل دکمه فرمان و یک کنترل برچسب مطابق شکل ۳-۶ ایجاد کنید.

۲ - رویداد Click دکمه فرمان cmdshow را به صورت زیر تنظیم کنید:

Private Sub cmdshow_Click()

Dim inti As Integer

Dim intsum As Integer

For inti = 1 To 10

intsum = intsum + inti

Next inti

Iblsum. Caption = Iblsum. Caption + Str (intsum)

End Sub

در این رویداد ابتدا متغیرهای مورد نیاز تعریف می‌شوند. به منظور محاسبه مجموع اعداد ۱ تا ۱۰ از یک حلقه For استفاده شده است. این حلقه از عدد ۱ شروع و تا رسیدن حلقه به مقدار ۱۰ تکرار می‌شود و در هر مرحله از اجرای حلقه نیز با استفاده از متغیر intsum مجموع اعداد محاسبه می‌شود و با پایان یافتن حلقه مقدار مجموع intsum پس از تبدیل به نوع داده رشته‌ای در کنترل برچسب نمایش داده می‌شود.

۳- پروژه و فرم را با نام sumnumbers ذخیره کنید سپس آن را اجرا کرده و آزمایش نمایید.

۴- به اجرای پروژه خاتمه داده و به پنجره ویژوال بیسیک بازگردید.



مثال ۴: پروژه‌ای طراحی کنید که یک عدد طبیعی را از ورودی دریافت کند و مجموع اعداد زوج کوچک‌تر یا مساوی آن را نمایش دهد.

۱- برنامه ویژوال بیسیک را اجرا کرده، یک پروژه به همراه یک فرم، سه کنترل

برچسب و یک کنترل کادر متن به همراه یک کنترل دکمه فرمان مطابق شکل ۴-۶ و جدول ۴-۶ ایجاد کنید.

جدول ۴-۶ خصوصیات فرم

| خصوصیت | مقدار |
|---------|-------------|
| Name | frmsn |
| Caption | Show Number |



شکل ۴-۶

۲ - سپس رویداد Click دکمه Show Number را به این صورت تنظیم کنید:

```
Private Sub cmdshow_Click()
    Dim inti As Integer
    Dim intno As Integer
    Dim lngsum As Long
    intno = Val(txtno.Text)
    For inti = 1 To intno
        If intno mod inti=0
            lngsum=lngsum+inti
        End If
    Next inti
    lblsum.Caption=lblsum.Caption+Str(lngsum)
End Sub
```

۳ - پروژه و فرم را با نام shownumber ذخیره کنید سپس آن را اجرا کرده و آزمایش کنید.

۴ - اجرای پروژه را متوقف کرده و به پنجره ویژوال بیسیک باز گردید. اکنون شکل کلی حلقه For به شما معرفی می‌شود. همان‌طور که گفته شد متغیر شمارنده حلقه در هر بار اجرای حلقه به میزان یک واحد افزایش می‌یابد، اما گاهی اوقات لازم است تا میزان افزایش شمارنده به وسیله برنامه‌نویس تعیین شود یا از حلقه‌های کاهش استفاده شود که در آن‌ها مقدار شروع از مقدار خاتمه بزرگ‌تر است و در هر بار اجرای حلقه مقدار شمارنده کم می‌شود. شکل کلی دستور For به این صورت است:

```
For میزان افزایش یا کاهش شمارنده Step مقدار خاتمه To مقدار شروع = شمارنده حلقه
    ↓
    دستورات
Next شمارنده حلقه
```

در واقع بخش Step امکان تنظیم میزان افزایش یا کاهش مقدار شمارنده حلقه را فراهم می‌کند. مقدار پیش فرض برای این بخش مقدار یک است. می‌توانید برای حلقه‌های افزایشی از مقادیر مثبت و برای حلقه‌های کاهش‌دهنده از مقادیر منفی استفاده کنید. در این صورت حلقه زمانی متوقف می‌شود که مقدار شروع از مقدار خاتمه کوچک‌تر شود.



نکته در صورتی که مقدار شروع از مقدار خاتمه بزرگ تر باشد و مقدار Step مثبت باشد، حلقه اجرا نخواهد شد.

در صورت عدم استفاده از بخش Step مقدار افزایش شمارنده حلقه یک واحد خواهد بود

دستور Print

دستور Print می تواند انواع مقادیر ثابت، متغیر و مقدار خصوصیت اشیا را روی فرم نمایش دهد. در صورتی که از این دستور به تنهایی استفاده شود فقط یک خط خالی نمایش داده می شود و اگر از کاراکتر سمی کالن (;) استفاده شود، خروجی ها پشت سرهم و بدون فاصله قرار می گیرند و اگر از کاراکتر کاما (,) استفاده شود، هر خط نمایشی در روی فرم، توانایی نمایش ۱۰ کاراکتر را خواهد داشت و اگر خروجی بزرگ تر از ۱۰ کاراکتر باشد، خروجی بعدی در یک ناحیه جلوتر نمایش داده خواهد شد. به عنوان مثال به این دستورات توجه کنید:

```
Print "Ali" ; "Reza"
```

```
Print "Visual", "Basic"
```

```
Print "Student" ;
```

```
Print "Information"
```

در صورت اجرای دستورات فوق خروجی حاصل به این شکل خواهد بود:

```
AliReza
```

```
Visual Basic
```

```
StudentInformation
```

عبارات موجود در دستور اول به دلیل استفاده از کاراکتر سمی کالن پشت سرهم و عبارات موجود در دستور دوم به دلیل استفاده از کاراکتر کاما با فاصله از هم قرار گرفته اند و در دو دستور آخر نیز چون Print سوم به کاراکتر سمی کالن ختم می شود، خروجی Print چهارم پشت سر آن نمایش داده می شود.

به عنوان مثال به این دستورات توجه کنید و نتیجه اجرای آن ها را در شکل ۵-۶ مشاهده کنید.

```
Print "1234567890" , "1234567890"
```

```
Print "12345678901234567890"
```

```
Print "12345678901" , "1234567890"
```



شکل ۵-۶

در دستور Print اول هر عبارت نمایشی از ده کاراکتر تشکیل شده است؛ بنابراین دو عبارت نمایشی به ترتیب در نواحی اول و دوم در کنار هم قرار می‌گیرند. اما در دستور Print آخر چون عبارت نمایشی اول از ۱۱ کاراکتر تشکیل شده است، عبارت نمایشی دوم در ناحیه ۳ نمایش داده می‌شود. دستور Print دوم برای تشخیص بهتر اندازه نواحی استفاده شده است.

تمرین:



پروژه‌ای را طراحی کنید تا مطابق شکل ۶-۶ توانایی نمایش اعداد فرد و زوج کوچک‌تر از هر عدد طبیعی دلخواه موردنظر کاربر را داشته باشد. برای دریافت داده‌ها از کادر ورود داده استفاده کنید.



شکل ۶-۶

حلقه‌های For را می‌توان در داخل یکدیگر قرار داد و در این حالت با هر بار اجرای حلقه For اول، حلقه For داخل آن یک دور کامل اجرا می‌شود. شکل کلی نحوه استفاده از حلقه‌های متداخل به صورت زیر است:

میزان افزایش یا کاهش شمارنده Step مقدار خاتمه To شمارنده حلقه = مقدار شروع For

میزان افزایش یا کاهش شمارنده Step مقدار خاتمه To شمارنده حلقه = مقدار شروع For

.....
دستور(ات)

Next شمارنده حلقه

Next شمارنده حلقه



شکل ۶-۷

مثال ۵: پروژه‌ای طراحی کنید که با استفاده



از حلقه For کاراکترهای ستاره را مطابق شکل ۶-۷ نمایش دهد. برای این کار عملیات بعد را انجام دهید.

۱ - یک پروژه از نوع Standard EXE به همراه یک فرم و یک کنترل دکمه فرمان مطابق شکل ۶-۷ ایجاد کنید.

۲ - رویداد Click دکمه فرمان را به این صورت تنظیم کنید:

```
Private Sub cmdsshow_Click()
```

```
    Dim inti As Integer
```

```
    Dim intj As Integer
```

```
    For inti = 1 To 4
```

```
        For intj= 1 To 5
```

```
            Print "*";
```

```
        Next intj
```

```
    Print
```

```
Next inti
```

```
End Sub
```

۳ - پروژه و فرم را با نام showstart ذخیره کنید سپس برنامه را اجرا کرده و روی دکمه show کلیک کنید.

۴ - به اجرای برنامه خاتمه داده و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه‌ای طراحی کنید که شکل زیر را رسم نماید.

*

**

۲-۲-۶ حلقه While ... Wend و While ... Loop

این حلقه‌ها، دستورات را تا زمانی که شرط یا شرط‌های تعیین شده درست باشند، اجرا می‌کنند. از این نوع حلقه‌ها زمانی استفاده می‌شود که دفعات تکرار دستورات معین نباشد. شکل کلی این دستورات به صورت زیر است:

While (شرط یا شرط‌ها)
⋮
دستورات

Do While (شرط یا شرط‌ها)
⋮
دستورات

Wend

Loop

زمانی که اجرای برنامه به حلقه While برسد، ابتدا شرط یا شرط‌های موجود در جلوی کلمه کلیدی While ارزیابی می‌شوند و اگر نتیجه این ارزیابی درست باشد، دستورات بین While و Wend اجرا می‌شوند. با رسیدن به انتهای حلقه یعنی کلمه کلیدی Wend اجرای برنامه مجدداً به بخش While منتقل می‌شود و بررسی شرط یا شرط‌ها انجام می‌گیرد و به همین شکل اجرای حلقه تا زمانی که نتیجه ارزیابی شرط موجود در حلقه نادرست شود، ادامه می‌یابد. نحوه اجرای حلقه Do While مانند حلقه While است.



مثال ۶: پروژه‌ای طراحی کنید که هر بار مسافت بین دو شهر را براساس کیلومتر دریافت کند سپس مسافت دو شهر را به متر تبدیل کرده و نمایش دهد و این کار را تا دریافت مقدار صفر انجام دهد.

۱ - یک پروژه از نوع Standard EXE به همراه یک فرم مطابق با شکل ۸-۶ ایجاد کنید.



شکل ۸-۶

۲ - یک کنترل برچسب و دو کنترل دکمه فرمان روی فرم قرار داده و خصوصیت‌های آنها را تنظیم کنید.

۳ - رویداد دکمه Compute را به این صورت تنظیم کنید:

```
Private Sub cmdcompute_Click ()
```

```
Dim intno As Integer
```

```
Dim strdata As String
```

```
intno = 1
```

```
While (intno > ۰)
```

```
strdata = InputBox ("Enter Your Number:" , "Enter Data" , ۰)
```

```
intno = Val (strdata)
```

```
lblresult. Caption = intno*1000
```

```
Wend
```

```
End Sub
```

در این رویداد پس از تعریف متغیرها ابتدا مقدار متغیر `intno` روی مقدار ۱ تنظیم می‌شود تا حلقه `While` بتواند اجرا شود. زیرا شرطی که برای حلقه در نظر گرفته شده است `intno < ۰` می‌باشد اگر `intno` مقداره‌ی نشود هیچ وقت امکان ورود داده و انجام محاسبات وجود نخواهد داشت. در مرحله بعد حلقه `While` اجرا می‌شود و با استفاده از یک کادر ورود مسافت دو شهر دلخواه دریافت شده و در متغیر `strdata` ذخیره می‌شود سپس با استفاده از تابع `Val` محتویات متغیر `strdata` به عدد تبدیل شده و در متغیر `intno` قرار می‌گیرد سپس با تبدیل این مقدار به واحد متر، مسافت در کنترل برچسب نمایش داده می‌شود. در این مرحله با رسیدن به دستور `Wend` اجرای برنامه به ابتدای حلقه یعنی دستور `While` منتقل می‌شود در نتیجه شرط `intno < ۰` بررسی شده و اگر نتیجه بررسی شرط درست باشد عملیات تکرار خواهد شد اما اگر نتیجه بررسی شرط نادرست باشد (در صورت ورود عدد صفر) اجرای حلقه خاتمه می‌یابد.

۴ - پروژه و فرم را با نام `Distance` ذخیره کنید.

۵ - پروژه را اجرا کرده و روی دکمه `Compute` کلیک کنید و پس از ورود چند مقدار مختلف عدد صفر را وارد کنید تا مجدداً به پنجره برنامه بازگردید.

۶ - روی دکمه `Exit` کلیک کنید و از برنامه خارج شده و به پنجره ویژوال بیسیک

بازگردید.



تمرین:

پروژه‌ای طراحی کنید تا مضرب‌های کوچک‌تر از ۱۰۰ عدد ۹ را روی فرم نمایش دهد. به این منظور عملیات زیر را به ترتیب انجام دهید:

۳-۲-۶ حلقه Do ... Loop While

این حلقه مشابه حلقه Do While ... Loop است؛ با این تفاوت که دستورات داخل حلقه یک بار اجرا شده، سپس شرط حلقه بررسی می‌شود. بنابراین در صورت نادرست بودن شرط، دستورات حداقل یک بار اجرا می‌شوند. شکل کلی این دستور به صورت زیر است:

Do



دستورات)

Loop While (شرط یا شرطها)



مثال ۷: پروژه‌ای طراحی کنید که یک عدد طبیعی را دریافت کند و مجموع ارقام آن را نمایش دهد. برای این کار عملیات زیر را به ترتیب انجام دهید:

- ۱- یک پروژه از نوع Standard EXE به همراه یک فرم با مشخصات جدول ۱۹-۷ ایجاد کنید.
- ۲- در این پروژه به منظور اجرای دستورات از رویداد Activate فرم استفاده می‌شود. این رویداد بعد از رویداد Load فرم و زمانی که فرم نمایش داده شده و فوکوس را کسب می‌کند اجرا می‌شود. بنابراین رویداد Activate فرم را به این صورت تنظیم کنید:

```
Private Sub Form_Activate()
```

```
Dim lngi As Long
```

```
Dim lngno As Long
```

```
Dim strinput As String
```

```
strinput = InputBox("Enter Your Number :","input DATA",0)
```

```
lngno = Val(strinput)
```

```
Do
```

```
lngi = lngno Mod 10
```

```
Sum=Sum+lngi
```

```
Ingno = Ingno \ 10  
Loop While (Ingno > 0)  
End Sub
```

۳ - اجرای پروژه را متوقف کرده و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه طراحی کنید که مضارب کوچکتر از ۱۰۰ عدد ۷ را نمایش دهد.

۴-۲-۶ دستورات خروج از حلقه Exit For و Exit Do

دستورات Exit For و Exit Do به برنامه نویس امکان می دهند که قبل از آن که حلقه به طور عادی خاتمه یابد به اجرای حلقه خاتمه دهد و از آن خارج شود. از دستور Exit For برای خروج از حلقه Next...For و از دستور Exit Do برای خروج از حلقه های Do...Loop While، Do While...Loop، Do...Loop Until و Do Until...Loop استفاده می شود.

مثال ۸: پروژه digit را به گونه ای تغییر دهید تا با استفاده از دستور Exit Do اجرای حلقه خاتمه یابد. به این منظور عملیات زیر را به ترتیب انجام دهید:
۱ - پروژه digit را باز کنید و رویداد Activate فرم را به این صورت تغییر دهید:

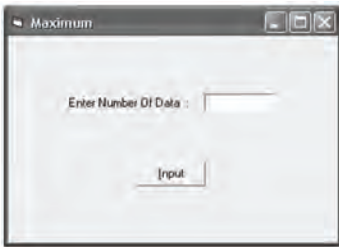


```
Private Sub Form_Activate()  
Dim lngi As Long  
Dim lngno As Long  
Dim strinput As String  
strinput = InputBox(«Enter Your Number :», «Input Data», 0)  
lngno = Val(strinput)  
Do  
    lngi = lngno Mod 10  
    sum = sum + lngi  
    lngno = lngno \ 10  
    If lngno < 0 Then Exit Do  
Loop While (True)  
End Sub
```


در این رویداد نیز مانند راه حل قبلی حلقه Do...Loop While به کار گرفته شده است. با این تفاوت که به جای شرط $Ingno > 0$ از مقدار True استفاده شده است بنابراین حلقه می‌تواند بدون محدودیت بارها اجرا شود اما با توجه به حالت پروژه باید اجرای حلقه با توجه به تعداد ارقام عدد وارد شده تنظیم شود. به این منظور از یک دستور If دیگر در انتهای حلقه استفاده شده است که شرط $Ingno < 0$ را بررسی می‌کند در صورتی که نتیجه این بررسی درست باشد (یعنی دیگر رقمی در عدد باقی نمانده باشد) با دستور Exit Do از اجرای مجدد حلقه جلوگیری به عمل می‌آورد.

۲ - تغییرات را ذخیره کرده و پروژه را اجرا و آزمایش کنید.

۳ - به اجرای پروژه خاتمه داده و به پنجره ویژوال بیسیک بازگردید.



شکل ۹-۶

مثال ۹: پروژه‌ای طراحی کنید که بزرگ‌ترین

عدد را بین n عدد دلخواه جستجو کرده و نمایش

دهد. برای این کار عملیات زیر را انجام دهید:

۱- یک پروژه جدید مطابق شکل ۹-۶ ایجاد کنید.

۲ - مطابق شکل ۹-۶ ابتدا کاربر تعداد اعداد خود را در کادر متن وارد می‌کند؛ سپس

با کلیک روی دکمه Input و با استفاده از کادرهای ورود داده اعداد خود را وارد می‌کند و

با پایان ورود اعداد، مقدار ماکزیمم روی فرم نمایش داده می‌شود. به علاوه در صورتی که

کاربر کاراکتر Q را در کادر ورود داده تایپ کند، برنامه از دریافت اعداد بعدی خودداری

می‌نماید. بنابراین رویداد Click دکمه Input را به این صورت تنظیم کنید:

```
Private Sub cmdinput_Click()
```

```
Dim inti As Integer, strno As String
```

```
Dim sngmax As Single
```

```
For inti = 1 To Val(txtno.Text)
```

```
strno = InputBox("Enter Your Number:", "InputBox", 0)
```

```
If strno = "Q" or strno = "q" Then Exit For
```

```
If inti = 1 Then
```

```
sngmax = Val(strno)
```

```
Else
```

```
If sngmax < Val (strno)Then sngmax =Val(strno)
```

```
End If
```

```
Next inti
```

```
MsgBox "Maximum Is :"+Str(sngmax)
```

```
End Sub
```

در این رویداد از یک حلقه For...Next برای دریافت اعداد استفاده خواهد شد. مقدار خاتمه حلقه با توجه به مقداری که کاربر در کادر متن مربوطه وارد می کند، تنظیم می شود. بخش عمده ای از دستورات موجود در حلقه برای محاسبه بزرگترین عدد به کار می روند، اما پس از آن که این رویداد اجرا شد، ابتدا با استفاده از یک کادر ورود داده مقدار اولین عدد را از کاربر دریافت می کند و بلافاصله مقدار تایپ شده در کادر ورود داده با یک دستور If بررسی می شود و اگر کاراکتر Q یا q تایپ شده باشد نتیجه شرط موجود درست بوده و در نتیجه دستور Exit For اجرا می شود که سبب خروج از حلقه و خاتمه دریافت اعداد خواهد شد؛ اما در غیر این صورت دستور If دوم بررسی می شود که در این صورت اگر اولین عدد داده شده باشد (یعنی شمارنده حلقه برابر با یک باشد) اولین عدد به عنوان بزرگترین عدد در نظر گرفته می شود و در متغیر sngmax ذخیره می شود، اما برای اعداد بعدی ابتدا مقدار تایپ شده در کادر ورود داده با مقدار ذخیره شده در متغیر sngmax مقایسه می شود و در صورتی که از آن بزرگتر باشد، مقدار جدید به عنوان بزرگترین مقدار، در متغیر sngmax ذخیره خواهد شد. به این ترتیب بزرگترین مقدار، محاسبه و در یک کادر پیام نمایش داده می شود.

۳ - پروژه و فرم را با نام maximum ذخیره کنید، سپس آن را اجرا نمایید.

۴ - مقدار ۱۰ را در کادر متن تایپ کرده و روی دکمه Input کلیک کنید.

۵ - اعداد ۷، ۲ و ۱۸ را به ترتیب وارد کنید، سپس کاراکتر Q را در کادر ورود داده

چهارم تایپ کنید و روی دکمه OK کلیک کنید. همان طور که مشاهده می کنید از ورود داده های بعدی جلوگیری به عمل می آید و بزرگترین عدد بین سه عدد وارد شده (۱۸) نمایش داده می شود.

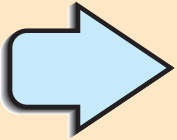
۶ - پروژه را برای مقادیر دیگری آزمایش کنید، سپس اجرای پروژه را پایان داده و به

پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه قبل را به گونه ای تنظیم کنید که توانایی محاسبه کوچکترین عدد را نیز داشته باشد.



Learn in English

For...Next Statement

.Repeats a group of statements a specified number of times

Syntax

For counter = start **To** end [*step*]

[*statements*]

Next [*counter*]

The **For...Next** statement syntax has these parts:

| Part | Description |
|------------|--|
| counter | Required. Numeric variable used as a loop counter. The variable can't be a Boolean or an array element. |
| start | Required. Initial value of counter. |
| end | Required. Final value of counter. |
| step | Optional. Amount counter is changed each time through the loop. If not specified, step defaults to one. |
| statements | Optional. One or more statements between For and Next that are executed the specified number of times. |

واژه‌نامه

| | |
|-------------|----------------------|
| Compute | محاسبه کردن |
| Counter | شمارنده |
| Even | زوج |
| Exit | خروج |
| Final | نهایی، آخری |
| Information | اطلاعات |
| Initial | ابتدایی، اولین، اصلی |
| Odd | فرد |
| Optional | اختیاری |
| Payment | مبلغ، بهای پرداختی |
| Price | بها، قیمت |
| Rank | رتبه |
| Required | لازم داشتن |
| Specify | تعیین کردن |
| Statement | دستور |
| Step | مرحله |
| Transport | حمل کردن |

خلاصه مطالب

- از دستور Select Case برای ارزیابی چند مقایسه به صورت یکجا استفاده می‌شود.
- از کنترل دکمه انتخاب (Option Button) برای ایجاد گزینه‌های متعدد، برای انتخاب کاربر استفاده می‌شود.
- برای اجرای دستورات با دفعات تکرار معین از دستور For ... Next استفاده می‌شود.
- حلقه‌های Do...Loop While و Do While ... Loop، While ... Wend و دستورات را تا زمانی که نتیجه شرط یا شرط‌های ذکر شده در آن‌ها درست باشند، اجرا می‌کنند.
- حلقه‌های Do...Loop Until و Do Until...Loop و دستورات را تا زمانی که نتیجه شرط یا شرط‌های ذکر شده در آن‌ها نادرست باشند، اجرا می‌کنند.

- دستور Print می‌تواند انواع پیام‌ها، عبارات رشته‌ای، مقدار متغیرها و خصوصیات فرم‌ها و کنترل‌ها را روی فرم نمایش دهد.
- از دستور Exit Do برای خروج از حلقه Do While...Loop، Do While...Loop While، Do...Loop Until و Do Until...Loop استفاده می‌شود.
- از دستور Exit For برای خروج از حلقه For ... Next استفاده می‌شود.
- از کنترل قاب (Frame) برای دسته‌بندی کنترل‌ها استفاده می‌شود.
- برای اضافه کردن یک پروژه جدید از نوع Standard EXE می‌توان از گزینه New Project در منوی File ویژوال بیسیک یا از دکمه Add Standard EXE Project در نوار ابزار استاندارد استفاده کرد.
- به منظور دسترسی به خاصیت‌ها و متدهای کنترل یک فرم در فرم دیگر می‌توان به صورت زیر عمل نمود:

نام متد یا خاصیت. نام کنترل. نام فرم

آزمون نظری

۱ - کدام حلقه شبیه به حلقه Do While ... Loop است؟

الف - For...Next ب - While ...Wend ج - Select Case د - While...Loop

۲ - پس از اجرای برنامه زیر مقدار متغیر Sum چقدر است؟

Sum= ۰

I=5

Do While (I>3)

For k=1 to 2

Sum = Sum+k*I

Next k

I=I-1

Loop

د- ۳۰

ج- ۲۹

ب- ۲۸

الف- ۲۷

۳ - حلقه زیر چند بار تکرار می شود؟

For i=7 To -2 Step -3

.....

Next i

د- ۵

ج- ۴

ب- ۳

الف- ۲

۴ - حلقه زیر چند بار تکرار می شود؟

While (True)

.....

Wend

الف- ۳۰ بار ب- ۴۰ بار ج- بی نهایت د- تکرار نمی شود.

۵ - خروجی برنامه پس از اجرای حلقه زیر چیست؟

i=6

While (i>4)

j=6

i=i - 1

Do While (j>3)

Print i,j

j=j - 1

Loop

Wend

| | | | | | | | |
|------|-----|----|-----|----|-----|----|-----|
| الف- | ۵ ۶ | ب- | ۵ ۵ | ج- | ۴ ۵ | د- | ۴ ۶ |
| | ۵ ۵ | | ۵ ۶ | | ۴ ۶ | | ۴ ۵ |
| | ۵ ۴ | | ۵ ۴ | | ۴ ۴ | | ۴ ۴ |
| | ۴ ۶ | | ۴ ۵ | | ۵ ۵ | | ۵ ۶ |
| | ۴ ۵ | | ۴ ۶ | | ۵ ۶ | | ۵ ۵ |
| | ۴ ۴ | | ۴ ۴ | | ۵ ۴ | | ۵ ۴ |

۶- خروجی برنامه پس از اجرای حلقه زیر چیست؟

For i=10 To 1 Step 2

Print "i" ;i

Next i

- الف- اعداد زوج بین ۱ و ۱۰ ب- اعداد فرد بین ۱ و ۱۰
- ج- اعداد ۱۰ تا ۱ د- هیچ‌گونه خروجی ندارد.
- ۷- در صورتی که مقدار متغیر intx برابر ۲ باشد، خروجی دستورات زیر چیست؟

Select Case intx

Case Is > 10 : Print intx*10

Case Is <= 4 : Print intx*4

Case Is = 7 : Print intx*2

End Select

الف- ۱۰ ب- ۸ ج- ۴ د- ۲

دستور Exit Do سبب خروج از کدام حلقه می‌شود؟

- الف- For ...Next ب- While ... Wend
- ج- Do While ... Loop د- گزینه‌های ب و ج صحیح هستند.

۹ - کدام حلقه مشابه حلقه زیر است؟

```
For i=4 to 1 Step -1
    ⋮
Next i
```

الف -

```
i=4
While (i>=1)
    ⋮
i=i+ 1
Wend
i=4
```

ب (

```
Do While (i>=1)
    ⋮
i = i- 1
Loop
```

ج -

```
i =4
While (i<=1)
    ⋮
i=i - 1
Wend
```

د -

```
i = 4
Do While (i >=1)
    ⋮
i = i + 1
Loop
```

۱۰ - کدام گزینه در رابطه با دستورات زیر صحیح است؟

```
For i=1 To 15
    If i Mod 3=0 Then Print i
Next i
```


الف- اعداد ۱ تا ۱۵ را نمایش می‌دهد.

ب- مضارب عدد ۳ از ۱ تا ۱۵ را نمایش می‌دهد.

ج- اعداد ۱ تا ۱۵ را که بر ۳ بخش پذیر هستند نمایش می‌دهد.

د- باقیمانده تقسیم اعداد ۱ تا ۱۵ را بر ۳ نمایش می‌دهد.

۱۱- برای دسته‌بندی کنترل‌ها از کنترل استفاده می‌شود.

الف- CheckBox ب- Frame ج- GroupBox د- Label

۱۲- با استفاده از کدام خصوصیت می‌توان کنترل OptionButton را در حالت انتخاب قرار داد؟

الف- Check ب- Index ج- Value د- Tab Index

۱۳- حلقه زیر چند بار تکرار می‌شود؟

k=2

Do

⋮

k=k*2

Loop Until (k>10)

د- ۲

ج- ۳

ب- ۴

الف- ۵

۱۴- In the For Statement, the step argument can be.....number.

a- positive

b- negative

c- integer

d- positive or negative

۱۵- تفاوت بین حلقه‌های While با حلقه‌های Until چیست؟

۱۶- کاربرد دستور Exit Do و Exit For را بیان کنید.

۱۷- نحوه عملکرد دستور For را همراه با بخش Step توضیح دهید.

۱۸- تفاوت دستور Select Case را با IF بیان کنید.

۱۹- کنترل دکمه انتخاب (Option Button) و قاب (Frame) را توضیح داده و کاربرد

هر یک از آن‌ها را بیان کنید.

۲۰- نحوه اضافه کردن یک پروژ به جدید به پروژه موجود را توضیح دهید.

آزمون عملی

- ۱ - پروژه‌ای طراحی کنید که مجموع اعداد سه رقمی را محاسبه کرده و نمایش دهد.
- ۲ - پروژه‌ای طراحی کنید که عدد طبیعی و دلخواهی را از کاربر دریافت کند و مقسوم علیه‌های زوج آن را محاسبه کرده و نمایش دهد.
- ۳ - پروژه‌ای طراحی کنید که یک جدول ضرب 10×10 را چاپ کند.
- ۴ - پروژه‌ای طراحی کنید که دو عدد طبیعی را دریافت کند و کوچک‌ترین مضرب مشترک آن‌ها را نمایش دهد.
- ۵ - خروجی حاصل از اجرای حلقه‌های زیر چه خواهد بود؟

الف -

ب -

```
For I= 1 to 4
    For j=1 to 3
        Print "*";
    Next j
    Print
Next I

J=10
Do While (True)
    If j>5 Then Exit Do
    For i=1 to j
        Print i;
    Next i
    Print
    j=j+ 1
Loop
```

- ۶ - پروژه‌ای طراحی کنید که دو عدد را دریافت کرده و بزرگ‌ترین مقسوم‌علیه آن‌ها را نمایش دهد.

توانایی ایجاد و استفاده از انواع رویه‌ها در ویژوال بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

- ۱- انواع رویه‌ها را در زبان برنامه‌نویسی ویژوال بیسیک توضیح دهد.
- ۲- رویه‌های رویداد را تعریف کند.
- ۳- رویه‌های فرعی و رویه‌های تابعی را تعریف کند و تفاوت بین آن‌ها را توضیح دهد.
- ۴- بتواند با استفاده از دستورات Exit Sub و Exit Function از رویه‌های فرعی و تابعی خارج شود.
- ۵- انواع روش‌های ارسال متغیرها به رویه‌ها و تفاوت بین آن‌ها را بیان کند.
- ۶- تفاوت رویه‌های عمومی و محلی را بیان کند و بتواند انواع رویه‌ها را به صورت عمومی و محلی تعریف کند.

کلیات

معمولاً در طراحی پروژه‌های واقعی نیاز به دستورالعمل‌های فراوانی خواهید داشت و گاهی پیش می‌آید که مجموعه‌ای از دستورالعمل‌ها در بخش‌های مختلفی از پروژه یا حتی سایر پروژه‌ها به صورت تکراری استفاده می‌شوند. در این صورت می‌توان به جای تکرار، این مجموعه دستورات را یکبار در بخشی از پروژه نوشته و بارها مورد استفاده قرار داد. به این منظور در پروژه‌های بزرگ، برنامه به بخش‌های کوچک‌تر تقسیم می‌شود و در نتیجه زمان لازم برای خطایابی و خطازدایی و نوشتن دستورالعمل‌ها به شکل قابل توجهی کاهش می‌یابد. به علاوه ایجاد تغییرات در برنامه برای ارتقا و توسعه آن در آینده راحت‌تر و با خطای کمتر توأم خواهد بود. به این روش، برنامه‌نویسی ساخت یافته (Structural) می‌گویند. همان‌طور که می‌دانید این ویژگی در زبان ویژوال بیسیک نیز وجود دارد و با استفاده از انواع رویه‌ها (Procedure) می‌توانید به روش ساخت یافته نیز برنامه‌نویسی کنید. در ویژوال بیسیک، چهار نوع رویه وجود دارد که عبارتند از: رویه‌های فرعی (Sub Procedure)، رویه‌های تابعی (Function Procedure)، رویه‌های رویداد (Event Procedure) و رویه‌های آماده ویژوال بیسیک (Visual Basic Procedure). در این جا به توضیح سه گروه اول خواهیم پرداخت و در واحدکار بعد نحوه استفاده از رویه‌های آماده ویژوال بیسیک را فراخواهید گرفت.

۷-۱ رویه‌های فرعی (Sub Procedure)

گاهی اوقات لازم است تا برنامه‌نویس دستورات مورد نظر خود را به صورت یک بخش مستقل از سایر دستورات در پروژه قرار دهد و در هر زمان که لازم باشد در هر جایی از پروژه (مانند سایر رویه‌ها) از آن استفاده کند. در این صورت استفاده از رویه‌های فرعی می‌تواند یکی از انتخاب‌های مناسب برای وی باشد.

به عنوان مثال رویه‌های رویداد از نوع رویه‌های فرعی هستند که بارها از آن استفاده کرده‌اید. رویه‌های فرعی دیگر را نیز می‌توانید در ماژول فرم، تعریف و استفاده کنید. اما همان‌طور که می‌دانید رویه‌های رویداد به‌طور خودکار در زمان لازم اجرا می‌شوند، اما برای اجرای رویه‌های فرعی که برنامه‌نویس تعریف می‌کند باید از دستورات خاصی استفاده شود تا بتواند رویه را در زمان مناسب اجرا کند.

در این جا لازم است تا نحوه تعریف یک رویه را به طور کلی بیاموزید، برای تعریف یک رویه فرعی به این شکل عمل کنید:

(... , نوع آرگومان As نام آرگومان اول) نام رویه فرعی Sub Private

⋮

دستور(ات)

End Sub

تعریف یک رویه فرعی با کلمه کلیدی Private آغاز می‌شود. استفاده از این عبارت در تعریف یک رویه فرعی سبب می‌شود تا رویه مربوطه به عنوان یک رویه محلی تعریف شود؛ بنابراین فقط در سایر رویه‌های همان ماژول فرمی که در آن تعریف شده، قابل شناسایی و فراخوانی است. در ادامه تعریف رویه، کلمه کلیدی Sub و به دنبال آن نام رویه فرعی قرار می‌گیرد. برای نام‌گذاری رویه‌ها می‌توانید مانند متغیرها عمل کنید. پس از آن‌که نام رویه را تعیین کردید باید نام و نوع آرگومان‌های رویه را مشخص کنید. در واقع آرگومان‌ها متغیرهایی هستند که به برنامه‌نویس اجازه می‌دهند تا از این طریق مقادیر موردنظر را به رویه فرعی ارسال کنند. قوانین تعریف و نام‌گذاری آرگومان‌ها دقیقاً مشابه متغیرهاست.

البته استفاده از آرگومان‌ها اجباری نبوده و یک رویه فرعی می‌تواند یک، دو یا چند آرگومان داشته باشد یا این‌که اصلاً هیچ آرگومانی نداشته باشد. اما در صورت استفاده از آن‌ها باید در هنگام فراخوانی رویه، دقت لازم را برای انتقال مقادیر مورد نظر به آرگومان‌ها به عمل آورید.

پس از خط اول تعریف رویه، دستورات مورد نظر که در رویه اجرا می‌شوند، قرار داده شده و خاتمه رویه نیز با عبارت End Sub تعیین می‌شود. بدین صورت رویه شما در ماژول فرم تعریف می‌شود.



مثال ۱: پروژه‌ای طراحی کنید که طول و عرض یک مستطیل را از کاربر دریافت کند و با استفاده از یک رویه فرعی، محیط و مساحت آن را محاسبه کرده و نمایش دهد. برای انجام این کار مراحل بعد را انجام دهید:

۱ - برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۱-۷ ایجاد کنید.



شکل ۱-۷

می بینید که دو کادر متن برای دریافت طول و عرض مستطیل در نظر گرفته شده است و پس از ورود داده‌ها با کلیک روی دکمه Calculate محیط و مساحت مستطیل محاسبه خواهد شد. ۲- برای محاسبه محیط و مساحت مستطیل با استفاده از یک رویه فرعی، ابتدا باید آن را تعریف کنید بنابراین به پنجره ماژول فرم رفته و تعریف رویه را به این صورت تایپ کنید، سپس کلید Enter را بفشارید.

Private Sub rectangle()

عبارت End Sub در پایین دستوری که تایپ کرده‌اید به طور خودکار ظاهر می‌شود. در واقع این دو دستور محدوده رویه فرعی را تعیین می‌کنند و دستورات مربوط به رویه در بین این دو خط قرار می‌گیرند. رویه‌های فرعی نیز مانند رویه‌های رویداد دارای یک نام هستند. نام‌گذاری رویه‌های فرعی مانند نام‌گذاری متغیرهاست و از همان قوانین پیروی می‌کنند. ۳- در این مرحله باید دستورات مربوط به محاسبه محیط و مساحت مستطیل را در رویه فرعی rectangle تایپ کنید؛ بنابراین رویه را به صورت بعد تنظیم کنید:

Private Sub rectangle()

Dim sngl As Single, sngw As Single

Dim sngarea As Single, sngperi As Single

sngl = Val(txtl.Text)

sngw = Val(txtw.Text)

sngarea = sngl * sngw

sngperi = 2 * (sngl + sngw)

lblarea.Caption = "Area Is " + Str(sngarea)

lblperi.Caption = "Perimeter Is " + Str(sngperi)

End Sub

در این رویه ابتدا داده‌های ورودی در کادرهای متن به عدد تبدیل می‌شوند و در متغیرهای sngl و sngw ذخیره می‌شوند، سپس عملیات محاسبه مساحت و محیط انجام و در نهایت با استفاده از کنترل‌های برچسب نمایش داده می‌شوند.
پس از تعریف یک رویه فرعی لازم است نحوه استفاده از دستورات موجود در رویه یا به عبارت بهتر اجرای رویه فرعی را بدانید به این عمل فراخوانی رویه نیز می‌گویند. برای انجام این کار می‌توانید با توجه به نیاز خود یکی از روش‌های زیر را به کار گیرید:

(... , مقدار مربوط به آرگومان اول) نام رویه Call

... , مقدار مربوط به آرگومان اول نام رویه

فراخوانی می‌تواند با دستور Call یا بدون آن انجام شود سپس نام رویه و در صورت استفاده از آرگومان‌ها، مقادیر یا متغیرهای متناسب با هر یک را در داخل پرانتز تایپ کرده و با کاراکتر کاما (,) از یکدیگر جدا کنید.

۴ - برای آن‌که بتوانید دستورات موجود در رویه rectangle را اجرا کنید باید در رویداد Click دکمه Calculate، این دستور را تایپ کنید: Call rectangle

دستور Call در هنگام اجرا، رویه rectangle را فراخوانی می‌کند، به عبارت دیگر اجرای برنامه به رویه rectangle منتقل می‌شود و پس از رسیدن به عبارت End Sub در این رویه اجرای برنامه مجدداً به محل فراخوانی باز می‌گردد و دستور بعد از Call اجرا می‌شود.

۵ - پروژه و فرم را با نام rectangle ذخیره نمایید.

۶ - پروژه را اجرا نمایید. اعداد ۴ و ۶ را به عنوان طول و عرض مستطیل وارد کنید و روی دکمه Calculate کلیک کنید و نتیجه محاسبه را بررسی نمایید.

۷ - پروژه را برای مقادیر دیگر نیز بررسی کنید، سپس به پنجره ویژوال بیسیک بازگردید.



تمرین:

پروژه‌ای طراحی کنید که با استفاده از یک رویه فرعی، محیط و مساحت هر دایره به شعاع دلخواه R را محاسبه کند.



مثال ۲: پروژه‌ای طراحی کنید که عدد دلخواهی را از کاربر دریافت کرده و زوج یا فرد بودن آن را با استفاده از یک رویه فرعی مشخص کند. برای انجام این کار مراحل بعد را به ترتیب انجام دهید:

- ۱ - یک پروژه از نوع Standard EXE و یک فرم مطابق شکل ۲-۷ ایجاد کنید.
- ۲ - روی منوی Tools در پنجره ویژوال بیسیک کلیک کنید و سپس گزینه Add Procedure... را برگزینید تا کادر محاوره Add Procedure نمایش داده شود.
- ۳ - در کادر متن Name نام رویه را تایپ کنید. برای رویه این مثال، از عبارت oddoreven استفاده کنید (شکل ۳-۷).
- ۴ - در بخش Type گزینه Sub را برای ایجاد رویه فرعی انتخاب کنید.
- ۵ - در بخش سوم از تنظیمات کادر محاوره Add Procedure برای میدان دید و شناسایی رویه (Scope) دکمه انتخاب Private را برگزینید.



شکل ۲-۷



شکل ۳-۷

۶ - در پایان روی دکمه OK در کادر محاوره Add Procedure کلیک کنید تا رویه فرعی در ماژول فرم اضافه شود.

۷ - اکنون به پنجره ماژول فرم بروید و رویه oddoreven را به صورت زیر تنظیم کنید:

Private Sub oddoreven(sngno As Single)

If (sngno Mod 2) = 0 Then

 MsgBox "your number is even .» , , «Message»

Else

 MsgBox "your number is odd .» , , «Message»

End If

End Sub

رویه فرعی oddoreven دارای یک آرگومان با نام sngno می‌باشد که از نوع Single تعریف شده است، از این آرگومان برای انتقال عددی که می‌خواهید زوج یا فرد بودن آن را داخل رویه معین کنید، استفاده می‌شود. داخل رویه نیز با استفاده از عملگر Mod و یک If باقی‌مانده تقسیم عدد مورد نظر که در آرگومان sngno ذخیره شده است بر عدد دو بررسی می‌شود. اگر این مقدار برابر صفر باشد کادر پیام، عدد را به عنوان یک عدد زوج و در غیر این صورت عدد را به عنوان یک عدد فرد معرفی خواهد کرد.

۸- در این مرحله باید رویداد Click دکمه فرمان Compute به شکلی تنظیم شود تا در صورت کلیک روی دکمه Compute رویه فرعی oddoreven فراخوانی شده و زوج یا فرد بودن عدد مربوطه را معین کند، بنابراین رویه رویداد Click دکمه Compute را به این صورت تنظیم کنید:

```
Private Sub cmdcom_Click()
```


```
Dim snginput As Single
```

```
snginput = Val(txtno.Text)
```

```
Call oddoreven(snginput)
```

```
End Sub
```

در این رویداد پس از تعریف متغیر snginput، با استفاده از تابع Val عبارت تایپ شده در کادر متن به مقدار عددی تبدیل و در متغیر snginput ذخیره می‌شود. در مرحله بعد باید رویه فرعی oddoreven را فراخوانی کرد. برای این کار با استفاده از دستور Call رویه فراخوانی می‌شود و برای آن که عدد مورد نظر کاربر که در متغیر snginput ذخیره شده است در اختیار رویه قرار گیرد، نام متغیر snginput پس از نام رویه و در داخل پرانتز ذکر می‌شود، در این صورت در هنگام اجرای دستور Call بین متغیر snginput و متغیر sngno ارتباط برقرار می‌شود و در واقع هنگام اجرای رویه، مراجعه به متغیر sngno به منزله مراجعه، خواندن و ذخیره‌سازی اطلاعات در متغیر snginput است.

 **نکته** در صورتی که یک رویه دارای آرگومان باشد، اگر بعد از تعریف رویه، نام رویه را در ماژول فرم تایپ کنید؛ ویژوال بیسیک یک راهنمایی به صورت ToolTip را که شامل نام رویه و نام و نوع آرگومان‌های آن می‌باشد، نمایش می‌دهد (شکل ۴-۷).



شکل ۴-۷

- ۹ - پروژه و فرم را با نام oddeven ذخیره کرده، سپس آن را اجرا کنید.
- ۱۰ - در کادر متن فرم برنامه، عدد ۴ را تایپ کنید و روی دکمه Compute کلیک کنید. در این صورت رویه فراخوانی می‌شود و عدد ۴ که در متغیر snginput ذخیره شده است، در آرگومان sngno کپی می‌شود و سپس نوع عدد تعیین شده و در کادر پیام، پیغام «عدد زوج است» نمایش داده می‌شود.
- ۱۱ - پروژه را برای چند عدد دیگر بررسی کنید و به محیط برنامه‌نویسی ویژوال بیسیک باز گردید.

۲-۷ رویه‌های تابعی (Function Procedure)

نوع دیگری از رویه‌ها، رویه‌های تابعی هستند. رویه‌های تابعی شبیه به رویه‌های فرعی هستند با این تفاوت که از آن‌ها در مواردی که لازم باشد، استفاده می‌شود و پس از انجام مجموعه‌ای از محاسبات، مقداری به محل فراخوانی رویه، بازگشت داده می‌شود. قبلاً از بعضی رویه‌های تابعی آماده در ویژوال بیسیک استفاده کرده‌اید، برای مثال می‌توان به رویه تابعی Val، Str و نظایر آن‌ها اشاره کرد. تابع Val با دریافت یک رشته از کاراکترهای رقمی آن را به مقدار عددی تبدیل می‌کند و تابع Str با دریافت یک مقدار عددی آن را به رشته تبدیل می‌نماید. در ویژوال بیسیک امکان تعریف رویه‌های تابعی مورد نیاز توسط برنامه‌نویس نیز وجود دارد.

تعریف و فراخوانی رویه‌های تابعی نیز شبیه به رویه‌های فرعی است. برای تعریف رویه‌های تابعی می‌توانید رویه را در مازول فرم تایپ کنید یا از کادر محاوره Add Proc - dure استفاده کنید. شکل کلی تعریف رویه‌های تابعی به این صورت است:

نوع مقدار بازگشتی As (. . . , نوع آرگومان As نام آرگومان) نام تابع Private Function

.....
دستور(ات)
مقدار بازگشتی = نام تابع
.....

End Function

تعریف یک رویه تابعی مانند رویه فرعی بوده و با کلمه کلیدی Private آغاز شده و باعث می‌شود رویه تابعی به صورت محلی تعریف شود. در ادامه تعریف رویه تابعی کلمه کلیدی Function و به دنبال آن نام رویه تابعی قرار می‌گیرد. قوانین نام‌گذاری رویه‌های تابعی مانند رویه‌های فرعی و متغیرهاست.

پس از تعیین نام رویه تابعی باید نام و نوع آرگومان‌های رویه را مشخص کنید تا با استفاده از آن‌ها مقادیر مورد نیاز به رویه ارسال شوند. نحوه تعریف و استفاده از آرگومان‌ها در رویه‌های تابعی مانند رویه‌های فرعی می‌باشد و استفاده از آن‌ها و تعداد و نوع آرگومان‌ها اختیاری است. پس از تعریف رویه تابعی باید دستورات مورد نظر خود را در رویه قرار داده و خاتمه رویه را با عبارت End Function تعیین کنید.

اما نکته مهمی که در رابطه با رویه‌های تابعی وجود دارد و در رویه‌های فرعی موجود نیست، مربوط به نحوه بازگشت یک مقدار به محل فراخوانی می‌شود. همان‌طور که در شکل کلی تعریف رویه تابعی می‌بینید، باید مقداری را که می‌خواهید بازگشت دهید به نام تابع انتساب دهید. با رسیدن به این دستور و اجرای آن مقدار تعیین شده به محل فراخوانی بازگشت داده می‌شود. علاوه بر این باید نوع مقدار بازگشتی در انتهای خط اول تعریف رویه و پس از معرفی آرگومان‌ها معین شود. برای فراخوانی یک رویه تابعی می‌توانید به این شکل عمل کنید:

... , مقدار مربوط به آرگومان اول) نام رویه تابعی = نام یک متغیر متناسب با نوع داده بازگشتی

هم‌چنین می‌توانید رویه تابعی را بدون آن‌که مقداری را به عنوان نتیجه محاسبات بازگشت دهد، تعریف کنید. در این صورت فراخوانی رویه تابعی می‌تواند به صورت فراخوانی یک رویه فرعی باشد.

(... , مقدار مربوط به آرگومان اول) نام رویه Call

... , مقدار مربوط به آرگومان اول نام رویه

نکته: مقدار بازگشتی یک تابع می تواند به عنوان مقدار ارسالی برای تابع دیگری استفاده شود. نوع مقدار بازگشتی در تعریف تابع با توجه به نوع داده ای که به محل فراخوانی بازگشت می دهد تعیین می شود. اگر نوع مقدار بازگشتی در تعریف تابع تعیین نشود به طور پیش فرض از نوع داده Integer استفاده می شود.

مثال ۳: پروژه ای طراحی کنید که با استفاده از یک تابع، باقیمانده تقسیم دو عدد دلخواه را محاسبه کرده و نمایش دهد. برای این کار مراحل بعد را انجام دهید:



شکل ۵-۷

۱ - برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۵-۷ ایجاد کنید. دو کادر متن روی فرم قرار دارند که اعداد مورد نظر در آنها وارد خواهند شد و کاربر با کلیک روی دکمه Remainder باقی مانده تقسیم دو عدد را مشاهده خواهد کرد.

۲ - به پنجره ماژول فرم بروید و تابع remainder را به این صورت برای محاسبه باقیمانده تقسیم دو عدد تعریف کنید:

```
Private Function remainder (intno1 As Integer, intno2 As _ Integer)As Integer
```

```
Dim intresult As Integer
```

```
intresult = intno1 Mod intno2
```

```
remainder = intresult
```

```
End Function
```

تابع remainder دارای دو آرگومان $intno1$ و $intno2$ است که برای دریافت مقدار مقسوم و مقسوم علیه به کار می‌روند؛ سپس در بخش دستورات این رویه ابتدا یک متغیر محلی دیگر برای ذخیره‌سازی باقی‌مانده تقسیم، تعریف می‌شود و به دنبال آن با استفاده از عملگر Mod باقی‌مانده تقسیم دو عدد در این متغیر (intresult) ذخیره می‌شود. اما برای آن‌که نتیجه محاسبه به محل فراخوانی بازگشت داده شود از دستور $remainder = intresult$ استفاده می‌شود و نتیجه محاسبه که در متغیر intresult ذخیره شده است به نام تابع نسبت داده می‌شود. به علاوه چون متغیر intresult از نوع عدد صحیح (Integer) معرفی شده است، بنابراین در انتهای خط اول تعریف تابع نوع مقدار بازگشتی نیز از نوع Integer معرفی شده است.

۳- در این مرحله رویداد Click دکمه فرمان Remainder را به صورت زیر تنظیم کنید:

```
Private Sub cmdrem_Click()
```

```
Dim intm As Integer, intn As Integer
```

```
Dim intmod As Integer
```

```
intm = Val(txtno1.Text)
```

```
intn = Val(txtno2.Text)
```

```
intmod = remainder(intm, intn)
```

```
lblres.Caption = "Answer Is " + Str(intmod)
```

```
End Sub
```

با اجرای این رویه ابتدا مقادیری که در کادرهای متن تایپ شده‌اند، بعد از تبدیل در متغیرهای intm و intn ذخیره می‌شوند؛ سپس تابع remainder با متغیرهای intm و intn فراخوانی می‌شود و در نتیجه اجرای برنامه به بخش تعریف رویه تابعی remainder منتقل شده و بین متغیرهای ارسالی intm و intn با آرگومان‌های $intno1$ و $intno2$ ارتباط برقرار می‌کند، سپس اجرای دستورات موجود در رویه تابعی آغاز می‌شود و با انجام محاسبات و رسیدن به دستور $remainder = intresult$ و انجام آن، اجرای برنامه به محل فراخوانی یعنی رویداد Click دکمه remainder (intmod = remainder(intm, intn)) بازمی‌گردد و مقدار موجود در متغیر intresult در متغیر intmod ذخیره می‌شود و در پایان این مقدار در کنترل برچسب نمایش داده شده و رویداد خاتمه می‌یابد. نوع متغیر intmod با نوع مقدار بازگشتی در تعریف تابع و همین‌طور نوع متغیر intresult هماهنگی لازم را دارد.

۴ - پروژه و فرم را با نام remainder ذخیره کنید، سپس آن را اجرا نمایید.

۵ - اعداد ۱۲ و ۵ را به ترتیب در کادرهای متن وارد کنید، سپس روی دکمه Remainder کلیک کرده و نتیجه محاسبه را مشاهده نمایید.

۶ - پروژه را برای مقادیر دیگری نیز آزمایش کنید، سپس به پنجره ویژوال بیسیک بازگردید.



تمرین:

پروژه فوق را به گونه‌ای تنظیم کنید که اگر عدد اول از عدد دوم کوچک‌تر باشد با استفاده از یک رویه فرعی، خطای کاربر به وی گزارش شود و محاسبات نیز انجام نشود. پروژه‌ای طراحی کنید که یک عدد دلخواه را از کاربر دریافت کند و با استفاده از یک رویه تابعی قدر مطلق آن را محاسبه کرده و نمایش دهد. پروژه‌ای طراحی کنید که با استفاده از یک تابع، قرینه هر عدد دلخواه را نمایش دهد. رویه‌ای تابعی بنویسید که هر عدد دلخواه x را دریافت کند و مقدار معکوس آن را $\frac{1}{x}$ محاسبه کرده و بازگشت دهد.

۳-۷ روش‌های ارسال مقادیر به رویه‌های فرعی و تابعی

تاکنون نحوه استفاده از رویه‌های فرعی و تابعی و چگونگی انتقال مقادیر مورد نیاز به رویه‌ها را فرا گرفتید. در واقع دو روش مختلف برای ارسال مقادیر مورد نیاز به رویه‌ها وجود دارد که هر یک ویژگی‌هایی را به همراه دارند. اولین روش فراخوانی متغیرها با مرجع (Call By Reference) و دومین روش فراخوانی متغیرها با مقدار (Call By Value) است.

۱-۳-۷ روش فراخوانی با مرجع

در این روش، ارتباط مستقیمی بین متغیر ارسالی و آرگومان متناظر آن در رویه مربوطه برقرار می‌شود به گونه‌ای که متغیر و آرگومان متناظر با آن به محل مشترکی در حافظه اشاره می‌کنند. بنابراین هرگونه تغییر در محتویات آرگومان‌ها در رویه، سبب تغییر در متغیرهای ارسالی خواهد شد. در واقع شما تاکنون با این روش، مقادیر خود را به رویه‌ها

ارسال کرده‌اید.

در صورتی که بخواهید متغیرها را با روش ارسال با مرجع به یک رویه ارسال نمایید، کلمه کلیدی ByRef را در ابتدای نام آرگومان‌ها در تعریف رویه ذکر کنید. البته در صورت عدم استفاده از کلمه کلیدی ByRef نیز نحوه ارسال متغیرها به طور پیش فرض از نوع فراخوانی با مرجع می‌باشد.

فرض کنید می‌خواهید با استفاده از یک رویه فرعی محتویات دو متغیر را با یکدیگر جابه‌جا کنید، برای این منظور رویه زیر را در نظر بگیرید:

```
Private Sub moving(ByRef sngx As Single,ByRef sngy As Single)
```

```
Dim sngtemp As Single
```

```
sngtemp = sngy
```

```
sngy = sngx
```

```
sngx = sngtemp
```

```
End Sub
```

رویه فرعی moving دو آرگومان دارد که برای دریافت مقادیر دو متغیر به کار می‌روند و در ابتدای نام هر یک از آن‌ها از کلمه کلیدی ByRef استفاده شده است، این عبارت تعیین می‌کند که نحوه ارسال متغیرها به رویه به صورت فراخوانی با مرجع باشد.

هم‌چنین با تعریف یک متغیر واسطه با نام sngtemp در داخل رویه فرعی، ابتدا مقادیر یکی از متغیرها را در متغیر واسطه (sngtemp) ذخیره کرده، سپس مقدار متغیر sngx را در متغیر sngy و در آخر مقدار متغیر واسطه را در متغیر sngx کپی می‌کند. به این صورت مقادیری که در آرگومان‌های sngx و sngy قرار دارند با یکدیگر تعویض می‌شوند؛ اما باید ببینیم که چگونه از این رویه می‌توان برای تعویض مقدار دو متغیر در خارج از رویه استفاده کرد. فرض کنید که برای این کار رویه فرعی moving به صورت زیر فراخوانی می‌شود:

```
Dim x As Single
```

```
Dim y As Single
```

```
x = 28
```

```
y = -10
```

```
Call moving(x, y)
```

```
Print x, y
```

پس از تعریف و مقداردهی دو متغیر x و y ، رویه فرعی فراخوانی شده است. در این جا اجرای برنامه به محل رویه منتقل می شود و چون از کلمه کلیدی ByRef در ابتدای نام آرگومانها در تعریف رویه استفاده شده است، ارتباط مستقیمی بین متغیرهای ارسالی و آرگومانهای رویه moving به وجود می آید؛ در نتیجه تغییر در محتویات آرگومانهای $sngx$ و $sngy$ سبب تغییر در محتویات متغیرهای ارسالی x و y و تعویض محتویات دو متغیر می شوند. بنابراین پس از خاتمه اجرای رویه، دستور Print مقادیر جابه جا شده را برای متغیرهای x و y در روی فرم نمایش خواهد داد.

نکته اگر رویه فرعی moving به صورت $Call\ moving(x+1, y+1)$ فراخوانی شود مقدار متغیرها پس از فراخوانی رویه همان مقادیر قبل از فراخوانی خواهد بود.
اگر مقادیر ثابت به رویه فرعی moving ارسال شود کلمه های کلیدی ByRef هیچ تأثیری روی نحوه ارسال آنها نمی گذارد.
در روش فراخوانی با مرجع، هم نام یا غیرهم نام بودن متغیرهای ارسالی با آرگومانهای رویه اهمیتی ندارد.

تمرین:



پروژه ای طراحی کنید تا بتوانید با استفاده از یک فرم، یک کنترل دکمه فرمان و با استفاده از رویه فرعی moving تعویض محتویات دو متغیر را با استفاده از روش فراخوانی با مرجع بررسی کنید.

۲-۳-۷ روش فراخوانی با مقدار

گاهی اوقات لازم است تا بین متغیرهای ارسالی و آرگومانهای متناظر آنها ارتباطی به وجود نیاید و در صورت تغییر در مقدار آرگومانها، در محتویات متغیرهای ارسالی تغییر حاصل نشود. در این صورت امکان استفاده از روش فراخوانی با مرجع مفید نیست. روش دیگری که برای انجام این گونه عملیات مناسب است، روش فراخوانی با مقدار می باشد. در این روش با ذکر کلمه کلیدی ByVal به جای عبارت ByRef از ایجاد ارتباط مستقیم بین متغیرهای ارسالی و آرگومانها جلوگیری به عمل می آید و فقط مقادیر موجود در متغیرهای ارسالی در داخل آرگومانهای متناظر آنها کپی خواهد شد. به عنوان

مثال اگر تعریف رویه فرعی moving را به این صورت تغییر دهید:

```
Private Sub moving(ByVal sngx As Single, ByVal sngy As Single)
```

در صورت فراخوانی رویه فرعی moving با حالت فوق، در واقع عدد ۲۸ در آرگومان sngx رویه و عدد ۱۰ - در آرگومان sngy رویه کپی خواهند شد و پس از تعویض محتویات این دو آرگومان در داخل رویه و خروج از رویه، مقادیر مربوط به این آرگومان‌ها از بین خواهد رفت و با بازگشت به محل فراخوانی و اجرای دستور Print همان مقادیر قبل از فراخوانی برای متغیرهای x و y نمایش داده می‌شوند.



تمرین:

رویه moving را در پروژه‌ای که در تمرین قبل ایجاد کرده‌اید به شکل فراخوانی با مقدار تغییر دهید سپس آن را اجرا کرده و نتیجه را با حالت قبل بررسی کنید.



مثال ۴: یک رویه فرعی بنویسید تا دو عدد را دریافت کرده، میانگین آن‌ها را محاسبه کند و نمایش دهد. برای این کار یک رویه average را به صورت زیر بنویسید:

```
Private Sub average(ByVal sngno1 As Single, ByVal sngno2 _
```

```
As Single, ByVal sngave As Single)
```

```
sngave = (sngno1 + sngno2) / 2
```

```
End Sub
```

در این رویه از سه آرگومان استفاده شده است. آرگومان‌های sngno_۱ و sngno_۲ برای ارسال اعداد به داخل رویه و به صورت فراخوانی با مقدار تعریف شده‌اند، زیرا لازم نیست این دو آرگومان در متغیرهایی که در زمان فراخوانی به رویه ارسال می‌شوند، تغییری ایجاد کنند. اما آرگومان سوم یعنی sngave با روش ارسال با مرجع تعریف شده است؛ زیرا پس از محاسبه معدل دو عدد در محل فراخوانی مورد استفاده قرار می‌گیرد. برای چنین رویه‌ای، فراخوانی به صورت زیر خواهد بود:

```
Call average (x , y , average)
```

در فراخوانی فوق ابتدا مقادیر متغیرهای ارسالی x و y در آرگومان‌های sngno_۱ و sngno_۲

کپی می‌شوند و بین متغیر average و آرگومان sngave در رویه به گونه‌ای ارتباط

برقرار می‌شود که تغییر در مقدار آرگومان sngave، محتویات متغیر ارسالی average را نیز تحت تأثیر قرار می‌دهد و سبب اعمال همان تغییرات روی آن خواهد شد. می‌بینید که با این روش نیز می‌توان با استفاده از رویه‌های فرعی نتیجه محاسبات را به محل فراخوانی بازگشت داد. به این صورت پس از محاسبه متوسط دو عدد و ذخیره این مقدار در متغیر sngave رویه خاتمه یافته و به محل فراخوانی باز می‌گردد. اکنون می‌توانید در محل فراخوانی رویه با استفاده از متغیر average به معدل دو عدد دسترسی پیدا کنید.



تمرین:

یک پروژه از نوع Standard EXE به همراه یک فرم به گونه‌ای طراحی کنید تا با استفاده از رویه average بتوان معدل هر دو عدد دلخواه را محاسبه و روی فرم مشاهده کرد.

۴-۷ نحوه استفاده از نام آرگومان‌ها در رویه‌ها

تاکنون با استفاده از رعایت ترتیب در ارسال آرگومان‌های رویه‌ها توانستیم داده‌ها را به آرگومان‌های متناظرشان نسبت دهیم، شما می‌توانید با استفاده از نام آرگومان‌ها در زمان فراخوانی، ترتیب ذکر داده‌ها را بدون در نظر گرفتن ترتیب قرارگیری آن‌ها در تعریف رویه، به رویه ارسال کنید.

این روش باعث می‌شود در زمان زیاد بودن تعداد آرگومان‌ها کار فراخوانی آسان‌تر و احتمال اشتباه در ارسال داده‌ها به آرگومان‌ها کمتر شود. برای استفاده از نام آرگومان‌ها در زمان فراخوانی می‌توانید به صورت زیر عمل کنید: مقدار ارسالی = نام آرگومان
فرض کنید می‌خواهید یک رویه تابعی بنویسید که به وسیله آن بتوان مجموع اعداد زوج بین دو عدد طبیعی دلخواه را محاسبه کرد. برای این کار به این دستورات توجه کنید.

$$(m < n)$$

Function Myfunc(m As Integer, n As Integer) As Single

Dim i As Integer, sum As Single

If m Mod 2 = 0 Then m = m+2 Else m = m+ 1

sum= 0

```
For i = m To n-1 Step 2
```

```
sum=sum+i
```

```
Next i
```

```
Myfunc = sum
```

```
End Function
```

در تعریف این تابع دیده می‌شود دو آرگومان m و n (با فرض $m < n$) به تابع ارسال می‌شوند. در تابع ابتدا زوج یا فرد بودن m به وسیله یک `if` بررسی می‌شود تا بتوان عدد زوج بعد از m را تعیین کرد اگر m زوج باشد حاصل $M \bmod 2 = 0$ درست خواهد بود و با اضافه شدن دو واحد به m عدد زوج بعدی محاسبه می‌شود اما اگر m فرد باشد نتیجه $M \bmod 2 = 0$ نادرست خواهد بود و با اضافه شدن یک واحد به m باز عدد زوج بعدی محاسبه می‌شود، سپس مجموع اعداد زوج بین m و n با یک حلقه `For` و توسط دستور $Sum = Sum + i$ به دست می‌آید و در پایان مقدار `Sum` به عنوان مجموع اعداد زوج بین m و n بازگشت داده می‌شود. حلقه `For` نیز تا رسیدن به مقدار $n - 1$ اجرا شده تا در صورت زوج بودن n خود n در مجموع وارد نشود.

بعد از درک نحوه کارکرد تابع به فراخوانی آن می‌پردازیم. تاکنون نحوه فراخوانی چنین توابعی را فرا گرفته‌اید، مثلاً برای نمایش مجموع اعداد زوج بین ۲ و ۱۲ از فرمان بعد استفاده می‌کنیم:

```
Print Myfunc (2,12)
```

اگر در فراخوانی جای اعداد ۲ و ۱۲ عوض شود نتیجه کار نامناسب خواهد بود. اما با فراخوانی با استفاده از نام آرگومان‌ها امکان اشتباه از بین می‌رود. برای نمونه در مثال فوق فراخوانی با استفاده از نام آرگومان‌ها به صورت زیر خواهد بود:

```
Print Myfunc (n := 12 , m := 2)
```

می‌بینید که برای فراخوانی با استفاده از نام آرگومان‌ها باید از نام آرگومان به همراه «:=» در قبل از مقدار ارسالی استفاده کنید. البته فراخوانی قبل را می‌توان به صورت زیر نیز نوشت:

```
Print Myfunc (m := 2 , n := 12)
```



تمرین:

رویه تابعی `Myfunc` را به گونه‌ای تنظیم کنید که بدون فرض $M < N$ نیز بتواند

محاسبات را به‌طور صحیح انجام دهد.

۵-۷ خروج از یک رویه با استفاده از دستورات Exit Sub و Exit Function

گاهی اوقات لازم است تا قبل از این که تمام دستورات یک رویه اجرا شده و به انتهای رویه برسید، از رویه خارج شده و به محل فراخوانی بازگردید. به این منظور برای خروج از یک رویه فرعی از دستور Exit Sub و برای خروج از یک رویه تابعی از دستور Exit Function استفاده کنید.



مثال ۵: یک رویه تابعی تعریف کنید تا تعداد ارقام هر عدد طبیعی دلخواه را محاسبه کرده و نمایش دهد. برای این کار رویه تابعی digits را به صورت زیر تعریف کنید:

```
Private Function digits(ByVal sngno As Integer) As Integer
```

```
    Dim intdigits As Integer
```

```
    If sngno < 0 Then
```

```
        digits = -1
```

```
        Exit Function
```

```
    End If
```

```
    While (sngno > 0)
```

```
        0 ngno = sngno \ 1
```

```
        intdigits = intdigits + 1
```

```
    Wend
```

```
    digits = intdigits
```

```
End Function
```

این تابع دارای یک آرگومان است که برای انتقال عدد مورد نظر به داخل تابع استفاده شده است. علاوه بر این روش ارسال عدد مورد نظر نیز ارسال با مقدار خواهد بود تا تغییر روی مقدار آرگومان sngno در رویه تابعی، تأثیری روی متغیر ارسالی نگذارد. در ضمن نوع مقدار داده بازگشتی یعنی تعداد ارقام نیز از نوع صحیح است؛ بنابراین نوع داده بازگشتی در انتهای خط اول رویه و از نوع Integer تعریف می‌شود.

به هر حال در صورت فراخوانی این تابع و اجرای آن ابتدا مقدار متغیر ارسالی در

آرگومان sngno کپی می‌شود، سپس با اجرای یک دستور If مقدار عدد ارسال شده بررسی می‌شود و در صورتی که مقدار آن کوچک‌تر از صفر باشد، مقدار ۱- را بازگشت می‌دهد. از این مقدار می‌توان در محل فراخوانی برای اطمینان از نحوه ارسال صحیح مقدار به تابع استفاده کرد سپس با استفاده از دستور Exit Function اجرای رویه خاتمه یافته و به محل فراخوانی باز می‌گردید. این عمل سبب خواهد شد تا کاربر از ارسال مقادیر عددی اشتباه مطلع شود، اما اگر مقدار ارسالی بزرگ‌تر از صفر باشد با استفاده از یک حلقه While...Wend تعداد ارقام عدد محاسبه شده و در متغیر intdigits ذخیره می‌شود و پس از خاتمه حلقه این مقدار با استفاده از دستور digits= intdigits بازگشت داده خواهد شد.



تمرین:

پروژه‌ای طراحی کنید تا با استفاده از یک رویه فرعی بتواند تعداد ارقام هر عدد طبیعی دلخواه را محاسبه کرده و در یک کادر پیام نمایش دهد، به علاوه در صورت ارسال مقادیر منفی به رویه، پیام خطایی نمایش داده شود و اجرای رویه نیز خاتمه یابد.

۶-۷ رویه‌های محلی و عمومی

رویه‌هایی که تاکنون تعریف شده‌اند از نوع محلی یا به عبارت دیگر Private می‌باشند و فقط در همان ماژول که تعریف می‌شوند قابل شناسایی و فراخوانی هستند، اما گاهی اوقات لازم است تا یک رویه تابعی یا فرعی را که در یک ماژول فرم تعریف شده است در ماژول فرم دیگری فراخوانی کنید.



شکل ۶-۷

در این صورت در زمان اجرای برنامه و با فراخوانی رویه مزبور پیام خطایی مطابق شکل ۶-۷ نمایش داده می‌شود.

۱-۶-۷ نحوه ایجاد رویه‌های عمومی در ماژول فرم


برای آن که عملیات فراخوانی یک رویه از داخل ماژول فرم دیگری امکان‌پذیر باشد به جای کلمه کلیدی Private، از کلمه کلیدی Public استفاده کنید. به این شکل رویه به


عنوان یک رویه عمومی معرفی خواهد شد و علاوه بر فراخوانی آن در ماژولی که تعریف شده است در سایر ماژول‌ها امکان فراخوانی خواهد داشت، به چنین رویه‌هایی رویه‌های عمومی می‌گویند.

برای فراخوانی رویه‌های عمومی باید قبل از ذکر نام رویه، نام ماژول فرمی را که رویه در آن تعریف شده است، تایپ کنید و سپس کاراکتر نقطه را تایپ کرده و نام رویه را ذکر نمایید. این مطلب را می‌توان به این صورت بیان کرد:

Call (مقادیرارسالی) نام رویه فرعی. نام فرمی که رویه فرعی در آن تعریف شده است

(مقادیرارسالی) نام رویه تابعی. نام فرمی که رویه تابعی در آن تعریف شده است.

 **نکته** در صورتی که حوزه شناسایی (Scope) یک رویه در زمان تعریف، تعیین نشود رویه به طور پیش فرض به صورت عمومی تعریف خواهد شد.

 **مثال ۶:** پروژه‌ای طراحی کنید که میزان مصرف برق هر ساختمان دلخواهی را محاسبه کرده، مقادیر فعلی و قبلی را که از کنتور برق قرائت شده‌اند، وارد کند. با محاسبه میزان مصرف، هزینه برق مصرفی در ساختمان مورد نظر به دست می‌آید. به این منظور مراحل بعد را به ترتیب انجام دهید:

۱ - یک پروژه جدید از نوع Standard EXE ایجاد کنید، سپس یک فرم مطابق شکل ۷-۷ برای دریافت مقادیر ورودی ایجاد کنید.



شکل ۷-۷

در این فرم از چهار کادر متن برای دریافت تاریخ قرائت کنتور و میزان مصرف برق در مراجعه قبلی و مراجعه فعلی استفاده می‌شود و از دکمه Compute برای انجام محاسبات و نمایش نتیجه آن روی فرم دیگری استفاده می‌شود.

۲- برای انجام محاسبات از یک رویه تابعی و یک رویه فرعی استفاده کنید، این رویه‌ها را در ماژول فرم و به این صورت تنظیم کنید:

```
Public Function expense (ByVal sngpv As Single, ByVal sngcv As Single) As Currency
```

```
    Dim sngresult As Single , intlevel As Integer
```

```
    Dim curpay As Currency
```

```
    sngresult = sngcv - sngpv
```

```
    Select Case sngresult
```

```
        Case Is < 500 :
```

```
            intlevel = 1
```

```
        Case Is < 1000 :
```

```
            intlevel = 2
```

```
        Case Is < 1500 :
```

```
            intlevel = 3
```

```
        Case Else
```

```
            intlevel = 4
```

```
    End Select
```

```
    Call payment (intlevel, sngresult, curpay)
```

```
    expense = curpay
```

```
End Function
```

```
Private Sub payment(intlevel As Integer, ByVal sngresult _
```

```
    As Single, ByRef curpay As Currency)
```

```
    Select Case intlevel
```

```
        Case 1 : curpay = sngresult * 100
```

```
        Case 2 : curpay = sngresult * 250
```

```
        Case 3 : curpay = sngresult * 500
```

```
        Case 4 : curpay = sngresult * 1000
```

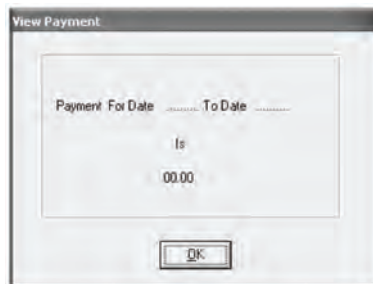
```
    End Select
```

```
End Sub
```

رویه تابعی expense به صورت عمومی تعریف شده و دارای دو آرگومان از نوع Single برای دریافت مقادیر مصرف قبلی و فعلی است. این تابع با توجه به مقدار مصرف برق بر اساس کیلووات، سطح مصرف و هزینه آن را محاسبه کرده و به صورت یک مقدار از نوع Currency بازمی گرداند، پس از فراخوانی این تابع ابتدا مقادیر قرائت شده به ترتیب در آرگومان‌های sngpv و sngcv کپی می‌شوند، سپس تفاضل این دو مقدار از یکدیگر کسر شده و به عنوان میزان مصرف در متغیر sngresult ذخیره می‌شوند و بر همین اساس با استفاده از یک دستور Select Case سطح مصرف (intlevel) نیز تعیین می‌شود تا با توجه به این سطح، هزینه مصرف برق محاسبه شود. به این منظور چهار سطح مصرف از کم تا خیلی زیاد در نظر گرفته شده است، پس از محاسبه سطح مصرف برق، نوبت محاسبه هزینه آن می‌شود. این کار با استفاده از یک رویه فرعی انجام می‌شود. رویه payment با دریافت سطح مصرف (intlevel) و مقدار مصرف برق (sngresult)، هزینه آن را با استفاده از یک دستور Select Case بر اساس سطح مصرف محاسبه کرده و در متغیر curpay ذخیره می‌کند. به علاوه متغیر curpay به صورت فراخوانی با مرجع به رویه ارسال می‌شود تا میزان هزینه در محل فراخوانی قابل استفاده باشد، این رویه از نوع محلی تعریف شده است. پس از آن که رویه payment خاتمه یافت، اجرای برنامه به محل فراخوانی آن یعنی رویه تابعی expense بازگشته و با استفاده از دستور $expense = curpay$ مقدار هزینه مصرف برق را به محل فراخوانی بازمی‌گرداند.

۳- نام کادرهای متن مربوط به تاریخ قرائت و میزان مصرف قبلی را به ترتیب روی tx - pd و txtpv و تاریخ قرائت و میزان مصرف فعلی را به ترتیب روی txtcd و txtcv تنظیم کنید.

۴ - برای نمایش میزان هزینه مصرف برق در یک فرم جداگانه، احتیاج به یک فرم جدید خواهید داشت، بنابراین یک فرم جدید مطابق شکل ۸-۷ ایجاد کنید، سپس فرم و پروژه را با نام viewpayment ذخیره کنید.



شکل ۸-۷

سپس رویداد Activate این فرم و رویداد Click دکمه OK را به صورت زیر تنظیم کنید:

```
Private Sub Form_Activate()
```

```
Dim sngpay As Single
```

```
Dim sngpv As Single
```

```
Dim sngcv As Single
```

```
sngpv = Val(frmpower.txtpv.Text)
```

```
sngcv = Val (frmpower.txtcv.Text)
```

```
sngpay = frmpower.expense (sngpv, sngcv)
```

```
lblview.Caption = "Payment For Date " + Trim (frmpower.txtpd.Text) + _
```

```
" To Date "+Trim (frmpower.txtcd.Text)
```

```
lblpay.Caption = sngpay
```

```
End Sub
```

```
Private Sub cmdok_Click()
```

```
Unload Me
```

```
frmpower.SetFocus
```

```
frmpower.txtpd.SetFocus
```

```
End Sub
```

در این فرم از رویداد Activate فرم View Payment استفاده شده است تا پس از نمایش فرم، گزارش محاسبات در روی فرم به نمایش درآید. در این رویه با فراخوانی تابع expense محاسبات انجام می‌شود. همان‌طور که ملاحظه می‌کنید، برای فراخوانی این تابع ابتدا نام فرمی که تابع در ماژول آن تعریف شده، یعنی frmpower ذکر شده است. البته این کار زمانی مؤثر است که رویه به صورت عمومی (Public) تعریف شده باشد در این صورت در زمان تایپ نام فرم (frmpower) و کاراکتر نقطه یک لیست حاوی نام رویه expense که در ماژول frmpower تعریف شده است، نمایش داده خواهد شد (شکل ۹-۷). نکته دیگری که در فراخوانی رویه expense مهم است، نحوه ارسال مقادیر تایپ شده در کادرهای متن است.

برای دسترسی خصوصیات کنترل‌های یک فرم در فرم دیگر می‌توانید ابتدا نام فرم، سپس نام کنترل و بعد نام خصوصیت را ذکر کنید؛ به این صورت مقادیر مصرف برق با استفاده از تابع Val به نوع عددی تبدیل شده و به تابع ارسال می‌شوند. مقدار هزینه مصرف برق با استفاده از تابع expense محاسبه شده و برگردانده می‌شود و در نهایت این مقدار در متغیر sngpay ذخیره می‌شود. در ادامه اجرای رویه Activate تاریخ‌های مربوط به قرائت کنتور به وسیله کنترل برچسب lblview و هزینه مصرف برق به وسیله کنترل برچسب lblpay نمایش داده می‌شوند.



شکل ۹-۷

رویداد Click دکمه OK نیز به‌گونه‌ای تنظیم شده است تا در صورتی که کاربر روی این دکمه کلیک کند، فرم View Payment بسته شده و فوکوس با متد SetFocus به فرم frmpower و سپس کنترل کادر متن txtpd داده شود.

۵ - به ماژول فرم frmpower بروید و دستور frmview.show را در رویداد Click دکمه Compute و دستور Unload Me را در رویداد Click دکمه Exit تایپ کنید.

۶ - فرم و پروژه را با نام power ذخیره کنید. پروژه را اجرا کنید و داده‌ها را مطابق شکل ۱۰-۷ در کادرهای متن تایپ کرده و روی دکمه Compute کلیک کنید، نتیجه محاسبات مطابق شکل ۱۱-۷ در فرم دوم قابل مشاهده خواهد بود.

۷ - با کلیک روی دکمه Exit به اجرای پروژه خاتمه دهید و به پنجره ویژوال بیسیک بازگردید.



شکل ۷-۱۰



شکل ۷-۱۱



تمرین: پروژه power را به گونه‌ای تنظیم کنید که رویه فرعی payment در ماژول فرم View Payment تعریف شود.

۷-۶-۲ نحوه ایجاد رویه‌های محلی و عمومی با استفاده از ماژول کد

روش دیگری نیز در ویژوال بیسیک وجود دارد که در آن به جای تعریف رویه‌های موردنظر در ماژول فرم از ماژول کد (Code Module) استفاده می‌شود. ماژول کد یکی دیگر از اجزای تشکیل دهنده در پروژه‌هاست. ماژول‌های کد فقط از دستورالعمل‌ها تشکیل می‌شوند و شکل ظاهری مانند فرم‌ها ندارند. یکی دیگر از کاربردهای ماژول کد، تعریف متغیرهای عمومی است. نحوه تعریف و فراخوانی رویه‌ها در یک ماژول کد مانند ماژول فرم است.



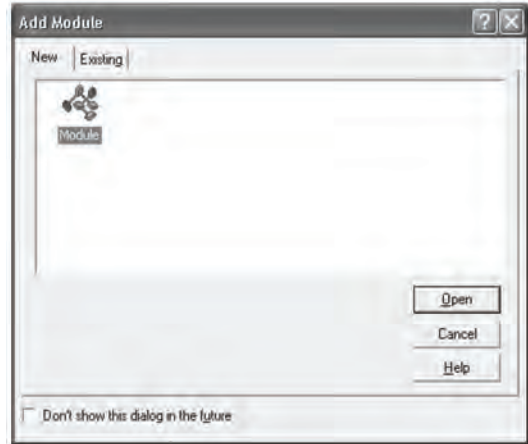
مثال ۷: پروژه power را به گونه‌ای تغییر دهید تا رویه‌های مربوطه را با استفاده از ماژول کد تعریف و فراخوانی کند. به این منظور عملیات زیر را به ترتیب انجام دهید:

- ابتدا یک ماژول کد به پروژه اضافه کنید به این منظور در پنجره ویژوال بیسیک و در پنجره پروژه کلیک راست کنید سپس به ترتیب روی گزینه Add و بعد روی گزینه Module کلیک کنید (شکل ۷-۱۲). کادر محاوره Add Module مطابق شکل ۷-۱۳ نمایش داده می‌شود.

- در کادر محاوره Add Module، روی دکمه Open کلیک کنید تا یک ماژول کد به پروژه power اضافه شود. اکنون به پنجره خصوصیات بروید و خصوصیت Name ماژول کد را روی مقدار mdlpower تنظیم کنید.



شکل ۱۲-۷




شکل ۱۳-۷

۳ - در پنجره پروژه روی آیکن ماژول کد کلیک راست کنید و گزینه Save ModuleAs...

را انتخاب کنید، سپس ماژول کد را با نام power ذخیره کنید. همان طور که در کادر
محواره ذخیره سازی ماژول مشاهده می کنید، پسوند فایل های ماژول کد BAS است.

۴ - در این مرحله به ماژول فرم frmpower بروید و رویه های expense و payment را
به ماژول کد mdlpower انتقال دهید.

۵ - به ماژول فرم frmview بروید و عبارت frmpower را از ابتدای نام رویه expense
حذف کنید. توجه داشته باشید اگر رویه ای در ماژول کد به صورت عمومی تعریف شود،
تمام ماژول ها می توانند آن را فراخوانی کنند و نیازی به استفاده از نام ماژول کد در زمان
فراخوانی رویه نیست.

 **نکته** در صورتی که یک متغیر در یک ماژول کد به صورت عمومی تعریف شود، در تمام رویه
قابل شناسایی و استفاده است

۶ - تغییرات را ذخیره کرده و پروژه را اجرا کنید و با همان مقادیر مثال ۵ آزمایش
کنید.

۷ - در پایان روی دکمه Exit کلیک کنید تا اجرای پروژه خاتمه یابد، سپس به پنجره
ویژوال بیسیک بازگردید.

۷-۷ رویه‌های رویداد (Event Procedures)

تاکنون از این‌گونه رویه‌ها بارها استفاده کرده‌اید و همان‌طور که می‌دانید یکی دیگر از ویژگی‌های زبان برنامه‌نویسی ویژوال بیسیک پشتیبانی از رویدادهاست. این نوع از رویه‌ها در زمان ایجاد یک شیء مانند کنترل یا فرم همراه با آن‌ها به طور هم‌زمان به وجود می‌آیند و زمانی که رویداد مربوط به یک شیء رخ می‌دهد، رویه رویداد و دستورات موجود در آن به طور خودکار اجرا می‌شوند. شکل کلی یک رویه رویداد به این صورت است:

(آرگومان‌های رویداد) نام رویداد - نام شیء Private Sub

⋮

دستورات

End Sub

شیء می‌تواند یک کنترل یا یک فرم باشد. در رویه‌های رویداد کنترل‌ها، نام شیء، مقدار خصوصیت Name کنترل و در رویه‌های رویداد فرم‌ها، از کلمه کلیدی Form به عنوان نام شیء استفاده می‌شود.


در بعضی از رویه‌های رویداد بخش دیگری به نام آرگومان‌های رویداد نیز وجود دارند. آرگومان‌ها در واقع، متغیرهایی هستند که با توجه به نوع رویداد، اطلاعاتی را به داخل رویه منتقل می‌کنند تا برنامه‌نویس با استفاده از آن‌ها مقاصد خود را در رویداد پیگیری کند. رویه‌های رویداد مربوط به ماوس و صفحه کلید از این دسته رویدادها هستند که از آرگومان‌ها استفاده می‌کنند.

البته تاکنون از رویه‌های رویدادی مانند Click, DblClick, Load, Unload و نظایر آن‌ها در رابطه با فرم‌ها و کنترل‌ها استفاده کرده‌اید. در مباحث آینده نیز نحوه استفاده از رویه‌های رویداد دیگری را فرامی‌گیرید. در این‌جا لازم است به ارایه توضیحاتی در رابطه با مهم‌ترین رویدادهای فرم‌ها و کنترل‌ها بپردازیم.

۷-۷-۱ رویه رویداد Activate

این رویه رویداد مربوط به فرم‌هاست و زمانی اجرا می‌شود که فرم، فوکوس می‌گیرد. وقتی پروژه‌ای از چند فرم تشکیل شده باشد و فوکوس از یک پنجره به پنجره دیگر

برنامه منتقل شود، رویداد Activate مربوط به پنجره‌ای که فوکوس را در اختیار گرفته است، اجرا خواهد شد. لازم به ذکر است که این رویه رویداد فاقد هر گونه آرگومان می‌باشد. این رویداد پس از رویداد Load اجرا می‌شود.

 **نکته** رویداد Activate در زمان انتقال فوکوس از پنجره یک برنامه به پنجره برنامه دیگر اجرا نخواهد شد.

۲-۷-۲ رویه رویداد Click

این رویه نیز فاقد آرگومان است و زمانی اجرا می‌شود که کاربر روی فرمی که فعال است یا کنترل موجود در آن کلیک کند.

۳-۷-۲ رویه رویداد Deactivate

این رویداد نیز فاقد آرگومان و مخصوص فرم‌هاست و عملکردی شبیه به رویداد Activate دارد، با این تفاوت که زمان اجرای آن هنگامی است که فرم، فوکوس خود را بین پنجره‌های یک برنامه از دست می‌دهد و اطلاعات فرم در حافظه باقی می‌ماند.

۴-۷-۲ رویه رویداد DbClick

این رویه رویداد در فرم‌ها و بسیاری از کنترل‌ها وجود دارد و فاقد هرگونه آرگومان است. این رویداد زمانی رخ می‌دهد که کاربر روی فرمی که فعال است یا کنترل موجود در آن، عمل دابل کلیک انجام دهد.

۵-۷-۲ رویه رویداد Load

این رویه نیز مربوط به فرم‌هاست و زمانی اجرا می‌شود که اطلاعات مربوط به فرم وارد حافظه اصلی شده و فرم، روی دسک‌تاپ نمایش داده شود. در صورتی که بخواهید دستوراتی را در زمان اجرای برنامه انجام دهید، از این رویداد استفاده کنید.


۶-۷-۲ رویه رویداد Unload

این رویداد نیز مخصوص فرم‌هاست و فاقد آرگومان می‌باشد. زمانی که فرم را می‌بندید

یا به عبارت دیگر اطلاعات فرم از حافظه کامپیوتر خارج شود، این رویداد اجرا می‌شود. در صورتی که بخواهید دستوراتی را در زمان خاتمه برنامه یا بسته شدن فرم اجرا کنید، از این رویه استفاده کنید.

۷-۷-۷ رویه رویداد GotFocus

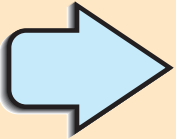
این رویداد نیز بین فرم‌ها و کنترل‌ها مشترک است و زمانی که فرم یا کنترل فوکوس را در اختیار می‌گیرند، اجرا می‌شود. در هنگامی که یک فرم، فوکوس را به دست می‌آورد ابتدا رویداد Activate و سپس رویداد GotFocus اجرا می‌شود.

 **نکته** در صورتی که در یک فرم رویداد Activate و GotFocus به‌طور هم‌زمان مورد استفاده قرار گیرند فقط رویداد Activate اجرا خواهد شد.

در صورتی که در یک فرم کنترلی وجود داشته باشد که بتواند فوکوس بگیرد (مانند کنترل Text Box) رویداد GotFocus اجرا نمی‌شود.

۷-۷-۸ رویه رویداد LostFocus

این رویداد نیز بین فرم‌ها و کنترل‌ها مشترک است و زمانی که فرم و کنترل، فوکوس خود را از دست می‌دهند، اجرا می‌شود. هنگامی که یک فرم، فوکوس را از دست می‌دهد ابتدا رویداد LostFocus و سپس رویداد Deactivate اجرا می‌شود.



Learn in English

Introduction to Procedures

You can simplify programming tasks by breaking programs into smaller logical components. These components called procedures can then become building blocks that let you enhance and extend Visual Basic. Procedures are useful for condensing repeated or shared tasks, such as frequently used calculations, text and control manipulation, and database operations. There are several types of procedures used in Visual Basic:

- Sub procedures do not return a value.
- Function Procedures return a value.
- Event Procedure.
- Visual Basic Procedure

واژه‌نامه

| | |
|--------------|---------------|
| Component | جزء |
| Condense | خلاصه کردن |
| Enhance | افزودن |
| Extend | توسعه دادن |
| Frequently | تکراری |
| Introduction | معرفی، آشنایی |
| Manipulation | دستکاری کردن |
| Private | خصوصی |
| Procedure | رویه |
| Public | عمومی |
| Scope | میدان دید |

خلاصه مطالب

- در برنامه‌نویسی به روش ساخت یافته، برنامه به بخش‌های کوچک‌تر به نام رویه تقسیم می‌شود.
- رویه‌ها در ویژوال بیسیک به چهار گروه رویه‌های فرعی و تابعی، رویه‌های رویداد و رویه‌های آماده و ویژوال بیسیک تقسیم می‌شوند.
- رویه‌های رویداد در زمان ایجاد یک شیء ایجاد می‌شوند و در زمانی که رویداد مربوطه رخ می‌دهد به‌طور خودکار اجرا می‌شوند.
- رویه‌های فرعی و تابعی را می‌توانید با تایپ در ماژول فرم یا با استفاده از کادر محاوره Add Procedure تعریف کنید.
- برای فراخوانی رویه‌های فرعی از دستور Call استفاده می‌شود.
- برای تعریف یک رویه محلی کلمه کلیدی Private را در ابتدای تعریف رویه قرار دهید.
- یک رویه محلی فقط در همان ماژولی که تعریف می‌شود قابل شناسایی و فراخوانی است.
- برای تعریف یک رویه عمومی، کلمه کلیدی Public را در ابتدای تعریف رویه قرار دهید.
- یک رویه عمومی در تمام ماژول‌های موجود در پروژه، قابل شناسایی و فراخوانی است.
- برای فراخوانی رویه‌ها از دوروش فراخوانی با مقدار یا فراخوانی با مرجع استفاده می‌شود.
- در فراخوانی رویه‌ها با مقدار یک کپی از داده‌ها در آرگومان قرار داده می‌شود.
- در فراخوانی رویه‌ها با مرجع تغییر در مقدار آرگومان‌های رویه سبب تغییر مقدار متغیرهای ارسالی می‌شود.
- در هنگام فراخوانی رویه‌ها می‌توان از نام آرگومان‌ها استفاده کرد.
- برای خروج از یک رویه فرعی و تابعی به ترتیب از دستورات Exit Sub و Exit Function استفاده می‌شود.

آزمون نظری

۱ - کدام رویداد هنگامی که یک فرم، فوکوس خود را از دست می‌دهد، اجرا می‌شود؟

الف - Activate ب - Deactivate ج - Load د - Unload

۲ - کدام گزینه در رابطه با تفاوت رویه‌های فرعی و رویه‌های تابعی درست است؟

الف - ارسال متغیرها به رویه ب - بازگشت یک مقدار به محل فراخوانی
ج - تعریف آرگومان‌ها د - محل تعریف رویه

۳ - پس از فراخوانی رویه تابعی myname به صورت «Basic»، «Visual»، «Call myname» مقدار بازگشتی چیست؟

Public Function myname (ByRef strfname As String, ByVal strlname As String) As String

Dim strname As String

strname = strfname + strlname

myname = strname

End Function

الف - «Visual» ب - «Visual Basic» ج - «Basic» د - «VisualBasic»

۴ - برای تعریف یک رویه عمومی از کلمه کلیدی استفاده می‌شود.

الف - Private ب - Public ج - Dim د - Sub

۵ - استفاده از کدام گزینه برای تعریف یک رویه فرعی مناسب است؟

الف - Sub ... End Sub ب - Function ... End Function

ج - Private د - Public

۶ - پسوند فایل‌های مازول کد در ویژوال بیسیک چیست؟

الف - VBC ب - BAS ج - CLM د - FBC

۷ - در صورتی که در یک رویه تابعی، نوع داده بازگشتی تعیین نشود از چه نوعی به طور

پیش فرض استفاده می‌شود؟

الف - Boolean ب - Integer ج - Byte د - String

۸ - به طور پیش فرض نحوه ارسال متغیرها به یک رویه از چه نوعی است؟

الف - ارسال با مقدار ب - ارسال با مرجع ج - محلی د - استاتیک

۹ - در صورت تعریف یک رویه با کلمه کلیدی Private در مازول کد.....

الف - رویه مربوطه در تمام مازول‌های کد قابل فراخوانی است.

- ب- رویه مربوطه در تمام ماژول‌ها قابل فراخوانی است.
 ج- رویه مربوطه در تمام ماژول‌های فرم قابل فراخوانی است.
 د- رویه مربوطه فقط در همان ماژول کدی که تعریف شده است قابل فراخوانی است.
 ۱۰- پس از فراخوانی تابع test به صورت test (x As Long) مقدار بازگشتی چند است؟
 (با فرض این که $x = 3$ باشد).

Public Function test (ByVal y As Long) As Long

test = y * y * y

End Function

د- ۷۱

ج- ۲۷

ب- ۹

الف- ۳

۱۱- نوع مقدار بازگشتی در رویه تابعی زیر چیست؟

Private Function Compute (into As Integer) As Boolean

د- عدد اعشاری

ج- رشته

ب- عدد صحیح

الف- منطقی

۱۲- عملکرد رویه فرعی Subchange چیست؟

Public Sub Subchange (ByRef m As String, ByRef n As String)

Dim strchange As String

strchange = m

m = n

n = strchange

End sub

الف- محتویات m و n را در رویه فرعی عوض می‌کند.

ب- محتویات m را در n قرار می‌دهد.

ج- محتویات m و n را در محل فراخوانی عوض می‌کند.

د- محتویات m و n را در رویه فرعی و هم در محل فراخوانی عوض می‌کند.

13 - Which of the following procedures can return a value?

a- Sub Procedure

b- VB Procedure

c- Functions

d- Event Procedure

۱۴- انواع رویه‌ها در ویژوال بیسیک را نام برده و هر یک را به طور خلاصه شرح دهید.

۱۵- تفاوت بین رویه‌های فرعی و تابعی را بیان کنید.

۱۶- انواع روش‌های فراخوانی رویه‌ها را توضیح دهید

۱۷- فرق بین رویه‌های عمومی و محلی را بیان کنید.

۱۸- نحوه ایجاد و استفاده از یک ماژول کد را توضیح دهید.

آزمون عملی

۱ - پروژه‌ای طراحی کنید که دمای یک اتاق را براساس فارنهایت دریافت کرده و با استفاده از یک رویه تابعی مقدار آنرا به سلسیوس محاسبه کرده و نمایش دهد.

$$c = \frac{5}{9}(F - 32)$$

۲ - پروژه‌ای طراحی کنید که یک عدد را دریافت کرده و با استفاده از یک رویه تابعی بخش‌پذیری آنرا بر اعداد ۳ و ۷ مشخص نماید.

۳ - پروژه‌ای طراحی کنید که یک عدد به عنوان مقدار زمان بر اساس ثانیه دریافت کرده و با استفاده از یک رویه فرعی مقدار ساعت آنرا محاسبه نموده و نمایش دهد.

۴ - پروژه‌ای طراحی کنید که با استفاده از یک رویه تابعی زوج یا فرد بودن هر عدد دلخواهی را مشخص کند.

۵ - پروژه‌ای طراحی کنید که موجودی یک مشتری بانک را براساس ریال دریافت کند و با استفاده از یک رویه فرعی آنرا به واحد یورو تبدیل کرده و نمایش دهد.
(۱۷۰۰ ریال = ۱ یورو)

توانایی استفاده از انواع رویه‌های آماده در ویژوال بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

۱- بتواند با توابع رشته‌ای مانند Asc، Chr، InStr، Left، Right، Len، Mid، Replace، Space

String و StrComp کار کند.

۲- بتواند با توابع تاریخ و ساعت مانند Date، Day، Month، MonthName، Weekday،

Year، Hour، Minute، Second و WeekdayName، Timer کار کند.

۳- بتواند از کنترل‌های کادر علامت (CheckBox)، دکمه انتخاب (OptionButton)،

کادر لیست (ListBox) و کادر لیست ترکیبی (ComboBox) استفاده کند و خصوصیات، رویدادها و متدهای آن‌ها را توضیح دهد.

کلیات

در ویژوال بیسیک رویه‌های متعددی وجود دارند که به صورت آماده همراه زبان برنامه‌نویسی ویژوال بیسیک در اختیار شما قرار می‌گیرند. این رویه‌ها می‌توانند از نوع رویه‌های فرعی یا رویه‌های تابعی باشند که با توجه به نیازهای روزمره برنامه‌نویسان گردآوری شده‌اند تا کار برنامه‌نویسی آسان‌تر شود و زمان لازم برای طراحی و خطازدایی دستورات کاهش پیدا کند. این رویه‌های آماده می‌توانند دارای یک یا چند آرگومان یا فاقد هر گونه آرگومان باشند، بعضی از آن‌ها مقداری را به عنوان نتیجه محاسبات بازگشت داده و بعضی هیچ مقداری را بازگشت نمی‌دهند. در صورت استفاده از این رویه‌ها ویژوال بیسیک در زمان اجرای برنامه، دستورات مربوط به رویه مورد نظر را در اختیار برنامه قرار می‌دهد. تاکنون با بعضی از این رویه‌ها مانند Val, InputBox, MsgBox, Str و غیره آشنا شده‌اید. در این جا به معرفی بعضی از رویه‌های موجود در ویژوال بیسیک می‌پردازیم.

۸-۱ توابع رشته‌ای ویژوال بیسیک

۸-۱-۱ تابع Asc

این تابع یک مقدار رشته‌ای را دریافت می‌کند و کداسکی (ASCII) اولین کاراکتر آن را به دست آورده، برمی‌گرداند. مقادیری که این تابع بازمی‌گرداند مطابق جدول کدهای اسکی یعنی صفر تا ۲۵۵ می‌باشد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:
Asc (string)

جدول ۸-۱

| خروجی | مثال |
|-------|-------------------------|
| ۶۷ | Print Asc ("Computer") |
| ۷۲ | Print Asc ("Hard Disk") |
| ۶۸ | Print Asc ("D") |
| ۴۹ | Print Asc ("I") |
| ۴۹ | Print Asc ("123") |

string یک عبارت رشته‌ای است که می‌تواند یک ثابت یا متغیر رشته‌ای یا مقدار بازگشتی یک تابع که از نوع رشته‌ای است، باشد. به عنوان مثال به مواردی که در جدول ۸-۱ ارایه شده است، توجه کنید:



مثال ۱: یک رویه تابعی بنویسید تا در پروژه‌هایی که از کادر متن برای دریافت مقادیر عددی استفاده می‌شود با استفاده از آن بتوان اعدادی را که در کادر متن وارد می‌شوند و در اولین کاراکتر آن‌ها از کاراکترهای غیررقمی استفاده شده است، تشخیص داده و از تبدیل این مقادیر به عدد جلوگیری به عمل آید.

همان‌گونه که در مثال‌های قبل مشاهده کردید گاهی اوقات لازم است که مقادیر عددی را از کادرهای متن به صورت یک رشته دریافت کنید و سپس آن‌ها را با استفاده از تابع Val به مقادیر عددی تبدیل کنید، اما در صورتی که یک مقدار رشته‌ای با کاراکترهای غیررقمی شروع شود، تابع Val مقدار صفر و در صورت استفاده از کاراکترهای غیررقمی در سایر مکان‌ها بجز اولین کاراکتر، تابع Val تا رسیدن به اولین کاراکتر غیررقمی، رشته را به عدد تبدیل می‌کند. برای جلوگیری از این مسأله می‌توانید از تابع Asc استفاده کنید. بنابراین تابع checkinput را به صورت زیر طراحی کنید:

```
Private Function checkinput(ByVal strno As String) As Boolean
```

```
If Asc(strno) >= Asc("0") And Asc(strno) <= Asc("9") Then
```

```
    checkinput = True
```

```
Else
```

```
    checkinput = False
```

```
End If
```

```
End Function
```

این تابع دارای یک آرگومان ورودی است که رشته strno را دریافت می‌کند و رشته مورد نظر با روش فراخوانی با مقدار در این متغیر کپی می‌شود، سپس با استفاده از تابع Asc کداسکی اولین کاراکتر از رشته strno بررسی می‌شود. اگر این مقدار بین ۴۸ تا ۵۷ باشد نشان دهنده این است که اولین کاراکتر از نوع رقمی است (محدوده کاراکترهای صفر تا ۹ مقادیر اسکی ۴۸ تا ۵۷ است) و در نتیجه مقدار منطقی True را بازمی‌گرداند؛ ولی اگر رشته با کاراکتر غیررقمی شروع شود، مقدار False بازگردانده خواهد شد، به این صورت می‌توانید با بررسی مقدار بازگشتی در رویه فراخوان از مقدار وارد شده در کادر متن مورد نظر مطلع شوید.



نکته اگر مقدار ارسالی به تابع Asc یک رشته خالی باشد ("") در زمان اجرای برنامه خطا رخ داده و پیام Invalid procedure call or argument نمایش داده می‌شود.



تمرین:

رویه قبل را به گونه‌ای تنظیم کنید تا در صورت ارسال یک رشته خالی در کادر متن از بروز خطا به وسیله تابع Asc جلوگیری شود.

۲-۱-۸ تابع Chr

عملکرد تابع Chr برعکس تابع Asc است. این تابع نیز یک آرگومان از نوع عددی دارد که کد اسکی کاراکتر مورد نظر را تعیین می‌کند و با توجه به کداسکی، کاراکتر مربوط به آن را به عنوان مقدار بازگشتی باز می‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Chr (code)

جدول ۲-۸

| مثال | خروجی |
|-----------------|-------|
| Print Chr (97) | a |
| Print Chr (33) | ! |
| Print Chr (116) | t |

آرگومان code یک عدد صحیح بین صفر تا ۲۵۵ است. به عنوان مثال به مواردی که در جدول ۲-۸ ارایه شده است، توجه کنید.



مثال ۲: یک رویه فرعی بنویسید تا با استفاده از دو حلقه تودرتو شکل زیر را ایجاد کرده و امکان تعیین نوع کاراکتر را نیز داشته باشد.

\$
 \$\$
 \$\$\$
 \$\$\$\$
 \$\$\$\$\$

برای انجام این کار رویه printchar را به صورت زیر بنویسید:

```
Private Sub printchar(ByVal intcode As Integer)
    Dim inti, intj As Integer
    For inti = 1 To 5
        For intj = 1 To inti
            Print Chr(intcode);
        Next intj
        Print
    Next inti
End Sub
```


همان‌طور که در این رویه فرعی مشاهده می‌کنید، یک آرگومان با نام `intcode` وجود دارد که از نوع عدد صحیح است و کداسکی کاراکتر مورد نظر را برای نمایش شکل تعیین می‌کند. پس از فراخوانی این رویه کداسکی مورد نظر در آرگومان `intcode` ذخیره می‌شود، سپس با استفاده از دو حلقه `For` و تابع `Chr` شکل مورد نظر با کاراکتر تعیین شده ایجاد می‌شود. حلقه اول در رویه، کنترل سطرها را به عهده دارد و حلقه دوم در داخل آن با توجه به مقدار شمارنده حلقه اول (`inti`) کاراکترها را در هر سطر نمایش می‌دهد. در واقع با هر بار اجرای حلقه اول، حلقه دوم به اندازه مقدار متغیر `inti` تکرار خواهد شد. وجود کاراکتر؛ در انتهای اولین دستور `Print` نیز سبب می‌شود تا کاراکترها در هر سطر در کنار هم نمایش داده شوند و دستور `Print` دوم باعث می‌شود تا پس از خاتمه نمایش کاراکترها در یک سطر، کاراکترهای بعدی در سطر جدید نمایش داده شوند و به این صورت شکل مورد نظر روی فرم ایجاد می‌شود. برای فراخوانی این رویه می‌توانید از دستور زیر استفاده کنید:

```
Call printchar (36)
```


۳-۱-۸ تابع `Len`

این تابع می‌تواند یک رشته را دریافت کند و تعداد کاراکترهای آن را به صورت یک عدد صحیح محاسبه نماید. این تابع یک آرگومان از نوع رشته‌ای دارد که وظیفه انتقال رشته را به رویه به عهده دارد. شکل کلی نحوه استفاده از این تابع به صورت زیر است که در آن `string` یک عبارت رشته‌ای است.

```
Len(string)
```

به عنوان مثال دستور زیر مقدار ۱۲ را نمایش خواهد داد:

```
Print Len("Microsoft XP")
```

 **نکته** تابع `Len` برای داده‌های نوع تاریخ نیز مانند رشته‌ها تعداد کاراکترها را مشخص می‌کند. تابع `Len` برای داده‌های نوع منطقی عدد ۲ را بازگشت می‌دهد.

۴-۱-۸ تابع `Left`

یکی دیگر از توابع رشته‌ای است که می‌تواند تعداد معینی از کاراکترها را از یک عبارت رشته‌ای و از سمت چپ آن جدا کرده و بازگرداند. شکل کلی نحوه استفاده از این

تابع به صورت زیر است:

Left (string,length)

آرگومان string یک عبارت رشته‌ای است و رشته موردنظر را دریافت می‌کند. آرگومان length نیز یک عبارت از نوع عددی است. این تابع به تعداد آرگومان length، کاراکترها را از سمت چپ رشته string بازمی‌گرداند. نوع مقدار بازگشتی این تابع از نوع رشته‌ای می‌باشد. به عنوان مثال به مواردی که در جدول ۳-۸ ارایه شده است، توجه کنید.

جدول ۳-۸

| خروجی | مثال |
|-------------------|-------------------------------------|
| Mi | Print Left ("Microsoft Windows",2) |
| Micro | Print Left ("Microsoft Windows",5) |
| Microsoft Windows | Print Left ("Microsoft Windows",17) |



مثال ۳: یک رویه فرعی بنویسید تا شماره دانشجویی یک دانشجوی دلخواه را دریافت کرده و سال ورودی را نمایش دهد.
 (با فرض این که شماره دانشجویی یک کد ۷ رقمی است که دو رقم اول سال و دو رقم دوم ماه ورودی را به دانشگاه مشخص می‌کند.) به عنوان مثال کد ۸۸۰۷۲۴۵ مشخص می‌کند دانشجو در سال ۸۸ و ماه مهر به دانشگاه وارد شده است.

```
Private Sub yearentry (Ingid As Long)
```

```
Dim intyear As Integer
```

```
intyear = Left (Ingid, 2)
```

```
intyear = intyear + 1300
```

```
Print intyear
```

```
End Sub
```

در این رویه آرگومان Ingid شماره دانشجویی را دریافت می‌کند سپس با استفاده از تابع Left، ۲ رقم سمت چپ آن را که تعیین‌کننده سال ورود دانشجو است از آرگومان Ingid جدا کرده و در متغیر intyear ذخیره می‌نماید. در مرحله بعد عدد ۱۳۰۰ به مقدار قبلی در متغیر intyear اضافه می‌شود تا سال به صورت ۴ رقمی به وسیله دستور Print نمایش داده شود.

۵-۱-۸ تابع Right

عملکرد این تابع مشابه تابع Left است با این تفاوت که این تابع تعداد کاراکترهای موردنظر را از سمت راست رشته‌ای که دریافت می‌کند، بازگشت می‌دهد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Right (string,length)

آرگومان string یک عبارت رشته‌ای است و رشته موردنظر را دریافت می‌کند. آرگومان length نیز یک عبارت از نوع عددی است که تعداد کاراکترهایی را که از سمت راست رشته string جدا می‌شوند، معین می‌کند. نوع مقدار بازگشتی این تابع از نوع رشته‌ای است.

مثال ۴: رویه فرعی yearentry را به‌گونه‌ای تغییر دهید که علاوه بر سال، ماه ورود دانشجو به دانشگاه را نیز نمایش دهد. به این منظور می‌توانید دستورات زیر را به رویه فرعی yearentry اضافه کنید.



Dim intmonth As Integer

intmonth = Left (Ingid, 4)

intmonth = Right (intmonth, 2)

Print intmonth

در این دستورات پس از تعریف متغیر موردنیاز با استفاده از تابع Left چهار رقم سمت چپ شماره دانشجو در متغیر intmonth ذخیره می‌شود سپس با استفاده از تابع Right دو رقم سمت راست ارقام جدا شده، به‌دست می‌آید که ماه را مشخص می‌کنند و به‌وسیله یک دستور Print نمایش داده می‌شوند.

۶-۱-۸ تابع Mid

این تابع می‌تواند تعداد معینی کاراکتر را از یک رشته جدا کند و دارای یک آرگومان اختیاری و دو آرگومان اجباری است. شکل کلی نحوه استفاده از این تابع به این صورت است:

Mid (string , start , length)

تابع Mid با توجه به تعدادی که آرگومان length تعیین می‌کند، کاراکترهایی را از رشته string بازمی‌گرداند. آرگومان start محلی را مشخص می‌کند که عمل جدا کردن کاراکترها از آن‌جا شروع می‌شود. اگر مقدار این آرگومان بزرگ‌تر از تعداد کاراکترهای رشته string باشد، رشته‌ای با طول صفر بازگشت داده می‌شود. آرگومان start از نوع عددی Long و

آرگومان length از نوع عددی Long می‌باشند و استفاده از آن اختیاری است. اگر از این آرگومان استفاده نشود تابع Mid کاراکترهایی را که از آرگومان start شروع می‌شوند تا انتهای رشته بازگشت خواهد داد. به عنوان مثال به مواردی که در جدول ۴-۸ ارایه شده است، توجه کنید.

جدول ۴-۸

| مثال | خروجی |
|-------------------------------------|-----------------|
| Print Mid ("Microsoft Windows",1,2) | Mi |
| Print Mid ("Microsoft Windows",6,4) | soft |
| Print Mid ("Microsoft Windows",3) | crosoft Windows |
| Print Mid ("Microsoft Windows",30) | - |



مثال ۵: یک رویه فرعی با نام uppercase با تابع Mid بنویسید. تا یک متن را دریافت کند و حروف کوچک آن را به حروف بزرگ تبدیل نماید. برای این کار رویه مزبور را به صورت زیر تغییر دهید:

```
Private Sub uppercase(ByVal strtext As String)
    Dim strchar As String * 1, strresult As String
    Dim intlength As Long, inti As Long
    intlength = Len(strtext)
    For inti = 1 To intlength
        strchar = Mid(strtext, inti, 1)
        If Asc(strchar) >= Asc("a") And Asc(strchar) <= Asc("z") Then
            strresult = strresult + Chr(Asc(strchar) - 32)
        Else
            strresult = strresult + strchar
        End If
    Next inti
    txttext.Text = strresult
End Sub
```

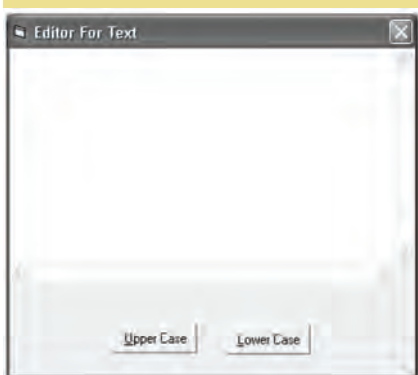
در این رویه به جای تابع Left از تابع Mid استفاده شده است تا کاراکترها یکی یکی جدا و بررسی شوند. لازم به ذکر است که در این تابع و در آرگومان دوم از متغیر inti استفاده

شده که شمارنده حلقه است و برای آرگومان سوم (که تعداد کاراکترهایی را که جدا می‌شوند، معین می‌کند) مقدار ۱ منظور شده است. بنابراین در هر بار اجرای حلقه موقعیت مربوط به کاراکتری که جدا می‌شود یک واحد زیاد خواهد شد و چون تعداد یک کاراکتر برای جدا شدن انتخاب شده است هر بار یک کاراکتر از رشته، مورد بررسی قرار خواهد گرفت؛ این امر باعث می‌شود تا استفاده از تابع Right نیز الزامی نباشد و بتوان این خط از رویه را نیز حذف کرد.



تمرین:

پروژه‌ای با نام editor و از نوع Standard EXE، مطابق شکل ۸-۱ ایجاد کنید که با استفاده از دو رویه مشابه رویه uppercase و یک کادر متن، واژه‌پردازی طراحی شود که کاربر بتواند در صورت نیاز با استفاده از دکمه‌های فرمان، محتویات آن‌را به حروف کوچک یا بزرگ تبدیل کند.



شکل ۸-۱

۸-۱-۷ Replace تابع

این تابع می‌تواند رشته‌ای را داخل رشته دیگر پیدا کند و آن را با رشته سوم جایگزین کرده و به صورت یک رشته جدید بازگرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Replace(string,find,replace,start,count,compare)

این تابع شش آرگومان دارد، سه آرگومان اول اجباری و سه آرگومان دوم اختیاری هستند. تابع، رشته find را در داخل رشته string جستجو کرده و رشته replace را به جای آن در رشته string جایگزین می‌کند. آرگومان start از نوع عددی بوده و موقعیت شروع جستجو را در رشته string تعیین می‌کند و اگر از آن استفاده نشود، مقدار ۱ در نظر گرفته می‌شود. آرگومان count نیز از نوع عددی است و تعداد دفعات جستجو و جایگزینی را تعیین می‌کند. اگر از این آرگومان استفاده نشود به طور پیش فرض مقدار ۱- در نظر گرفته می‌شود که عمل جستجو و جایگزینی را تا رسیدن به انتهای رشته string انجام خواهد داد.

آخرین آرگومان `compare` است که اگر مقدار آن برابر یک (`vbTextCompare`) باشد، تابع `Replace` در جستجوی رشته `find` در `string` بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود، اما اگر مقدار آن برابر صفر (`vbBinaryCompare`) باشد بین حروف کوچک و بزرگ تفاوت قائل می‌شود. در این جا ذکر این نکته ضروری است که تابع `Replace` هیچ تغییری در رشته `string` ایجاد نمی‌کند و حاصل عملیات جستجو و جایگزینی به صورت یک رشته مستقل بازگشت داده خواهد شد. به عنوان مثال به مواردی که در جدول ۵-۸-۸ ارائه شده است، توجه کنید.

جدول ۵-۸

| مثال | خروجی |
|--|-------------------|
| <code>Print Replace ("ABC ABC ABC","A","123")</code> | 123BC 123BC 123BC |
| <code>Print Replace ("ABC ABC ABC","A","123",3)</code> | C123BC 123BC |
| <code>Print Replace ("ABC ABC ABC","A","123",2)</code> | 123BC 123BC ABC |
| <code>Print Replace ("ABC aBC ABC","A","123")</code> | 123BC aBC 123BC |
| <code>Print Replace ("ABC aBC ABC","A","123",1)</code> | 123BC 123BC 123BC |
| <code>Print Replace ("ABC ABC ABC","B","")</code> | AC AC AC |
| <code>Print Replace ("ABC ABC ABC","D","123")</code> | ABC ABC ABC |

نکته اگر مقدار آرگومان `Replace` یک رشته خالی ("") باشد تمام مواردی که در رشته `string` پیدا می‌شوند، حذف خواهند شد.

اگر مقدار آرگومان `start` بزرگ‌تر از یک باشد و رشته `find` قبل از موقعیت تعیین شده (یعنی آرگومان `start`) در رشته `string` وجود داشته باشد، این بخش‌ها حذف می‌شوند.



شکل ۲-۸

تمرین:

واژه‌پرداز `editor` را مطابق شکل ۲-۸ به گونه‌ای تنظیم کنید تا کاربر توانایی پیدا کردن و جایگزین کردن عبارات را در متن تایپ شده داشته باشد.

۸-۱-۸ تابع StrComp


از این تابع برای مقایسه دو رشته با یکدیگر استفاده می‌شود. اگر دو رشته با هم مساوی باشند عدد صفر را بازگشت خواهد داد و در غیر این صورت اگر رشته اول بزرگ‌تر از رشته دوم باشد عدد ۱ و در صورتی که رشته اول کوچک‌تر از رشته دوم باشد عدد -۱ بازگردانده خواهد شد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:


StrComp (string1,string2,compare)

آرگومان‌های string1 و string2 از نوع رشته‌ای هستند و شامل دو عبارتی هستند که با هم مقایسه می‌شوند. آرگومان compare از نوع عددی بوده و استفاده از آن اختیاری است و در صورتی که مقدار آن برابر با ۱ (vbTextCompare) باشد تابع StrComp بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود ولی اگر مقدار آن برابر صفر (vbBinaryCompare) باشد بین حروف کوچک و بزرگ تفاوت قائل می‌شود. به عنوان مثال به مواردی که در جدول ۸-۶-۸ ارایه شده است، توجه کنید.

جدول ۸-۶

| مثال | خروجی |
|----------------------------------|-------|
| Print StrComp("basic","Basic",1) | 0 |
| Print StrComp("basic","Basic",0) | 1 |
| Print StrComp("Basic","basic") | -1 |

 **نکته** در صورت عدم استفاده از آرگومان Compare به‌طور پیش‌فرض مقدار صفر در نظر گرفته می‌شود.

 **مثال ۶:** به رویه تابعی بعد توجه کنید. این رویه دو رشته را دریافت کرده و معین می‌کند که آیا دو رشته مساوی هستند یا خیر.

```
Private Function strcompare(strstring1 As String, strstring2 As String) As Boolean
    If StrComp(strstring1, strstring2, vbTextCompare) = 0 Then
        strcompare = True
    Else
        strcompare = False
    End If
End Function
```

این تابع دو آرگومان رشته‌ای دارد که عبارت‌های رشته‌ای برای انجام مقایسه را دریافت می‌کند و با استفاده از تابع StrComp آن‌ها را مورد مقایسه قرار می‌دهد و در صورتی که دو رشته برابر باشند، مقدار صفر بازگشت داده می‌شود و این مقدار در یک دستور If بررسی شده و در نتیجه دستور پس از Then اجرا می‌شود و مقدار منطقی True را به عنوان مقدار بازگشتی بازمی‌گرداند. اما در صورت عدم تساوی دو رشته، تابع، مقدار منطقی False را بازگشت خواهد داد. البته در آرگومان سوم تابع StrComp نیز از مقدار vbText- Compare به جای عدد یک استفاده شده است که سبب می‌شود تابع بین حروف کوچک و بزرگ تفاوتی قائل نشود. بنابراین اگر این تابع به صورت ("Hard", "hard") strcompare فراخوانی شود مقدار True و اگر به صورت ("Disk" و "Ram") strcompare فراخوانی شود مقدار False را بازگشت خواهد داد.

تمرین:



برای پروژه editor یک کادر محاوره ورود به نرم‌افزار ایجاد کنید که دارای یک کلمه رمز و کلمه کاربر باشد، به علاوه برای بررسی درستی کلمه رمز و کاربر از تابع StrComp استفاده کنید.

نکته در صورتی که آرگومان char یک رشته باشد، فقط اولین کاراکتر تکرار می‌شود.



۹-۱-۸ تابع InStr

این تابع می‌تواند دو رشته را دریافت کرده و موقعیت یکی را در دیگری پیدا کند. این تابع دارای چهار آرگومان است و شکل کلی نحوه استفاده از آن به صورت زیر است:

InStr (start, string1 , string2, compare)


این تابع دارای دو آرگومان رشته‌ای string1 و string2 است که رشته string2 را در string1 جستجو می‌کند و در صورتی که نتیجه جستجو مثبت باشد موقعیت آن را به صورت یک عدد بازمی‌گرداند و در غیر این صورت مقدار صفر را برمی‌گرداند. آرگومان start از نوع عددی بوده و استفاده از آن اختیاری است این آرگومان می‌تواند نقطه شروع جستجوی string2 را در string1 تعیین کند و در صورتی که از آن استفاده نشود، موقعیت جستجو از اولین کاراکتر آغاز می‌شود. آرگومان چهارم compare است که مانند آرگومان

قبلی اختیاری است و می‌توانید از آن استفاده نکنید، اگر مقدار این آرگومان یک (vbTextCompare) باشد تابع در هنگام جستجو بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود، اما اگر مقدار این آرگومان صفر (vbBinaryCompare) باشد یا از آن استفاده نشود، در زمان جستجو بین حروف کوچک و بزرگ تفاوت قائل خواهد شد.

به عنوان مثال دستور (Instr (1, "Visual Basic", "i") رشته "i" را در رشته "Visual Basic" جستجو می‌کند و عدد ۲ را باز می‌گرداند، اما دستور (Instr (5, "Visual Basic", "i") مقدار ۱۱ را باز می‌گرداند. چون جستجو از کاراکتر پنجم آغاز می‌شود، بنابراین موقعیت رشته "i" در کلمه Basic بازگشت داده می‌شود. برای مثال می‌توانید به مواردی که در جدول ۷-۸ ارایه شده است، توجه کنید.

جدول ۷-۸

| مثال | نتیجه |
|--|-------|
| Print Instr (1, "cpu CPU CPU", "CPU") | ۵ |
| Print Instr ("CPU CPU CPU", "cpu") | ۰ |
| Print Instr (1, "cpu CPU CPU", "cpu", 0) | ۱ |
| Print Instr (4, "CPU CPU CPU", "cpu", 1) | ۵ |
| Print Instr (1, "CPU CPU CPU", "") | ۱ |
| Print Instr (3, "CPU CPU CPU", " ") | ۴ |

 **نکته** در توابعی که از آرگومان compare برای تنظیم نحوه مقایسه استفاده می‌شود، اگر از این آرگومان استفاده نشود و دستور Option Compare Text در بخش تعاریف ماژول فرم نوشته شود، تابع بین حروف کوچک و بزرگ تفاوت قائل نخواهد شد. در توابعی که آرگومان compare به کار می‌رود می‌توانید به جای مقادیر صفر و یک به ترتیب از ثابت‌های vbTextCompare و vbBinaryCompare استفاده کنید.

تمرین:



یک رویه تابعی بنویسید تا یک متن را دریافت کند و اولین جمله آن را

بازگرداند.

۲-۸ توابع تاریخ و ساعت

ویژوال بیسیک علاوه بر رویه‌های آماده‌ای که در رابطه با داده‌های عددی و رشته‌ای دارد، دارای رویه‌هایی برای انجام عملیات روی داده‌های تاریخ و ساعت نیز می‌باشد که به معرفی آن‌ها می‌پردازیم. اما ذکر یک نکته در این جا لازم است و آن این که این رویه‌ها با توجه به تنظیمات منطقه‌ای که در برنامه Regional and Language Options وجود دارد، نتیجه‌های مختلفی را در بر خواهند داشت. برای هماهنگی با مثال‌های کتاب توصیه می‌شود که از تنظیمات پیش فرض ویندوز استفاده کنید و در صورت تغییر در این تنظیمات، تنظیمات ناحیه‌ای را روی حالت انگلیسی (English) قرار دهید.

۱-۲-۸ تابع Date

این تابع، تاریخ سیستم را در اختیار شما قرار می‌دهد و فاقد هر گونه آرگومان است به علاوه با استفاده از این تابع همراه با دستور Date می‌توانید تاریخ سیستم را نیز تنظیم کنید. برای تنظیم تاریخ سیستم با دستور Date از این الگو استفاده کنید:

Date = date

جدول ۸-۸ دستور Date

Date = "2-20-1970"

Date = # 12/7/ 2003#

Date = "Aug 18,2004"

مقدار date می‌تواند از نوع داده رشته‌ای، تاریخ یا مقدار بازگشتی تابع Date باشد. به عنوان مثال موارد ارائه شده در جدول ۸-۸ نحوه استفاده از دستور Date را نشان می‌دهند.



مثال ۷: پروژه‌ای طراحی کنید که کاربر بتواند تاریخ سیستم را به وسیله آن تنظیم کند. برای این کار عملیات زیر را به ترتیب انجام دهید:

جدول ۸-۹ خصوصیات فرم

| مقدار | خصوصیت |
|----------|---------|
| frmset | Name |
| Set Date | Caption |

۱ - برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم مطابق شکل ۳-۸ و جداول ۹-۸ الی ۱۱-۸ ایجاد کنید.



۸ - ۳

جدول ۸-۱۰ خصوصیات کنترل‌ها

| کنترل / خصوصیت | Label | Label | Label | TextBox | TextBox |
|----------------|-------|-------|------------|----------|---------|
| Name | lbls۱ | lbls۲ | lbls۳ | txtmonth | txtday |
| Caption | / | / | mm dd yyyy | - | - |

جدول ۸-۱۱ خصوصیات کنترل‌ها

| کنترل / خصوصیت | TextBox | Command Button | Command Button |
|----------------|---------|----------------|----------------|
| Name | txtyear | cmdok | cmdcancel |
| Caption | - | &OK | &Cancel |

در این مرحله رویداد Click دکمه OK را به صورت زیر تنظیم کنید:

```
Private Sub cmdok_Click()
    If (Val(txtday.Text) > 0 And Val(txtday.Text) <= 31) And _
        (Val(txtmonth.Text) > 0 And Val(txtmonth.Text) <= 12) And _
        (Val(txtyear.Text) > 1900 And Val(txtyear.Text) <= 3000) Then
        Date = Trim(txtmonth.Text) + String(1, "/") + _
            Trim(txtday.Text) + String(1, "/") + Trim(txtyear.Text)
    Else
        MsgBox "Invalid Date !", , "ERROR"
    End If
End Sub
```

در این رویداد با استفاده از یک دستور If مقادیر تایپ شده در کادرهای متن روز (txtday)، ماه (txtmonth) و سال (txtyear) بررسی می‌شود در صورتی که مقادیر اشتباهی وارد شود، پیام خطایی در یک کادر پیام نمایش داده می‌شود اما در صورت مناسب بودن مقادیر ورودی با استفاده از عملگر +، توابع Trim و تابع String، داده‌های ورودی به شکل رشته‌ای حاوی تاریخ تنظیم می‌شوند و در پایان، این مقدار رشته‌ای به تابع Date داده می‌شود تا تاریخ سیستم را تنظیم کند.

دستور Unload Me را در رویداد Click دکمه Cancel بنویسید، سپس پروژه و فرم را با نام setdate ذخیره کنید.

پروژه را اجرا کرده و مقادیر ۱۰، ۲۵ و ۲۰۰ را به ترتیب برای ماه، روز و سال تایپ کنید، سپس روی دکمه OK کلیک کنید.

در ویندوز به کادر محاوره ویژگی‌های تاریخ و ساعت بروید و تاریخ ثبت شده را بررسی کنید.

این بار مقادیری را به صورت غلط تایپ کنید و روی دکمه OK کلیک کنید. همان‌طور که می‌بینید پیام خطایی روی دسک‌تاپ مشاهده می‌شود. به اجرای پروژه خاتمه داده و به پنجره ویژوال بیسیک بازگردید.

۲-۲-۸ تابع Day

تابع Day یک مقدار از نوع تاریخ را دریافت کرده و یک عدد بین ۱ تا ۳۱ به عنوان شماره روز بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Day (date)

این تابع یک آرگومان دارد که می‌تواند از نوع تاریخی، رشته‌ای یا عددی باشد که بیانگر تاریخ مشخصی است. به عنوان مثال به موارد ارائه شده در جدول ۱۲-۸ توجه کنید. در مثال چهارم جدول ۱۲-۸ از یک مقدار عددی از نوع صحیح استفاده شده است؛ این مقدار با توجه به روزهایی که از ابتدای سال سپری می‌شود، مورد بررسی قرار خواهد گرفت و در نتیجه مقدار روزها از ابتدای سال تا ماه جاری از عدد مربوطه کم می‌شود تا مقدار بازگشتی بین مقدار ۱ تا ۳۱ قرار گیرد.

جدول ۸-۱۲

| مثال | نتیجه |
|------------------------------|-------|
| Print Day (# 1/5/1980 #) | ۵ |
| 0Print Day ("march 15,198 ") | ۱۵ |
| Print Day ("1/25/1980 ") | ۲۵ |
| Print Day (50) | ۱۸ |

۸-۲-۳ تابع Now

این تابع می‌تواند در هر لحظه تاریخ و ساعت جاری سیستم را در اختیار شما قرار دهد. مقدار بازگشتی این تابع از نوع تاریخ می‌باشد و فاقد آرگومان است. شکل کلی نحوه استفاده از آن به این صورت است:

Now

به عنوان مثال نتیجه فراخوانی این تابع می‌تواند یک عبارت رشته‌ای به صورت
 # 12/10/2004 6:43:11P m # باشد.

۸-۲-۴ تابع Month

تابع Month با دریافت یک مقدار از نوع تاریخ، عددی بین صفر تا ۱۲ را که بیانگر شماره ماه تاریخ مربوط است، باز می‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت بعد است:

Month (date)

آرگومان date یک عبارت عددی یا رشته‌ای است که بیانگر یک مقدار تاریخی می‌باشد. برای مثال به موارد ارایه شده در جدول ۸-۱۳ توجه کنید.

جدول ۸-۱۳

| مثال | نتیجه |
|--------------------------------|-------|
| Print Month (" 8/10/2000 ") | ۸ |
| Print Month (" 11/23/1980 ") | ۱۱ |
| Print Month (" March 2,1960 ") | ۳ |
| Print Month (# 9-1-99 #) | ۹ |
| Print Month (# 10-30-1974 #) | ۱۰ |

۵-۲-۸ تابع MonthName

این تابع می‌تواند با دریافت شماره ترتیب ماه‌های سال، نام آن‌ها را به صورت یک عبارت رشته‌ای بازگرداند. این تابع دارای دو آرگومان است و شکل کلی نحوه استفاده از آن به صورت زیر است:

MonthName (month , abbreviate)

آرگومان اول این تابع month است که از نوع عددی بوده و شماره ماه‌های سال را از ۱ تا ۱۲ دریافت می‌کند. آرگومان دوم abbreviate اختیاری و از نوع منطقی است، اگر این آرگومان True باشد نام ماه به صورت خلاصه و در صورتی که False باشد نام ماه به صورت کامل بازگردانده می‌شود. در صورت عدم استفاده از این آرگومان مقدار پیش فرض False است. به عنوان مثال به موارد ارایه شده در جدول ۸-۱۴ توجه کنید.

جدول ۸-۱۴

| مثال | نتیجه |
|----------------------------|-------|
| Print MonthName (40 False) | April |
| Print MonthName (40 True) | Apr |
| Print MonthName (5) | May |

۶-۲-۸ تابع Weekday

این تابع یک عبارت رشته‌ای یا تاریخی را دریافت کرده و یک عدد صحیح که بیانگر شماره روزهای هفته است، باز می‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Weekday (date, firstday)

این تابع دارای دو آرگومان است. آرگومان date از نوع تاریخی یا رشته‌ای است که بیانگر یک تاریخ می‌باشد و آرگومان firstday یک آرگومان اختیاری است که روز اول هفته را برای محاسبه شماره روز تعیین می‌کند. مقدار پیش فرض برای این آرگومان روز یکشنبه است. این آرگومان می‌تواند یکی از مقادیر ارایه شده در جدول ۸-۱۵ باشد. این تابع عدد صحیحی بین ۱ تا ۷ را به ترتیب برای روزهای یکشنبه، دوشنبه تا جمعه باز می‌گرداند. به عنوان مثال به موارد ارایه شده در جدول ۸-۱۶ توجه کنید.

جدول ۸-۱۵

| ثابت رشته‌ای | ثابت عددی | روز هفته |
|--------------|-----------|----------|
| vbSunday | ۱ | یکشنبه |
| vbMonday | ۲ | دوشنبه |
| vbTuesday | ۳ | سه‌شنبه |
| vbWednesday | ۴ | چهارشنبه |
| vbThursday | ۵ | پنج‌شنبه |
| vbFriday | ۶ | جمعه |
| vbSaturday | ۷ | شنبه |

جدول ۸-۱۶

| مثال | نتیجه |
|---------------------------------------|-------|
| Print Weekday ("March 14, 1970") | ۷ |
| Print Weekday (#1/15/1960#) | ۶ |
| Print Weekday (#1/15/1960#, 5) | ۲ |
| Print Weekday (#1/15/1960#, vbFriday) | ۱ |
| Print Weekday ("11/20/1980") | ۵ |

۷-۲-۸ تابع Year

این تابع یک عبارت رشته‌ای، تاریخی، عددی یا ترکیبی (Variant) را که حاوی داده‌ای از نوع تاریخ است، دریافت کرده و مقدار سال را به صورت یک عدد صحیح بازگشت می‌دهد. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Year (date)

جدول ۸-۱۷

| مثال | نتیجه |
|---------------------------|-------|
| Print Year ("Aug 22, 69") | ۱۹۶۹ |
| Print Year ("11/22/1980") | ۱۹۸۰ |
| Print Year (#10/9/2000#) | ۲۰۰۰ |
| Print Year ("3-2-95") | ۱۹۹۵ |

آرگومان date به یک مقدار از نوع تاریخ اشاره می‌کند. به عنوان مثال به موارد ارایه شده در جدول ۸-۱۷ توجه کنید.

۸-۲-۸ تابع Time

این تابع ساعت سیستم را در اختیار شما قرار می‌دهد و فاقد هرگونه آرگومان است. به علاوه با استفاده از این تابع همراه با دستور Time می‌توانید ساعت سیستم را نیز تنظیم کنید. برای تنظیم ساعت سیستم با دستور Time از الگوی زیر استفاده کنید:

Time = time

جدول ۸-۱۸ دستور Time

| |
|-----------------------|
| Time = "17:2:30" |
| Time = "5:17" |
| Time = "8:15 PM" |
| Time = # 8:15:20 AM # |

time یک عبارت از نوع رشته‌ای، ساعت یا مقدار بازگشتی تابع Time است. به عنوان مثال موارد ارایه شده در جدول ۸-۱۸ نحوه استفاده از دستور Time را نشان می‌دهند.

در جدول فوق دستورات اول و دوم بدون استفاده از عبارات AM و PM زمان سیستم را تنظیم می‌کنند. در چنین مواقعی با توجه به مقدار ساعت تعیین شده، زمان سیستم روی قبل یا بعدازظهر تنظیم می‌شود. در مثال اول چون ساعت ۱۷ ذکر شده است، زمان روی ۵ بعدازظهر و در مثال دوم چون ساعت ۵ ذکر شده است، زمان روی ۵ صبح تنظیم خواهد شد.

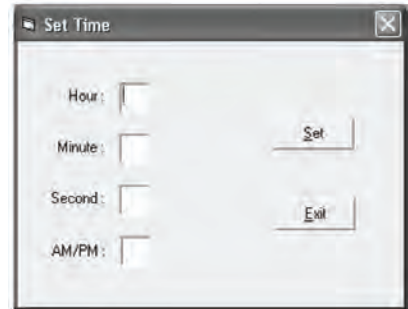


مثال ۸: پروژه‌ای طراحی کنید که کاربر بتواند ساعت سیستم را به وسیله آن تنظیم کند. برای این کار عملیات زیر را به ترتیب انجام دهید:

- ۱- یک پروژه از نوع Standard EXE ایجاد کنید که شامل فرمی مطابق شکل ۴-۸ و جداول ۱۹-۸ الی ۲۱-۸ باشد.

جدول ۱۹-۸ خصوصیات فرم

| خصوصیت | مقدار |
|---------|----------|
| Name | frmset |
| Caption | Set Time |



۴ - ۸

جدول ۲۰-۸ خصوصیات کنترل‌ها

| کنترل / خصوصیت | Label | Label | Label | Label | TextBox |
|----------------|--------|----------|----------|---------|---------|
| Name | lblh | lblm | lbls | lblampm | txthour |
| Caption | Hour : | Minute : | Second : | AM/PM | - |

جدول ۲۱-۸ خصوصیات کنترل‌ها

| کنترل / خصوصیت | TextBox | TextBox | TextBox | Command Button | Command Button |
|----------------|-----------|-----------|---------|----------------|----------------|
| Name | txtminute | txtsecond | txtampm | cmdset | cmdexit |
| Caption | - | - | - | &Set | &Exit |

۲ - رویداد دکمه Set را به صورت زیر تنظیم کنید:

```
Private Sub cmdset_Click()
```

```
Dim strtime As String
```

```
strtime = Trim(txthour.Text)+":"+Trim(txtminute.Text)+_
```

```
":"+Trim(txtsecond.Text)+""+Trim(txtampm.Text)
```

```
Time =strtime
```

```
End Sub
```

- ابتدا با استفاده از خصوصیت Text کادرهای متن مقدار ساعت، دقیقه، ثانیه و قبل یا بعد از ظهر زمان موردنظر به صورت یک رشته در متغیر strtime ذخیره می‌شود و سپس این مقدار با استفاده از تابع Time به عنوان زمان سیستم ثبت می‌شود.
- ۳ - پروژه و فرم را با نام settime ذخیره کنید، سپس آن را اجرا نمایید.
- ۴ - مقادیر ۱۰، ۲۵، ۱۸ و AM را به ترتیب برای ساعت، دقیقه، ثانیه و به عنوان زمان قبل از ظهر در کادرهای متن مربوطه تایپ کنید و روی دکمه Set کلیک کنید. سپس تغییرات را در نوار وظیفه ویندوز بررسی کنید.
- ۵ - به اجرای پروژه خاتمه داده و به پنجره ویژوال بیسیک بازگردید.

تمرین:



با استفاده از تابع Time و کنترل Timer یک زمان‌سنج دیجیتال را طراحی کنید.

۹-۲-۸ تابع Hour

این تابع با دریافت یک مقدار رشته‌ای یا تاریخی شامل زمان، مقدار ساعت را به صورت یک عدد صحیح بین صفر و ۲۳ بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Hour (time)

آرگومان time می‌تواند از نوع رشته‌ای، تاریخ و زمان یا ترکیبی (Variant) باشد که بیانگر مقادیر زمانی هستند. به عنوان مثال به موارد ارایه شده در جدول ۲۲-۸ توجه کنید.



نکته اگر آرگومان time مقادیر زمانی بزرگتر از مقادیر مجاز برای ساعت، دقیقه یا ثانیه باشد، در هنگام اجرای برنامه پیام خطای Syntax error یا Type mismatch نمایش داده می شود.
 اگر در آرگومان time مقدار بعد از ظهر (PM) یا قبل از ظهر (AM) تعیین نشود، مقدار ساعت با توجه به مقدار زمان در آرگومان time محاسبه می شود.

جدول ۸-۲۲

| مثال | نتیجه |
|------------------------------|-----------------|
| Print Hour (#8:18:10 AM#) | ۸ |
| Print Hour (#22:05:00 #) | ۲۲ |
| Print Hour (" 10:50:43 PM ") | ۲۲ |
| Print Hour (" 13:20 ") | ۱۳ |
| Print Hour (" 4:10 ") | ۴ |
| Print Hour (Now) | ساعت جاری سیستم |

۱۰-۲-۸ تابع Minute

این تابع با دریافت یک مقدار رشته‌ای یا تاریخی شامل زمان، مقدار دقیقه را به صورت یک عدد صحیح بین صفر و ۵۹ بازمی گرداند. شکل کلی نحوه استفاده از این تابع به این صورت است:

Minute (time)

آرگومان time می تواند از نوع رشته‌ای، تاریخ و زمان یا ترکیبی (Variant) باشد که بیانگر مقادیر زمانی هستند. برای مثال به موارد ارایه شده در جدول ۸-۲۳ توجه کنید.

جدول ۸-۲۳

| مثال | نتیجه |
|--------------------------------|----------------------------------|
| Print Minute (#8:18:10 AM#) | ۱۸ |
| Print Minute (#22:05:00#) | ۵ |
| Print Minute (" 10:50:43 PM ") | ۵۰ |
| Print Minute (" 13:20") | ۲۰ |
| Print Minute (" 4:10:3") | ۱۰ |
| Print Minute (Now) | دقیقه با توجه به ساعت جاری سیستم |



نکته اگر مقادیر زمانی مورد استفاده بزرگ‌تر از مقادیر مجاز برای ساعت، دقیقه یا ثانیه باشند، در هنگام اجرای برنامه پیام خطای Syntax error یا Type mismatch مشاهده می‌شود.

۸-۲-۱۱ تابع Second

این تابع نیز با دریافت یک مقدار رشته‌ای یا تاریخی شامل زمان، مقدار ثانیه را به صورت یک عدد صحیح بین صفر تا ۵۹ بازمی‌گرداند. شکل کلی نحوه استفاده از این تابع به صورت زیر است:

Second (time)

آرگومان time می‌تواند از نوع رشته‌ای، تاریخ و زمان یا ترکیبی (Variant) باشد که بیانگر مقادیر زمانی هستند. برای مثال به موارد ارایه شده در جدول ۸-۲۴ توجه کنید.

جدول ۸-۲۴

| مثال | نتیجه |
|-------------------------------|--|
| Print Second (# 8:18:10 Am #) | ۱۰ |
| Print Second (#22:05:00 #) | ۰ |
| Print Second (" 10:50:43 PM") | ۴۳ |
| Print Second (" 13:20:2") | ۲ |
| Print Second (Now) | مقدار ثانیه با توجه به ساعت جاری سیستم |



نکته اگر مقادیر زمانی مورد استفاده بزرگ‌تر از مقادیر مجاز برای ساعت، دقیقه یا ثانیه باشند، در هنگام اجرای برنامه پیام خطای Syntax error یا Type mismatch مشاهده می‌شود.

۸-۳ کنترل کادر در لیست

کنترل‌های کادر لیست یکی دیگر از کنترل‌های ویژوال بیسیک هستند که به کاربر اجازه می‌دهند تا از بین چند مقدار مختلف یکی را انتخاب کند. به عنوان مثال می‌توان به کادرهای لیستی که در کادر محاوره تنظیمات ویژگی‌های نمایشی ویندوز وجود دارند اشاره کرد.

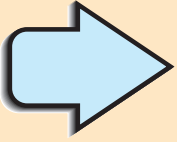
کادرهای لیست انواع مختلفی شامل کادرهای لیست معمولی (ListBox) و کادرهای لیست ترکیبی بازشو (ComboBox) دارند. تفاوتی که بین این دو نوع کنترل علاوه بر شکل ظاهری وجود دارد این است که کادرهای لیست ساده فاقد کادرمتن هستند اما کادرهای لیست ترکیبی بازشو از یک کادر لیست ساده و یک کادرمتن تشکیل می‌شوند.

جدول ۲۵-۸

| توضیح | خاصیت/متد/رویداد |
|--|------------------|
| این خاصیت اعضای موجود در کادر لیست را نگهداری می‌کند، این خاصیت در تمام انواع کنترل‌های کادر لیست وجود دارد. | List |
| این خاصیت در تمام کنترل‌های کادر لیست وجود دارد و از نوع منطقی است. اگر مقدار این خاصیت را روی True تنظیم کنید. اعضای موجود در کادر لیست به صورت صعودی مرتب می‌شوند، اما اگر مقدار آن روی False تنظیم شود اعضای موجود در کنترل به همان ترتیبی که در خاصیت List تایپ شده‌اند، نمایش داده می‌شوند. | Sorted |
| اگر این خاصیت در کنترل‌های ListBox روی مقدار Standard-۰ تنظیم شود، کنترل به صورت یک کادر لیست ساده و در صورتی که روی مقدار CheckBox-۱ تنظیم شود، هر یک از اعضا در کنترل به همراه یک کادر علامت نمایش داده می‌شوند، این خاصیت می‌تواند در کنترل‌های ComboBox سه مقدار را کسب کند؛ اگر روی مقدار Dropdown Combo-۰ تنظیم شود کنترل به صورت یک کادر لیست بازشو به همراه یک کادر متن و اگر روی مقدار Simple Combo List-۱ تنظیم شود، کنترل به صورت یک کادر لیست بازشو ساده به همراه یک کادرمتن و در صورتی که روی مقدار Dropdown List-۲ تنظیم شود به صورت یک کادر لیست بازشو و بدون کادر متن نمایش داده می‌شود. | Style |
| این خاصیت در کنترل کادر لیست بازشو از نوع Simple Combo و Dropdown Combo قابل استفاده است و عملکرد آن مشابه همین خاصیت در کنترل کادر متن است. | Locked |
| این خاصیت در کنترل کادر لیست ساده و کنترل‌های کادر لیست بازشو بجز نوع Dropdown List قابل استفاده است و عنوان عضو انتخاب شده در کنترل را در خود نگهداری می‌کند. | Text |

ادامه جدول ۲۵-۸

| خاصیت / متد / رویداد | توضیح |
|----------------------|--|
| AddItem | <p>این متد می‌تواند عضوی را به کنترل کادر لیست اضافه کند و شکل کلی نحوه استفاده از آن به صورت بعد است:</p> <p style="text-align: center;">(عنوان عضو) AddItem . نام کنترل کادر لیست</p> |
| Clear | <p>به وسیله این متد می‌توان تمام اعضای موجود در کنترل کادر لیست را حذف کرد. شکل کلی نحوه استفاده از این متد به صورت زیر است:</p> <p style="text-align: center;">Clear . نام کنترل کادر لیست</p> |
| RemoveItem | <p>این متد می‌تواند هر یک از اعضای موجود در کنترل را براساس شماره اندیس آن‌ها در خاصیت List حذف کند. شماره اندیس از مقدار صفر برای اولین عضو آغاز می‌شود و به ترتیب قرار گرفتن اعضا در لیست افزایش می‌یابد. شکل کلی نحوه استفاده از این متد به این صورت است:</p> <p style="text-align: center;">شماره‌اندیس عضو در کنترل) RemoveItem . نام کنترل کادر لیست</p> |
| Click | <p>این رویداد در تمام انواع کادرهای لیست قابل استفاده است و زمانی اجرا می‌شود که کاربر یکی از اعضای موجود در کادر لیست را انتخاب کند.</p> |
| Change | <p>این رویداد در کنترل‌های کادر لیست باز شو از نوع Simple Combo و Dropdown Combo قابل استفاده است و زمانی اجرا می‌شود که محتویات کادر متن در کادر لیست تغییر کند.</p> |



Learn in English

Len Function: Returns a Long containing the number of characters in a string or the number of bytes required to store a variable.

Syntax

Len(*string* | *varname*)

The **Len** function syntax has these parts:

| Part | Description |
|---------|--|
| String | Any valid string expression. If string contains Null , Null is returned. |
| Varname | Any valid variable name. If <i>varname</i> contains Null , Null is returned. If <i>varname</i> is a Variant , Len treats it the same as a String and always returns the number of characters it contains. |

واژه‌نامه

| | |
|---------------|--------------------------|
| Abbreviate | کوتاه کردن، مختصر کردن |
| Argument | آرگومان |
| Binary | دودویی، مبنای دو |
| Compare | مقایسه کردن |
| Day | روز |
| Editor | واژه‌پرداز |
| Error | خطا |
| Expression | عبارت |
| Hour | ساعت |
| Invalid | اشتباه |
| Length | طول |
| Minute | دقیقه |
| Mismatch | عدم تطابق، مطابقت نداشتن |
| Month | ماه |
| Month Name | نام ماه |
| Remove | رفع کردن، برداشتن |
| Replace | جایگزین کردن |
| Right | راست |
| Second | ثانیه |
| Sort | مرتب کردن |
| Space | فضای خالی |
| Start | شروع |
| Syntax | شکل نوشتاری دستور |
| Syntax | نحو، ترکیب |
| Time | زمان |
| Treat | رفتار کردن |
| Week Day | روز هفته |
| Week day Name | نام روز هفته |
| Year | سال |

خلاصه مطالب

- تابع ASC یک رشته را دریافت کرده و کداسکی اولین کاراکتر را برمی گرداند.
- تابع Chr کداسکی یک کاراکتر را دریافت کرده و کاراکتر معادل آن را برمی گرداند.
- تابع InStr دو رشته را دریافت کرده، یکی را در دیگری جستجو می کند.
- تابع Left تعداد معینی از کاراکترهای یک رشته را از سمت چپ آن برمی گرداند.
- تابع Right تعداد معینی از کاراکترهای یک رشته را از سمت راست آن برمی گرداند.
- تابع Len یک رشته را دریافت کرده و طول آن را معین می کند.
- تابع Mid یک رشته را دریافت کرده و تعداد معینی کاراکتر را از آن جدا می کند.
- تابع Replace رشته ای را در رشته دیگری جایگزین می کند.
- تابع StrComp دو رشته را دریافت کرده و با هم مقایسه می کند.
- با استفاده از تابع Date می توان تاریخ سیستم را به دست آورده یا تنظیم کرد.
- تابع Day یک مقدار از نوع تاریخ را دریافت کرده و شماره روز آن را برمی گرداند.
- با استفاده از تابع Now می توان تاریخ و ساعت سیستم را به دست آورد.
- تابع Month یک مقدار از نوع تاریخ را دریافت کرده و شماره ماه آن را برمی گرداند.
- تابع MonthName با دریافت شماره ماه، نام آن را برمی گرداند.
- تابع Weekday یک عبارت رشته ای یا تاریخی را دریافت کرده و شماره ترتیب روز را در هفته برمی گرداند.
- تابع Year یک عبارت رشته ای، تاریخی را دریافت کرده و مقدار سال آن را برمی گرداند.
- با استفاده از تابع Time می توان ساعت سیستم را به دست آورد.
- تابع Hour یک عبارت رشته ای یا تاریخی شامل زمان را دریافت کرده و مقدار ساعت را برمی گرداند.
- تابع Minute یک عبارت رشته ای یا تاریخی شامل زمان را دریافت کرده و مقدار دقیقه را برمی گرداند.
- تابع Second یک عبارت رشته ای یا تاریخی شامل زمان را دریافت کرده و مقدار ثانیه را برمی گرداند.
- با استفاده از تابع Timer می توان ثانیه های بعد از نیمه شب را محاسبه کرد.

آزمون نظری

- ۱ - حاصل اجرای دستور "123", Len "Windows", Right چیست؟
 الف - "۱۲۳" ب - "Win" ج - "ows" د - "ind"
- ۲ - به وسیله کدام تابع می‌توان تعدادی از کاراکترهای یک رشته را جدا کرد؟
 الف - Instr ب - Mid ج - Len د - Asc
- ۳ - حاصل اجرای دستور (3, "Computer", Left) چیست؟
 الف - "Com" ب - "ter" ج - "puter" د - ""
- ۴ - کدام تابع امکان جستجوی یک رشته را در رشته دیگر فراهم می‌کند؟
 الف - Len ب - InStr ج - Replace د - Space
- ۵ - کدام گزینه در رابطه با خروجی دستور «10:12:36» Second درست است؟
 الف - ۱۰ ب - ۱۲ ج - ۳۶ د - ۴۸
- ۶ - حاصل عبارت (5,5, "Computer", Mid) چیست؟
 الف - omput ب - uter ج - puter د - Computer
- ۷ - به وسیله کدام تابع می‌توان ساعت و تاریخ جاری سیستم را به دست آورد؟
 الف - Date ب - Time ج - Now د - گزینه‌های الف و ج صحیح هستند.
- ۸ - حاصل عبارت (3, "t", "o", "Book", Replace) چیست؟
 الف - "Botk" ب - "Bttt" ج - "Bttk" د - "Btok"
- ۹ - Which of the following data types does return by the Len Function?
 a- integer b- string c- date d- boolean
- ۱۰ - تفاوت و شباهت توابع Date و Now را بیان کنید.
- ۱۱ - نحوه عملکرد توابع زیر را توضیح دهید.
 الف - Left ب - Mid ج - Strcomp د - Instr
- ۱۲ - نحوه عملکرد توابع زیر را توضیح دهید.
 الف - Day ب - Asc ج - Month Name د - Replace
- ۱۳ - نحوه عملکرد توابع Year، Hour و Minute را توضیح دهید.

آزمون عملی

- ۱ - یک رویه فرعی بنویسید که یک عدد را دریافت کرده، کوچک‌ترین رقم آن را پیدا کند و نمایش دهد.
- ۲ - یک رویه فرعی بنویسید که یک رشته را دریافت کرده و کاراکترهای آن را به صورت معکوس نمایش دهد. به عنوان مثال عبارت Computer را به صورت retupmoC نمایش دهد.

نحوه استفاده از رویدادهای ماوس و صفحه کلید

هدفهای رفتاری

- پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:
- ۱- رویدادهای ماوس و صفحه کلید را بیان کند.
 - ۲- بتواند از رویدادهای ماوس و صفحه کلید استفاده کند.
 - ۳- کنترل‌های شکل (Shape) و خط (Line) را شرح داده، خصوصیات آن‌ها را توضیح دهد.
 - ۴- بتواند از کنترل‌های شکل (Shape) و خط (Line) استفاده کند.

کلیات

یکی از مسایلی که در برنامه‌نویسی برای سیستم‌های عامل با رابط گرافیکی مانند ویندوز مورد توجه قرار می‌گیرد، عکس‌العمل‌های مناسب در رابطه با اتفاقاتی مانند فشردن دکمه‌های ماوس، حرکت اشاره‌گر آن یا فشردن کلیدها در صفحه کلید است. در زبان ویژوال بیسیک رویدادهای مناسب در این زمینه برای برآورده کردن نیازهای برنامه‌نویسان در نظر گرفته شده‌اند.

۹-۱ رویدادهای ماوس

رویدادهای ماوس در ویژوال بیسیک به چهار گروه تقسیم می‌شوند که عبارتند از: Click، DbClick، MouseDown، MouseUp، MouseMove و DragDrop. در این واحد کار رویدادهایی را که در این رابطه در ویژوال بیسیک وجود دارند، فرا می‌گیرید.

۹-۱-۱ رویداد MouseDown

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یکی از دکمه‌های ماوس به پایین فشرده شود.

`Private Sub MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)`

همان‌طور که مشاهده می‌کنید این رویداد برخلاف رویدادهایی که تاکنون دیده‌اید دکمه‌ای را که کاربر فشرده است، Button دارای چهار آرگومان است. آرگومان اول یعنی مشخص می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۹-۱ باشد به این شکل می‌توانید هر دکمه‌ای از ماوس را که فشرده می‌شود، شناسایی کنید.

جدول ۹-۱

| ثابت رشته‌ای | ثابت عددی | توضیح |
|----------------|-----------|---|
| vbLeftButton | ۱ | در صورتی که دکمه سمت چپ ماوس فشرده شود. |
| vbRightButton | ۲ | در صورتی که دکمه سمت راست ماوس فشرده شود. |
| vbMiddleButton | ۴ | در صورتی که دکمه وسط ماوس فشرده شود. |
| Right Left | ۳ | هر دو |

آرگومان Shift معین می‌کند که در هنگام فشردن دکمه‌های ماوس کدام یک از کلیدهای Alt, Ctrl, Shift یا ترکیبی از آنها فشرده شده است. مقادیر مربوط به آرگومان Shift در جدول ۹-۲ ارایه شده‌اند.

جدول ۹-۲

| توضیح | ثابت عددی | ثابت رشته‌ای |
|--|-----------|----------------------------------|
| در صورتی که کلید Shift فشرده شود. | ۱ | vbShiftMask |
| در صورتی که کلید Ctrl فشرده شود. | ۲ | vbCtrlMask |
| در صورتی که کلید Alt فشرده شود. | ۴ | vbAltMask |
| در صورتی که کلیدهای Shift+Ctrl فشرده شوند. | ۳ | vbShiftMask+vbCtrlMask |
| در صورتی که کلیدهای Shift+Alt فشرده شوند. | ۵ | vbShiftMask+vbAltMask |
| در صورتی که کلیدهای Ctrl+Alt فشرده شوند. | ۶ | vbCtrlMask+vbAltMask |
| در صورتی که کلیدهای Shift+Ctrl+Alt فشرده شوند. | ۷ | vbShiftMask+vbCtrlMask+vbAltMask |

آرگومان‌های X و Y از نوع عدد اعشاری بوده و موقعیت اشاره‌گر ماوس را در زمان فشرده شدن کلیدهای آن معین می‌کند. مقدار X فاصله از سمت چپ و Y فاصله از بالای فرم یا کنترل مربوطه می‌باشند. نام شیء نیز می‌تواند نام یک کنترل یا عبارت Form برای فرم‌ها باشد.

۹-۱-۲ رویداد MouseUp

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یکی از دکمه‌های ماوس که به پایین فشرده شده است، رها شود.

Private Sub `نام شیء` _MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)

آرگومان‌های این رویداد دقیقاً مشابه رویداد MouseDown است.

۹-۱-۳ رویداد MouseMove

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که اشاره‌گر ماوس روی فرم یا کنترل مربوطه حرکت کند.

Private Sub `نام شیء` _MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)

آرگومان‌های این رویداد نیز دقیقاً مشابه دو رویداد قبلی می‌باشند.



شکل ۹-۱



مثال ۱: پروژه‌ای طراحی کنید که

تصویری مطابق شکل ۹-۱ روی یک فرم نمایش دهد و اگر کاربر روی تصویر کلیک کند تصویر با حالت خوشحال نشان داده شود و در صورتی که کاربر کلیک راست کند تصویر با حالت ناراحت نشان داده شود.

۱ - یک پروژه از نوع Standard EXE ایجاد کرده و یک فرم با نام frmface و یک کنترل تصویر با نام imgface مطابق شکل ۹-۱ ایجاد کنید.

۲ - در کنترل تصویر با استفاده از کادرمحاوره Load Picture تصویر FACE10 را از مسیر

D:\Program Files\Microsoft Visual studio\Common\Graphics\Icons\Misc

انتخاب کنید. (با فرض این که برنامه ویژوال استودیو در درایو D: نصب شده باشد).

۳ - رویداد Mouse Down کنترل imgface را به صورت زیر تنظیم کنید.

```
Private Sub imgface_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
If Button = vbLeftButton Then
```

```
imgface.Picture = LoadPicture _
```

```
("C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Misc\FACE02.ICO")
```

```
If Button = vbRightButton Then
```

```
imgface.Picture = LoadPicture _
```

```
("C:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Misc\FACE04.ICO")
```

```
End Sub
```

در این رویداد با استفاده از دستور If مقدار آرگومان Button بررسی می‌شود اگر این مقدار برابر با

Vb Left Button باشد به این معنی است که دکمه سمت چپ ماوس فشرده شده است

بنابراین تابع Load Picture تصویر صورت را با حالت خنده نمایش می‌دهد. به همین

ترتیب اگر دکمه سمت راست ماوس فشرده شود نتیجه بررسی شرط موجود در دستور If دوم درست (True) می‌شود. بنابراین تصویر صورت با حالت ناراحت نمایش داده می‌شود.

- ۴ - پروژه و فرم را با نام Show Face ذخیره نمایید سپس پروژه را اجرا کنید و روی تصویر یک بار کلیک و بار دیگر کلیک راست کنید و نتیجه را بررسی نمایید.
- ۵ - اجرای برنامه را متوقف کرده و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه ShowFace را به شکلی تنظیم کنید تا پس از رها کردن دکمه سمت چپ یا راست، تصویر اول دوباره نمایش داده شود.

۴-۱-۹ رویداد DragDrop

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یک کنترل در روی فرم با استفاده از اشاره‌گر ماوس از یک محل به محل دیگر منتقل ((Drag & Drop) شود.

Private Sub نام شیء _ (Source As Control, X As Single, Y As Single) DragDrop

آرگومان Source شیء که عملیات انتقال (Drag & Drop) روی آن انجام شده است، تعیین می‌کند. اگر شیء یک کنترل باشد نام کنترل و اگر فرم باشد کلمه Form مورد استفاده قرار می‌گیرد. در رابطه با این رویداد باید توجه داشته باشید که رویداد DragDrop یک کنترل یا فرم زمانی اجرا می‌شود که کنترل دیگری پس از Drag شدن روی آن Drop شود. آرگومان‌های X و Y موقعیت نقطه‌ای را که عمل رها کردن (Drop) در آنجا صورت گرفته است، مشخص می‌کنند. اما لازم است بدانید که برای انجام عمل انتقال یک کنترل در روی فرم باید خصوصیت DragMode را برای کنترل موردنظر روی مقدار 1-Automatic تنظیم کنید و به علاوه دستور Source.Move x,y را در رویداد DragDrop فرم قرار دهید.



مثال ۲: پروژه Calculator را که در واحد کار پنجم طراحی کرده‌اید به شکلی تغییر

دهید تا بتوان با عمل Drag & Drop محتویات کادرمتن txtnum1 را در کادرمتن txtnum2 کپی کرد. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱- پروژه Calculator را باز کنید.

۲- خصوصیت Drag Mode کنترل کادرمتن $txtnum_1$ را روی مقدار ۱-Automatic تنظیم

کنید تا امکان انجام عمل Drag & Drop روی کنترل کادرمتن امکان پذیر باشد.

۳- رویداد Drag Drop کنترل کادرمتن $txtnum_2$ را به این صورت تنظیم کنید.

```
Private Sub txtnum2_DragDrop(Source As Control, X As Single, Y As Single)
```

```
txtnum2.Text = txtnum1.Text
```

```
End Sub
```

وقتی کنترل $txtnum_1$ در زمان اجرای برنامه با عمل Drag & Drop روی کنترل $txtnum_2$

قرار بگیرد این رویداد اجرا می شود و محتویات کادرمتن $txtnum_1$ را روی $txtnum_2$ ذخیره می کند.

۴- تغییرات را ذخیره کنید و برنامه را اجرا نمایید.

۵- یک عدد در کادرمتن اول تایپ کنید سپس با عمل Drag & Drop کادرمتن اول را

روی کادرمتن دوم بیندازید و نتیجه را بررسی کنید.

۶- اجرای برنامه را خاتمه داده و به پنجره ویژوال بیسیک بازگردید.

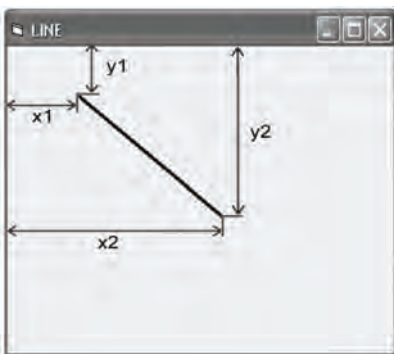


تمرین:

پروژه Calculator را به شکلی تنظیم کنید تا بتوان با عمل Drag & Drop محتویات

کادرمتن دوم را نیز در کادرمتن دوم کپی نمود.

۲-۹ کنترل خط (Line)



شکل ۲-۹

با استفاده از این کنترل می توان انواع خطوط

افقی و عمودی و مورب را رسم نمود. در این

کنترل خصوصیت های x_1 و y_1 مختصات نقطه

شروع برای ترسیم خط و خصوصیت های x_2 و y_2

مختصات نقطه انتهایی خط را مشخص می کنند

(شکل ۲-۹). به علاوه با استفاده از خصوصیت


BorderColor می توان رنگ خط و با خصوصیت

BorderWidth ضخامت خط را تعیین نمود.



شکل ۳-۹

مقادیری که خصوصیت `Border` -
`Style` می تواند کسب کند مشابه مقادیر
همین خصوصیت در کنترل `Line` است،
خصوصیت `Shape` نیز مقادیر مختلفی را
کسب می کند که در شکل ۳-۹ نمایش
داده شده اند.

 **نکته** واحد اندازه گیری پیش فرض در ترسیم اشکال هندسی در روی فرم، `Twip` است. اگر
بخواهید این واحد اندازه گیری را تغییر دهید، می توانید از خصوصیت `ScaleMode` فرم استفاده کنید.
این خصوصیت واحدهای اندازه گیری مختلفی مانند `Pixel`، `Twip`، `Character`، `Inch`، `Millimeter` و
`Centimeter` را در اختیار شما قرار می دهد.

۹-۴ رویدادهای صفحه کلید

در ویژوال بیسیک سه رویداد برای کلیدهای صفحه کلید در نظر گرفته شده است این
رویدادها شامل رویدادهای `KeyDown`، `KeyUp` و `KeyPress` هستند.

۹-۴-۱ رویداد `KeyDown`

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می شود که یکی از
کلیدهای صفحه کلید فشرده شوند.

Private Sub `KeyDown(KeyCode As Integer, Shift As Integer)` **نام شیء**

این رویداد دو آرگومان دارد. آرگومان `KeyCode` کداسکی کلید فشرده شده را معین
می کند. در صفحه کلید به هر کلید یک عدد صحیح مثبت بین صفر تا ۲۵۵ نسبت داده
شده است که با عنوان کداسکی (ASCII) نام گذاری شده اند. آرگومان `Shift` مقداری را با
توجه به فشرده شدن کلیدهای `Ctrl`، `Alt` و `Shift` یا ترکیبی از آن ها را که به طور هم زمان با


کلید اصلی فشرده شده‌اند در اختیار شما قرار می‌دهد. مقادیر مربوط به این آرگومان‌ها به صورت ضمیمه در انتهای کتاب ارایه شده‌اند. برای کنترل‌ها از نام کنترل و برای فرم‌ها کلمه Form به‌عنوان نام شیء استفاده می‌شود.

۲-۴-۹ رویداد KeyUp

شکل کلی این رویداد نیز به صورت زیر است و زمانی اجرا می‌شود که کلید فشرده شده در صفحه کلید رها می‌شود.

Private Sub KeyUp(KeyCode As Integer, Shift As Integer) نام شیء

آرگومان‌های این رویداد کاملاً مشابه رویداد KeyDown هستند و نام شیء می‌تواند نام یک کنترل یا عبارت Form برای فرم‌ها باشد.

 **نکته** برای تشخیص کاراکترهایی که با استفاده از کلید Shift تایپ می‌شوند (مانند @) و غیره) از آرگومان KeyCode و Shift به‌طور هم‌زمان استفاده کنید.

۳-۴-۹ رویداد KeyPress

شکل کلی این رویه رویداد به صورت زیر است و زمانی اجرا می‌شود که یکی از کلیدهای صفحه کلید فشرده شده و رها شود:

Private Sub KeyPress(KeyAscii As Integer) نام شیء

این رویه یک آرگومان از نوع عدد صحیح دارد که کداسکی کلید فشرده شده را مشخص می‌کند. مقادیر مربوط به این آرگومان به صورت ضمیمه در انتهای کتاب ارایه شده‌اند. نام شیء می‌تواند نام یک کنترل یا عبارت Form برای فرم‌ها باشد.

تفاوتی که بین این رویداد با رویدادهای دیگر صفحه کلید وجود دارد این است که رویداد KeyPress برای حروف بزرگ و کوچک حرفی کدهای متفاوتی ارایه می‌کند، در صورتی که رویدادهای KeyUp و KeyDown کدهای یکسانی را بازگشت می‌دهند. به عنوان مثال رویداد KeyPress برای کاراکتر A کد ۶۵ و برای کاراکتر a کد ۹۷ تولید می‌کند، اما رویداد KeyDown برای هر دو کاراکتر کد ۶۵ را تولید می‌کند. به‌علاوه رویداد KeyPress برای سایر کاراکترها مانند علائم < ، > ، ؟ و غیره کدهایی را مطابق جدول ضمیمه در انتهای کتاب تولید می‌کند.



- **نکته** رویداد KeyDown نسبت به KeyPress اولویت دارد و زودتر اجرا می‌شود.
- در رویداد KeyPress کد مربوط به کلیدهای رقمی در بخش عددی صفحه کلید مطابق با کد کلیدهای رقمی در بخش حرفی صفحه کلید است.
- رویداد KeyPress ترکیب کلید Ctrl با کاراکترهای حرفی و علائم [و] را پشتیبانی می‌کند.
- رویداد KeyPress از کلیدهای تابعی، حرکت مکان‌نما و نظیر آن‌ها پشتیبانی نمی‌کند.
- رویداد KeyPress از کلیدهای Enter، Backspace و ESC پشتیبانی می‌کند.

۴-۹- خصوصیت KeyPreview

این خصوصیت یکی از خصوصیات مهم در فرم‌هاست و نحوه اجرای رویه‌های رویداد صفحه کلید را برای فرم و کنترل‌های موجود در روی آن معین می‌کند. این خصوصیت از نوع منطقی می‌باشد و مقدار پیش فرض آن مقدار False است. در این حالت اگر کلیدی در صفحه کلید فشرده شود، رویدادهای صفحه کلید مربوط به کنترلی که فوکوس را در اختیار دارد، اجرا می‌شوند. اما اگر مقدار این خصوصیت روی True تنظیم شده باشد ابتدا رویدادهای صفحه کلید فرم اجرا می‌شوند و سپس رویدادهای صفحه کلید مربوط به کنترلی که فوکوس را در اختیار دارد، اجرا می‌شوند. در ضمن اگر هیچ کنترلی وجود نداشته باشد رویدادهای صفحه کلید فرم اجرا می‌شوند.



- **نکته** وقتی فوکوس روی کنترل دکمه فرمان قرار می‌گیرد، کلیدهای Enter و حرکت مکان‌نما نمی‌توانند رویدادهای صفحه کلید فرم و کنترل دکمه فرمان را اجرا نمایند.



مثال ۳: یک بازی طراحی کنید که بازیکن، یک توپ متحرک را با استفاده از کلیدهای جهت‌دار در داخل یک قاب کنترل کند و از برخورد آن با دیوارهای این قاب جلوگیری نماید، در صورتی که توپ با دیوارها برخورد کند بازی خاتمه یافته و براساس مدت زمانی که بازیکن توانسته است توپ را کنترل کند، امتیاز وی محاسبه می‌شود. به علاوه کاربر بتواند با فشردن کلید ترکیبی Ctrl+Q هر زمان که مایل باشد از بازی خارج شود و با استفاده از کلید ترکیبی Alt+P حرکت توپ را به طور موقت متوقف کرده و با استفاده از کلید ترکیبی Shift+S توپ را مجدداً به حرکت در آورد.



شکل ۴-۹

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE به همراه یک فرم با عرض و ارتفاع، ۴۷۸۵ و ۳۷۵۰، مطابق شکل ۴-۹ ایجاد کنید.

همان‌طور که مشاهده می‌کنید یک توپ با کنترل شکل (Shape) و یک کنترل زمان‌سنج برای ایجاد حرکت در توپ استفاده شده است، خصوصیات آن‌ها را مطابق جدول ۵-۹ تنظیم کنید.

جدول ۵-۹

| کنترل / خصوصیت | Timer | Shape |
|----------------|----------|----------|
| Name | tmrtimer | shpball |
| BackColor | — | سیاه |
| BackStyle | — | 1-Opaque |
| Enabled | False | — |
| Interval | ۲۰ | — |

۲- در بخش تعاریف، رویداد KeyDown و Load فرم را به صورت زیر تنظیم کنید:

Dim intdirection As Byte

Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)

Select Case KeyCode

Case 13: tmrtimer.Enabled = True

Case 37: intdirection = 1

Case 38: intdirection = 2

Case 39: intdirection = 3

Case 40: intdirection = 4

End Select

Select Case Shift

Case vbCtrlMask:

If KeyCode=81 Then Unload Me

Case vbAltMask:

If KeyCode=80 Then tmrTimer.Enabled=False

Case vbShiftMask:

If KeyCode=83 Then tmrTimer.Enabled=True

End Select

End Sub

Private Sub Form_Load()

intdirection = 1

End Sub

همان‌طور که در رویداد KeyDown فرم مشاهده می‌کنید با استفاده از یک دستور Select Case کلید فشرده شده بررسی می‌شود. اگر مقدار آرگومان KeyCode برابر ۱۳ باشد به معنی فشرده شدن کلید Enter خواهد بود، در این صورت خصوصیت Enabled کنترل زمان‌سنج True می‌شود تا کنترل زمان‌سنج شروع به کار کند. به این ترتیب فشردن کلیدهای جهت‌دار، کدهای ۳۷ تا ۴۰ را تولید می‌کنند که متناسب با کلید فشرده شده متغیر عمومی intdirection مقداردهی می‌شود. از این متغیر برای ایجاد حرکت متناسب با کلید فشرده شده در رویه رویداد کنترل زمان‌سنج استفاده می‌شود. این متغیر در رویه رویداد Load فرم مقدار ۱ را به دست می‌آورد. اگر مقدار این متغیر ۱ باشد به معنی حرکت به چپ، مقدار ۲ به معنی حرکت به بالا، مقدار ۳ به معنی حرکت به راست و مقدار ۴ به معنی حرکت به پایین خواهد بود.

سپس برای مدیریت کلیدهای ترکیبی Ctrl+Q، Alt+P، و Shift+S از دستور Select استفاده می‌شود و با بررسی مقدار آرگومان Sift ابتدا فشرده شدن کلیدهای Ctrl، Alt و Shift بررسی می‌شود. برای این کار در هر Case از ثابت‌های مربوط به هر یک از این کلیدها استفاده شده است و برای تشخیص کلیدهای Q، P، و S همراه کلیدهای Ctrl، Alt و Shift در Case هر یک از آن‌ها یک دستور If قرار داده شده است تا با استفاده از آرگومان KeyCode کلید حرفی نیز شناسایی شود و در صورت درست بودن نتیجه بررسی دستور موردنظر اجرا شود.

۳ - در این مرحله رویداد Timer کنترل زمان‌سنج را به صورت زیر تنظیم کنید:

```
Private Sub tmrtimer_Timer()
```

```
    Select Case intdirection
```

```
        Case 1: shpball.Left = shpball.Left - 50
```

```
        Case 2: shpball.Top = shpball.Top - 50
```

```
        Case 3: shpball.Left = shpball.Left + 50
```

```
        Case 4: shpball.Top = shpball.Top + 50
```

```
    End Select
```

```
    If shpball.Left <= 18 Or shpball.Left >= 4319 Or _
```

```
        shpball.Top <= 22 Or shpball.Top >= 2724 Then
```

```
        tmrtimer.Enabled = False
```

```
        MsgBox "Game Over !!!!", vbCritical, "THE END"
```

```
    End If
```

```
End Sub
```

می‌بینید که با استفاده از یک دستور Select Case مقدار متغیر intdirection که در رویداد KeyDown با توجه به کلید فشرده شده مقداردهی شده است، کنترل می‌شود و با توجه به این مقدار حرکت مناسب برای توپ در نظر گرفته می‌شود. برای مثال اگر مقدار متغیر intdirection برابر ۱ باشد به این معنی است که کلید جهت‌دار چپ فشرده شده است، در نتیجه از خصوصیت Left کنترل شکل ۵۰ واحد کم می‌شود که این سبب حرکت توپ به سمت چپ خواهد شد و این کار تا زمانی که کاربر کلید دیگری از کلیدهای جهت‌دار را فشار ندهد، هر ۲۰ میلی‌ثانیه یک بار تکرار می‌شود. این امر سبب خواهد شد تا توپ با یک حرکت ملایم به سمت چپ حرکت کند.

به همین صورت وقتی کلید جهت‌دار راست فشرده شود Case سوم اجرا می‌شود و با اضافه کردن ۵۰ واحد به خصوصیت Left توپ، آن را به سمت راست فرم حرکت می‌دهد و اگر کلید جهت‌دار بالا یا پایین را فشار دهد به ترتیب Case دوم یا چهارم اجرا شده و با کاهش یا افزایش خصوصیت Top توپ، آن را به بالا یا پایین حرکت خواهد داد. در ضمن با استفاده از یک دستور If در هر بار مکان قرارگیری توپ بررسی می‌شود تا در صورت برخورد توپ با دیواره‌های فرم، کنترل زمان‌سنج از کار بیفتد و پیام Game Over

با استفاده از یک کادر پیام نشان داده شود.

۴ - پروژه و فرم را با نام ball ذخیره کرده، سپس آن را اجرا کنید.

۵ - کلید Enter را فشرده و بلافاصله با کلیدهای جهت‌دار، توپ را کنترل کنید، سپس

کلیدها را رها کنید تا توپ با دیوارهای فرم برخورد کرده و پیام GameOver نمایش داده شود.

۶ - به پنجره ویژوال بیسیک بازگردید و برای آن که زمان سپری شده نیز نمایش داده

شود یک کنترل زمان سنج دیگر روی فرم قرار دهید، سپس خصوصیت Interval آن را

روی ۱۰۰۰ میلی‌ثانیه و خصوصیت Name و Enabled آن را به ترتیب روی tmrClock و

False تنظیم کنید.

۷ - اکنون به رویداد KeyDown بروید و در اولین Case دستور Select Case

tmrClock.Enabled=True را تایپ کنید تا با شروع به کار زمان سنج اول، زمان سنج دوم

نیز شروع به کار کند.

۸ - دستور Dim lngTime As Long را در بخش تعاریف ماژول فرم و دستور

lngTime=lngTime+1 را در رویداد() tmrClock_Timer تایپ کنید.

۹ - به رویداد() tmrTimer_Timer بروید و دستور MsgBox را به صورت بعد تغییر دهید

تا در پایان بازی زمان به دست آمده نمایش داده شود.

MsgBox «Game Over ! your Time Is: " _

+Str(lngTime)+" Second.",vbCritical,"The End"

۱۰ - تغییرات ایجاد شده را ذخیره کنید و پروژه را مجدداً اجرا و آزمایش نمایید و

در پایان به اجرای پروژه خاتمه داده، به پنجره ویژوال بیسیک بازگردید.

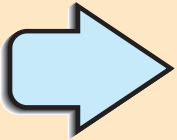
تمرین:



بازی ball را طوری تنظیم کنید که امکانات زیر به آن اضافه شود:

الف- امکان افزایش و کاهش سرعت حرکت توپ با کلیدهای + و - وجود داشته باشد.

ب- امکان توقف بازی با کلید Pause و شروع مجدد آن با کلید Esc وجود داشته باشد.



Learn in English

Responding to Mouse Events

You can use the `MouseDown`, `MouseUp`, and `MouseMove` events to enable your applications to respond to both the location and the state of the mouse. These mouse events are recognized by most controls.

| Event | Description |
|------------------------|---|
| <code>MouseDown</code> | Occurs when the user presses any mouse button. |
| <code>MouseUp</code> | Occurs when the user releases any mouse button. |
| <code>MouseMove</code> | Occurs each time the mouse pointer is moved to a new point on the screen. |

Dash
Dot
Middle
Pointer
Recognize
Responding
Shape
Solid
State
Transparent

واژه‌نامه
خط تیره
نقطه
میانی، وسط
اشاره‌گر
تشخیص دادن
پاسخ دادن
شکل
توپر
حالت، وضعیت
شفاف

خلاصه مطالب

- رویدادهای ماوس عبارتند از: MouseDown، MouseUp، MouseMove و DragDrop.
- رویداد MouseDown زمانی اجرا می‌شود که یکی از کلیدهای ماوس فشرده شود.
- رویداد MouseUp زمانی اجرا می‌شود که یکی از کلیدهای ماوس که فشرده شده است، رها شود.
- رویداد MouseMove زمانی اجرا می‌شود که اشاره‌گر ماوس روی فرم یا یک کنترل به حرکت درآید.
- رویداد DragDrop زمانی اجرا می‌شود که موقعیت یک کنترل روی فرم به وسیله ماوس تغییر کند.
- از کنترل Shape برای ایجاد اشکال هندسی و از کنترل Line برای ایجاد خط استفاده می‌شود.
- رویدادهای صفحه کلید عبارتند از: KeyDown، KeyUp و KeyPress.
- رویداد KeyDown و KeyPress زمانی اجرا می‌شوند که یک کلید در صفحه کلید فشرده شود.
- رویداد KeyUp زمانی اجرا می‌شود که یک کلید فشرده شده در صفحه کلید رها شود.
- خصوصیت KeyPreview نحوه اجرای رویدادهای صفحه کلید را بین فرم و کنترل آن تعیین می‌کند.

آزمون نظری

۱- کدام رویداد در صفحه کلید بین کاراکترهای حرفی کوچک و بزرگ تفاوت قائل می‌شود؟

الف - KeyDown ب - KeyPress ج - KeyUp د - KeyPreview

۲ - کدام خصوصیت در کنترل Shape, نوع شکل را معین می‌کند؟

الف - BorderStyle ب - Style ج - Shape د - Appearance

۳ - برای استفاده از رویداد DragDrop تنظیم کدام خصوصیت در کنترل الزامی است؟

الف - DragIcon ب - DragMode ج - DragDrop د - DropMode

۴ - کدام خصوصیت در کنترل Line نوع خط رسم شده را تعیین می‌کند؟

الف - Border Style ب - Border Width ج - Border Color د - Draw Mode

۵ - کدام گزینه در رابطه با رسم یک نقطه در روی فرم مناسب است؟

الف - Point ب - PSet ج - PointSet د - Step

۶ - کدام آرگومان در رویداد DragDrop, کنترل جابه‌جا شده را مشخص می‌کند؟

الف - Control ب - Source ج - Shift د - Ctrl

۷ - واحد اندازه‌گیری پیش‌فرض در کنترل Shape چیست؟

الف - سانتی‌متر ب - اینچ ج - Twip د - Pixel

۸ - در صورتی که خصوصیت KeyPreview روی مقدار True تنظیم شود

الف - رویدادهای صفحه کلید فرم زودتر از کنترل‌ها اجرا می‌شوند.

ب - رویدادهای صفحه کلید کنترل زودتر از فرم اجرا می‌شوند.

ج - فقط رویدادهای صفحه کلید کنترل‌ها اجرا می‌شوند.

د - فقط رویدادهای صفحه کلید فرم اجرا می‌شوند.

۹ - کدام آرگومان در رویداد KeyUp کداسکی کلید فشرده شده را مشخص می‌کند؟

الف - KeyCode ب - Key ج - Shift د - Keys

۱۰ - کدام آرگومان در رویداد KeyDown می‌تواند فشرده شدن کلیدهای تغییر حالت

مانند Alt را مشخص کند؟

الف - Shift ب - Alt ج - Control د - KeyCode

۱۱ - Which of the following mouse events can be used to respond mouse button clicks?

a- MouseUp

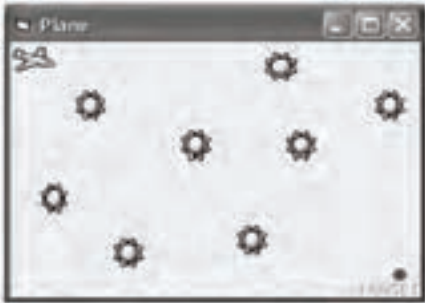
b- MouseDown

c- MouseMove

d- DragDrop

- ۱۲- تفاوت بین رویدادهای صفحه کلید KeyDown و Keypress را توضیح دهید.
- ۱۳- رویدادهای ماوس را نام برده و هر یک را همراه با آرگومان‌های مربوطه به اختصار شرح دهید.
- ۱۴- نقش و کاربرد آرگومان‌های KeyCode و Shift را در رویدادهای KeyDown و KeyUp بیان کنید.
- ۱۵- وظیفه آرگومان KeyAscii را در رویداد KeyPress توضیح دهید.
- ۱۶- رویداد DragDrop را همراه با آرگومان‌های آن توضیح دهید.
- ۱۷- وظیفه آرگومان‌های Button و Shift را در رویدادهای ماوس مشخص کنید.

آزمون عملی



- ۱- یک بازی طراحی کنید که مطابق شکل زیر، بازیکن یک هواپیمای متحرک را از میان موانع و بدون برخورد با آن‌ها عبور داده به مقصد برساند و بر اساس زمان تلف شده امتیاز وی محاسبه شود، به علاوه در صورت برخورد وی با دو مانع ادامه بازی امکان‌پذیر نباشد.

- ۲- پروژه‌ای طراحی کنید که مطابق شکل بعد حرکت یک زیردریایی در زیر آب را شبیه‌سازی نماید و علاوه بر امکان حرکت زیردریایی با صفحه کلید، بتوان از پریسکوپ زیردریایی در نزدیکی سطح آب نیز استفاده کرد و در ضمن خورشید و ابرهای موجود در آسمان نیز به صورت متحرک باشند و شرایط طبیعی مثل روز و شب را پدید آورند.



نحوه ایجاد انواع منو در ویژوال بیسیک

هدف‌های رفتاری

- ۱- انواع منوها را بیان کند.
- ۲- انواع منوها را ایجاد کند و بتواند از رویدادهای منوها استفاده کند.
- ۳- بتواند از کنترل‌های نوار پیمایش افقی و عمودی استفاده کرده، رویدادها و خصوصیات آن‌ها را به‌کاربرد.
- ۴- تعریف رابط گرافیکی MDI و SDI و تفاوت‌های آن‌ها را بیان کند.
- ۵- بتواند رابط گرافیکی MDI را ایجاد کند.
- ۶- نحوه ایجاد کادر محاوره باز کردن فایل را بداند.

کلیات

یکی از مهم‌ترین اجزای تشکیل دهنده برنامه‌هایی که برای سیستم عامل ویندوز تهیه می‌شوند نوارهای منو و گزینه‌های موجود در آن‌هاست. نوارهای منو و گزینه‌های موجود در آن دسترسی کاربر به امکانات برنامه را آسان‌تر کرده و شکل ظاهری مناسبی را به رابط گرافیکی می‌دهند تا کاربر با انتخاب گزینه‌های مورد نظر خود، عملیات مربوطه را انجام دهد. منوها دارای انواع مختلفی هستند، در شکل ۱-۱۰ نمونه‌هایی از آن‌ها را مشاهده می‌کنید.



شکل ۱-۱۰

۱۰-۱ نحوه طراحی انواع منو در ویژوال بیسیک

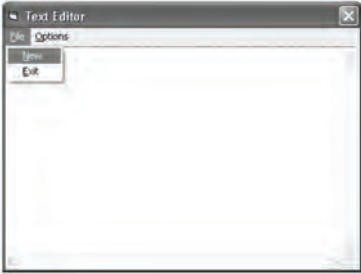
برای ایجاد منوها در فرم‌های خود می‌توانید از ابزار Menu Editor در پنجره ویژوال بیسیک استفاده کنید، برای اجرای این برنامه می‌توانید یکی از این روش‌ها را انتخاب کنید:

- ۱- در نوار ابزار استاندارد ویژوال بیسیک روی دکمه Menu Editor کلیک کنید.
- ۲- گزینه Menu Editor را از منوی Tools انتخاب نمایید.

۳ - از کلید ترکیبی Ctrl+E استفاده کنید.



مثال ۱: یک واژه پرداز مطابق شکل‌های ۱۰-۲ الی ۱۰-۴ و جداول ۱۰-۱ و ۱۰-۲ طراحی کنید تا کاربر توانایی انتخاب قالب‌بندی‌های مناسب را برای متن تایپ شده داشته باشد، به این منظور برنامه ویژوال بیسیک را اجرا کنید و عملیات بعد را به ترتیب انجام دهید:



شکل ۱۰-۲



شکل ۱۰-۳



شکل ۱۰-۴

جدول ۱۰-۱ خصوصیات فرم

| خصوصیت | مقدار |
|---------|-------------|
| Name | frmeditor |
| Caption | Text Editor |

جدول ۱۰-۲ خصوصیات کنترل

| کنترل | TextBox |
|------------|---------|
| Name | txttext |
| MultiLine | True |
| ScrollBars | Both-۳ |




شکل ۱۰-۵

۱ - یک پروژه از نوع Standard EXE ایجاد کنید که حاوی یک فرم با یک کنترل کادر متن مطابق شکل ۱۰-۵ باشد.

در شکل‌های ۱۰-۲ الی ۱۰-۴ ملاحظه می‌کنید، که نوار منوی فرم دارای دو منوی File و Options است. در منوی File دو گزینه وجود دارد، گزینه New که برای تایپ یک متن جدید در کادر متن استفاده می‌شود و گزینه Exit که برای خروج از برنامه به کار می‌رود.


منوی دوم یعنی Options نیز از گزینه‌های Font Size و Font Color تشکیل شده است که هر یک شامل گزینه‌هایی برای تنظیم اندازه و رنگ قلم هستند با کلیک روی اندازه‌ها یا رنگ‌ها، متن موجود در کادر متن به طور خودکار تنظیم می‌شود.

۲- برای ایجاد منو در روی فرم، در نوار ابزار استاندارد ویژوال بیسیک روی دکمه  Menu Editor کلیک کنید تا کادر محاوره Menu Editor نمایش داده شود. برای فعال کردن این کادر محاوره می‌توانید گزینه Menu Editor... را از منوی Tools برگزینید یا از کلید ترکیبی Ctrl+E استفاده کنید.

۳- برای ایجاد اولین منو، در کادر متن عنوان منو (Caption) عبارت File& را تایپ کنید، کاراکتر & سبب می‌شود کلید ترکیبی Alt+F برای باز کردن منو در نظر گرفته شود، به این نوع از کلید ترکیبی، کلید دسترسی سریع یا Hot Key گفته می‌شود. سپس در کادر متن نام منو (Name) عبارت mnufile را تایپ کنید. منوها و گزینه‌های موجود در آن‌ها نیز مانند فرم‌ها و کنترل‌ها دارای خصوصیت Name هستند و قوانین نام‌گذاری آن‌ها مانند فرم‌ها و کنترل‌هاست. مطابق شکل ۶-۱۰ عنوان منو در کادر لیست موجود در قسمت پایین کادر محاوره Menu Editor دیده می‌شود.





شکل ۶-۱۰


۴- برای ایجاد گزینه New داخل منوی File، ابتدا روی دکمه Next در کادر محاوره Menu Editor کلیک کنید، سپس روی دکمه  در کادر محاوره کلیک کنید. در کادر لیست عناوین منوها و در زیر عبارت File&، چهار کاراکتر نقطه ظاهر شده‌اند، این عمل سبب می‌شود که ویژوال بیسیک گزینه New را در منوی File نمایش دهد (شکل ۷-۱۰).



شکل ۷-۱۰

۵- عبارات New& و mnunew را به ترتیب در کادرهای متن عنوان و نام کادر محاوره Menu Editor تایپ کنید. مجدداً روی دکمه Next کلیک کنید و به همین صورت گزینه Exit را در منوی File ایجاد کنید. به علاوه با استفاده از دکمه‌های  و  در کادر محاوره Menu Editor می‌توانید گزینه‌ها را جابه‌جا کنید و به وسیله دکمه Delete در کادر محاوره Menu Editor گزینه مورد نظر خود را حذف نمایید.

۶- روی دکمه OK کلیک کنید و به پنجره طراحی فرم باز گردید. در پنجره طراحی روی منوی File کلیک کنید، می‌بینید منویی را که طراحی کرده‌اید در پنجره طراحی نیز قابل مشاهده است (شکل ۸-۱۰).


۷- مجدداً به کادر محاوره Menu Editor بروید، سپس در کادر لیست عناوین منوها روی گزینه Exit& کلیک کرده، بعد روی دکمه Next سپس دکمه  کلیک کنید.

۸- در کادرهای متن عنوان و نام منو عبارات Options& و mnuoptions را تایپ کنید.



شکل ۸-۱۰

۹- مانند گزینه New، گزینه‌های Font Size و Font Color را مطابق شکل ۹-۱۰ در منوی Options بسازید.

۱۰- در این مرحله گزینه‌های موجود در زیر منوهای، منوی Options را ایجاد کنید. بنابراین در کادر لیست عناوین منوها ابتدا روی گزینه Font & Color و بعد روی دکمه Insert و دکمه  کلیک کنید.

۱۱ - در کادر متن عنوان و نام منو به ترتیب عبارات 10 و mnu10 را تایپ کنید و به همین صورت گزینه‌های ۱۲ و ۱۴ را ایجاد کنید.

چون می‌خواهید گزینه‌های زیر منوی Font Size دارای چک مارک باشند و گزینه انتخاب شده در این زیر منو با چک مارک نمایش داده شود، گزینه ۱۲ را در کادر اسامی منوها انتخاب کرده، سپس در کادر علامت Checked در کادر محاوره Menu Editor کلیک کنید (شکل ۹-۱۰). این کادر علامت می‌تواند گزینه‌ها را به همراه یک چک مارک نمایش دهد. از این گزینه‌ها زمانی استفاده می‌شود که کاربر بخواهد از چند گزینه موجود در یک منو یا زیر منو، یکی را انتخاب نماید.

۱۲ - مانند مرحله ۱۱ گزینه‌های Blue، Green و Red را در زیر منوی Font Color ایجاد کنید. در این جا می‌خواهید برای این گزینه‌ها کلیدهای ترکیبی تعریف کنید، بنابراین روی گزینه Blue& در کادر لیست عناوین منوها کلیک کنید، سپس از کادر لیست Shortcut، گزینه Ctrl+B را برگزینید و به همین شکل کلید ترکیبی Ctrl+G و Ctrl+R را برای گزینه‌های Green و Red انتخاب کنید. در ضمن چون می‌خواهید به طور پیش فرض رنگ متن آبی باشد، بنابراین گزینه Blue را غیرفعال کنید. به این منظور در کادر لیست عناوین منوها گزینه Blue& را برگزینید و روی کادر علامت Enabled کلیک کنید (شکل ۹-۱۰). این کادر علامت اجازه می‌دهد تا گزینه‌های مورد نظر خود را فعال یا غیر فعال کنید. در صورتی که این کادر علامت انتخاب شود گزینه مربوطه قابل استفاده خواهد بود و در غیر این صورت امکان کلیک روی گزینه ممکن نیست، روی دکمه OK کلیک کنید و در پنجره طراحی فرم گزینه‌های منویی را که ایجاد کرده‌اید، بررسی نمایید.

در ضمن برای مخفی کردن منوها و گزینه‌های موجود در آن‌ها می‌توانید از کادر علامت Visible در کادر محاوره Menu Editor استفاده کنید، در صورتی که این کادر علامت از حالت انتخاب خارج شود منو یا گزینه مورد نظر مخفی خواهد شد.



شکل ۹-۱۰

۱۳ - اکنون در منوی File یک خط جداکننده بین گزینه‌های New و Exit ایجاد کنید. برای این کار کافی است یک گزینه جدید بین گزینه‌های New و Exit در منوی File ایجاد کرده و برای عنوان آن از کاراکتر تفریق (-) استفاده کنید (شکل ۹-۱۰).

۱۴ - در این مرحله رویدادهای مربوط به هر گزینه را تنظیم کنید. برای این کار کافی است در پنجره طراحی فرم روی گزینه مورد نظر کلیک کنید یا در ماژول فرم رویداد Click گزینه‌ها را انتخاب کنید. به پنجره ماژول فرم بروید و رویداد Click گزینه‌های منوی Font Size را به صورت زیر تنظیم کنید:

```
Private Sub mnu10_Click()
```

```
    mnu10.Checked = True  
    mnu12.Checked = False  
    mnu14.Checked = False  
    txttext.FontSize = 10
```

```
End Sub
```

```
Private Sub mnu12_Click()
```

```
    mnu10.Checked = False  
    mnu12.Checked = True  
    mnu14.Checked = False  
    txttext.FontSize = 12
```

```
End Sub
```

```
Private Sub mnu14_Click()
```

```
    mnu10.Checked = False  
    mnu12.Checked = False  
    mnu14.Checked = True  
    txttext.FontSize = 14
```

```
End Sub
```

از رویداد Click گزینه‌های mnu10، mnu12 و mnu14 استفاده شده است. در هر رویداد با توجه به گزینه انتخاب شده خصوصیت Checked آن روی True تنظیم می‌شود، این امر سبب می‌شود چک مارک در کنار گزینه مورد نظر نمایش داده شود. به علاوه خصوصیت Checked دو گزینه دیگر False خواهد شد و با استفاده از خصوصیت FontSize اندازه قلم متن با توجه به گزینه انتخاب شده تنظیم می‌شود.

۱۵- اکنون رویدادهای مربوط به گزینه‌های زیر منوی FontColor را به این صورت تنظیم کنید:

```
Private Sub mnubblue_Click()
```

```
    mnubblue.Enabled = False  
    mnugreen.Enabled = True  
    mnured.Enabled = True  
    txttext.ForeColor = vbBlue
```

```
End Sub
```

```
Private Sub mnugreen_Click()
```

```
    mnubblue.Enabled = True  
    mnugreen.Enabled = False  
    mnured.Enabled = True  
    txttext.ForeColor = vbGreen
```

```
End Sub
```

```
Private Sub mnured_Click()
```

```
    mnubblue.Enabled = True  
    mnugreen.Enabled = True  
    mnured.Enabled = False  
    txttext.ForeColor = vbRed
```

```
End Sub
```

```
Private Sub Form_Load()
```

```
    Call mnu12_Click  
    Call mnubblue_Click
```

```
End Sub
```

در این رویدادها با استفاده از خصوصیت Enabled گزینه‌ها و با توجه به گزینه انتخاب شده، مقدار این خصوصیت روی False تنظیم می‌شود تا گزینه انتخاب شده غیرفعال شود، زیرا با تغییر رنگ قلم به رنگ مربوطه، انتخاب مجدد همان گزینه لازم نیست، اما برای دو گزینه دیگر این خصوصیت روی True تنظیم می‌شود تا گزینه‌ها برای کاربر قابل انتخاب شوند.

به علاوه با استفاده از خصوصیت ForeColor کادر متن، رنگ متن تنظیم می‌شود و برای هماهنگی بیشتر، در رویداد Load فرم رویه رویدادهای mun12_Click و mnublu_Click فراخوانی می‌شوند تا اندازه و رنگ قلم روی مقدار ۱۲ و آبی تنظیم شوند. بنابراین هنگام اجرای برنامه اندازه و رنگ متن با گزینه‌های مربوطه در منو یکسان می‌شوند.

۱۶- پروژه و فرم را با نام texteditor ذخیره کنید، سپس آن را اجرا نمایید و نام و نام خانوادگی خود را در کادر متن بنویسید. مشاهده می‌کنید که رنگ قلم آبی و اندازه آن ۱۲ است.

۱۷- روی گزینه ۱۴ و سپس ۱۰ در منوی Font Size کلیک کنید و نتیجه را بررسی کنید، سپس روی گزینه‌های Green و Red در منوی Font Color کلیک کنید و نتیجه را بررسی کنید.

۱۸- به اجرای پروژه خاتمه دهید و به پنجره ویژوال بیسیک بازگردید.



تمرین:

پروژه فوق را طوری تنظیم کنید که با انتخاب گزینه New محتویات کادر متن حذف شود و با انتخاب گزینه Exit برنامه واژه‌پرداز خاتمه یابد. به علاوه دارای این شرایط باشد:

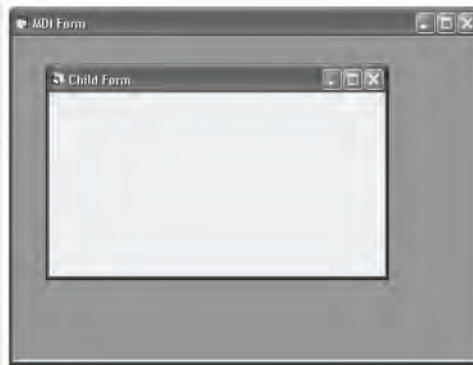
الف- یک منوی جدید با عنوان Edit به نوار منو اضافه کنید که امکان جستجو و جای‌گزینی عبارت مورد نظر را در متن داشته باشد.

ب- یک گزینه به منوی Options اضافه کنید تا مانند گزینه Font Color بتواند رنگ زمینه متن را تغییر دهد.

مطالعه آزاد

۲-۱۰ نحوه ایجاد و استفاده از رابط گرافیکی چند سندی یا MDI^۱

تاکنون پروژه‌هایی را که طراحی کرده‌اید از یک یا چند فرم تشکیل می‌شد که هر یک به طور مستقل روی دسک‌تاپ به نمایش درآمده و مورد استفاده قرار می‌گرفتند و بسته شدن، انتقال، به حداکثر رسانی یا به حداقل رسانی یک فرم تأثیری روی سایر فرم‌های برنامه نمی‌گذاشت، به این گونه برنامه‌ها رابط گرافیکی تک سندی یا فرم‌های SDI^۲ گفته می‌شود. اما ممکن است برنامه‌هایی مانند ویژوال بیسیک، Word، Excel یا انواع مشابهی را دیده باشید که در آن‌ها یک پنجره به عنوان پنجره اصلی وجود دارد و سایر فرم‌ها و پنجره‌ها در داخل این پنجره باز می‌شوند و به پنجره اصلی وابستگی دارند که در صورت انتقال یا بسته شدن پنجره اصلی، تمام پنجره‌های موجود در آن منتقل یا بسته می‌شوند، اما بسته شدن یا انتقال پنجره‌هایی که به پنجره اصلی وابسته هستند تأثیری روی پنجره اصلی برنامه نمی‌گذارد به این گونه از پنجره‌ها، رابط گرافیکی چند سندی یا MDI گفته می‌شود و به پنجره اصلی پروژه فرم والد (Parent) یا MDI و به پنجره‌ها و فرم‌های وابسته، فرم فرزند (Child) گفته می‌شود (شکل ۱۰-۱۰).



شکل ۱۰-۱۰


یک پروژه می‌تواند فقط یک فرم MDI داشته باشد، اما استفاده از چند فرم Child در یک پروژه بلامانع است. در ضمن استفاده از بعضی کنترل‌ها مانند دکمه فرمان، کادر متن، کادر تصویر و نظایر آن‌ها در روی فرم‌های MDI امکان‌پذیر نیست.

۱- رابط کاربر چند سندی (MDI (Multi Document Interface)
۲- رابط کاربر یک سندی (SDI (Single Document Interface)



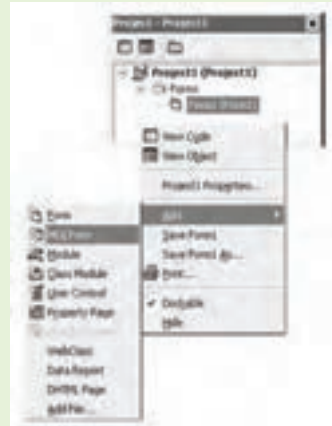
مثال ۲: پروژه texteditor را طوری تغییر دهید که مطابق شکل ۱۱-۱۰ به صورت یک رابط گرافیکی MDI مورد استفاده قرار گیرد و تایپ متن در یک فرم SDI انجام شود. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱- برنامه ویژوال بیسیک را اجرا کنید و یک پروژه از نوع Standard EXE ایجاد کنید.
۲- در پنجره پروژه کلیک راست کنید و از زیرمنوی Add گزینه MDI Form را انتخاب کنید (شکل ۱۲-۱۰) تا یک فرم از نوع MDI ایجاد شود. در این صورت کادر محاوره Add MDI Form (مطابق شکل ۱۳-۱۰) نمایش داده می‌شود. روی دکمه Open کلیک کنید تا یک فرم MDI به پروژه اضافه شود (شکل ۱۴-۱۰). برای ایجاد یک فرم از نوع MDI می‌توانید یکی از روش‌های زیر را نیز انتخاب کنید:

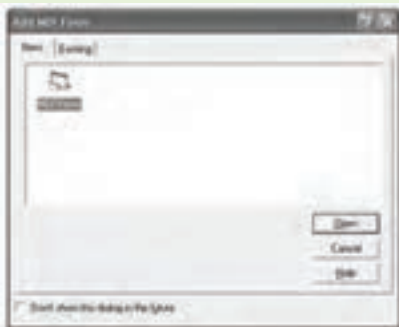
الف- از منوی Project در پنجره ویژوال بیسیک گزینه Add MDI Form را انتخاب کنید.
ب- روی علامت مثلث دکمه Add Form  در نوار ابزار استاندارد کلیک کرده و گزینه MDI Form را انتخاب کنید.



شکل ۱۱-۱۰



شکل ۱۲-۱۰




شکل ۱۳-۱۰



شکل ۱۴-۱۰

۳ - در پنجره پروژه، آیکن Form1 را انتخاب کرده و روی آن کلیک راست کنید و از منویی که ظاهر می‌شود، گزینه Remove Form1 را برگزینید تا فرم از پروژه خارج شود. برای خارج کردن یک فرم از پروژه می‌توانید گزینه (نام فرم Remove) را از منوی Project در پنجره ویژوال بیسیک انتخاب کنید.

 نکته در صورت خارج کردن یک فرم از پروژه، فایل فرم از روی دیسک حذف نمی‌شود.

۴ - مشخصات فرم MDI را مطابق جدول ۳-۱۰ تنظیم کنید، سپس فرم و پروژه را با نام neweditor ذخیره کنید.

جدول ۳-۱۰ خصوصیات فرم

| مقدار | خصوصیت |
|-------------|---------|
| mdieditor | Name |
| Text Editor | Caption |

۵ - در این مرحله فرم text editor.frm را که در مثال قبل ایجاد کرده‌اید به پروژه جاری اضافه کنید. سپس به پنجره طراحی این فرم بروید و کادر محاوره Menu Editor را فعال کنید و روی دکمه Delete کلیک کنید. گزینه File از کادر لیست اسامی عناوین منوها حذف می‌شود. به همین روش تمام گزینه‌ها را حذف کرده و در پایان روی دکمه OK کلیک کنید.

۶ - در این مرحله باید بین فرم‌های MDI و frmeditor ارتباط لازم را برقرار کنید تا فرم frmeditor به فرم MDI وابسته شود. به این منظور در پنجره پروژه روی آیکن فرم frmeditor کلیک کنید، سپس در پنجره خصوصیات، خصوصیت MDIChild را انتخاب کنید و مقدار آن را روی True تنظیم کنید.

۷ - همان‌طور که می‌بینید نوار منو از فرم Text Editor حذف شده است. نام فرم حاصل را روی frmtext تنظیم کنید سپس آن را با نام text ذخیره کنید.

۸ - اکنون به پنجره طراحی فرم MDI بروید و منویی را که از فرم قبلی حذف کرده‌اید مجدداً به همان شکل ایجاد کنید.

۹ - به پنجره ماژول فرم frmtext بروید و تمام رویدادهای مربوط به منوها را انتخاب کنید و به ماژول فرم MDI انتقال دهید (ابتدا عمل Cut و بعد در ماژول فرم MDI عمل Paste انجام دهید)، سپس رویداد Load فرم MDI و سایر رویدادها را به صورت زیر تنظیم کنید:

```
Private Sub MDIForm_Load()
```

```
    mdieditor.Width = 8000  
    mdieditor.Height = 7000  
    frmtext.Top = 500  
    frmtext.Left = 300  
    frmtext.Show  
    frmtext.txttext.FontSize = 12  
    frmtext.txttext.ForeColor = vbBlue
```

```
End Sub
```

```
Private Sub mnu10_Click()
```

```
    mnu10.Checked = True  
    mnu12.Checked = False  
    mnu14.Checked = False  
    frmtext.txttext.FontSize = 10
```

```
End Sub
```

```
Private Sub mnu12_Click()
```

```
    mnu10.Checked = False  
    mnu12.Checked = True  
    mnu14.Checked = False  
    frmtext.txttext.FontSize = 12
```

```
End Sub
```

```
Private Sub mnu14_Click()
```

```
    mnu10.Checked = False  
    mnu12.Checked = False  
    mnu14.Checked = True  
    frmtext.txttext.FontSize = 14
```

```
End Sub
```



```
Private Sub mnublu_Click()
```

```
    mnublu.Enabled = False  
    mnugreen.Enabled = True
```

```
mnured.Enabled = True
frmtext.txttext.ForeColor = vbBlue
End Sub
Private Sub mnugreen_Click()
    mnubblue.Enabled = True
    mnugreen.Enabled = False
    mnured.Enabled = True
    frmtext.txttext.ForeColor = vbGreen
End Sub
Private Sub mnured_Click()
    mnubblue.Enabled = True
    mnugreen.Enabled = True
    mnured.Enabled = False
    frmtext.txttext.ForeColor = vbRed
End Sub
Private Sub mnunew_Click()
    frmtext.txttext.Text = ""
End Sub
Private Sub mnuexit_Click()
    Unload Me
End Sub
```

۱۰- به پنجره ویژگی های پروژه بروید و فرم mdieditor را به عنوان اولین فرم برای نمایش در ابتدای اجرای پروژه انتخاب کنید، سپس تغییرات را ذخیره کنید و پروژه را اجرا نمایید.

۱۱ - پنجره های پروژه مطابق شکل ۱۱-۱۰ نمایش داده می شوند، سعی کنید پنجره فرزند را از محدوده فرم MDI خارج کنید. همان طور که انتظار می رود فرم فرزند به فرم MDI وابسته شده است.

۱۲ - روی دکمه  فرم فرزند کلیک کنید، مشاهده می کنید که فرم MDI بسته نمی شود. به اجرای پروژه خاتمه داده و مجدداً آن را اجرا کنید و این بار روی دکمه  پنجره MDI کلیک کنید. خواهید دید که پنجره فرزند نیز با پنجره MDI بسته می شود.

۱۳ - اجرای پروژه را متوقف کنید و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه قبل را طوری تنظیم کنید که در هنگام اجرای برنامه، فقط فرم MDI به همراه منوی File مشاهده شود و فرم فرزند به همراه سایر گزینه ها زمانی فعال شود که گزینه New از منوی File انتخاب شده و فرم فرزند فاقد دکمه Close و منوی کنترل باشد.

۳-۱۰ کنترل‌های نوار پیمایش افقی و عمودی

تاکنون از نوارهای پیمایش در ویندوز و پنجره‌های برنامه استفاده کرده‌اید و می‌دانید از این اجزا زمانی استفاده می‌شود که محتویات مورد نیاز برای نمایش در پنجره از ابعاد پنجره بزرگ‌تر و بیشتر بوده و کاربر با به‌کارگیری نوار پیمایش، محتویات پنجره را مرور کند. به عنوان نمونه کاملاً مشخصی از نوار پیمایش، می‌توان به پنجره My Computer اشاره کرد که وقتی تعداد فایل‌ها و پوشه‌ها بیش از اندازه زیاد باشد؛ این نوارهای پیمایش هستند که به کاربر در مشاهده محتویات یک پوشه کمک می‌کنند یا نوارهای پیمایش در کادرهای متن و نظایر آن‌ها را می‌توان نام برد.

می‌دانید که این کنترل‌ها از دو دکمه مثلثی شکل و یک دکمه مستطیل شکل متحرک در بین دکمه‌های مثلثی تشکیل شده‌اند. کاربر می‌تواند با کلیک روی دکمه‌های مثلثی، کشیدن دکمه مستطیل شکل یا کلیک روی نقطه‌ای خالی از کنترل نوار پیمایش، حرکت کند. در ویژوال بیسیک دو کنترل نوار پیمایش عمودی (Vertical ScrollBar) و نوار پیمایش افقی (Horizontal ScrollBar) برای استفاده در پروژه‌های برنامه‌نویسی وجود دارند. در این جا خصوصیت‌ها و رویدادهایی از این کنترل‌ها که مشترک هستند، بررسی می‌شود.

خصوصیات SmallChange و LargeChange

این دو خصوصیت مقدار تغییرات را هنگام استفاده از کنترل تعیین می‌کنند. خصوصیت SmallChange مقدار تغییرات را هنگامی که روی دکمه‌های مثلثی کنترل نوار پیمایش کلیک شود، تعیین می‌کند و خصوصیت LargeChange مقدار تغییرات را هنگامی که کاربر روی مکانی از نوار پیمایش بجز دکمه‌ها کلیک می‌کند، معین می‌نماید. این دو خصوصیت از نوع عددی هستند.

خصوصیات Min و Max

این دو خصوصیت مقدار حداکثر را که با رسیدن دکمه متحرک کنترل به انتهای نوار پیمایش و مقدار حداقل را که با رسیدن دکمه متحرک به ابتدای نوار پیمایش به دست می‌آید، تعیین می‌کنند. این دو خصوصیت نیز از نوع عددی صحیح هستند.

خصوصیت Value

این خصوصیت مقدار جاری را در نوار پیمایش معین می‌کند، این خصوصیت نیز از نوع عددی است.

رویداد Change

این رویداد زمانی رخ می‌دهد که مقدار خصوصیت Value کنترل تغییر کند. این مسأله وقتی اتفاق می‌افتد که روی دکمه‌های مثلثی شکل یا در مکان خالی روی نوار پیمایش کلیک کنید.

رویداد Scroll

این رویداد زمانی اجرا می‌شود که کاربر دکمه متحرک مستطیل شکل را در نوار پیمایش به وسیله ماوس جابه‌جا کند.



مثال ۳: یک برنامه برای مشاهده تصاویر ایجاد کنید که مطابق شکل ۱۵-۱۰ توانایی نمایش تصویر دلخواه را داشته باشد و در صورتی که تصویر بزرگ‌تر از اندازه فرم باشد بتواند با استفاده از نوارهای پیمایش تمام تصویر را مرور کند.

۱- یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم و کنترل تصویر (Image) با عرض و ارتفاع ۶۰۰۰ و ۵۰۰۰ با یک نوار منو مطابق شکل‌های ۱۵-۱۰ و ۱۶-۱۰ و جداول ۴-۱۰ و ۵-۱۰ باشد.



شکل ۱۶-۱۰



شکل ۱۵-۱۰

جدول ۴-۱۰ خصوصیات فرم

| خصوصیت | مقدار |
|---------|----------------|
| Name | frmeditor |
| Caption | Picture Viewer |

جدول ۵-۱۰ خصوصیات کنترل

| کنترل | TextBox |
|---------|---------|
| خصوصیت | Imgshow |
| Name | True |
| Stretch | |


۲- در این مرحله رویدادهای مربوط به گزینه‌های منوها را به این صورت تنظیم کنید:

```
Private Sub mnuopen_Click()  
    Dim txtpicture As String  
    txtpicture = InputBox("Enter Path and File Name :", "Open")  
    imgshow.Stretch = False  
    imgshow.Picture = LoadPicture(txtpicture)  
End Sub  
Private Sub mnuzoomin_Click()  
    imgshow.Stretch = True  
    imgshow.Width = imgshow.width * 2  
    imgshow.Height = imgshow.Height * 2  
End Sub  
Private Sub mnuzoomout_Click()  
    imgshow.Stretch = True  
    imgshow.Width = imgshow.width / 2  
    imgshow.Height = imgshow.Height / 2  
End Sub  
Private Sub mnunormal_Click()  
    imgshow.Stretch = False  
End Sub  
Private Sub mnuexit_Click()  
    Unload Me  
End Sub
```

همان‌طور که مشاهده می‌کنید در رویداد گزینه Open با استفاده از کادر ورود داده نام و مسیر فایل تصویر از کاربر دریافت می‌شود، سپس خصوصیت Stretch کنترل تصویر روی False تنظیم می‌شود تا تصویر با اندازه واقعی مشاهده شود و در ادامه تابع LoadPicture تصویر مورد نظر را در کنترل تصویر نمایش خواهد داد.

در رویداد گزینه‌های Zoom In و Zoom out ابتدا خصوصیت Stretch کنترل تصویر روی True تنظیم می‌شود تا امکان تغییر اندازه تصویر با تغییر اندازه کنترل امکان‌پذیر باشد. سپس با ضرب یا تقسیم عرض کنترل تصویر بر عدد ۲، حالت بزرگ‌نمایی زیادتر یا کمتر به وجود می‌آید.

رویداد گزینه Normal نیز می‌تواند با False کردن خصوصیت Stretch کنترل تصویر آن را مجدداً به حالت عادی خود باز گرداند.

۳- برای ایجاد نوار پیمایش عمودی در جعبه ابزار روی آیکن  VScrollBar دابل کلیک کنید، سپس روی آیکن  HScrollBar دابل کلیک کنید تا یک نوار پیمایش افقی نیز روی فرم ایجاد شود، سپس خصوصیات آن‌ها را مطابق جدول ۶-۱۰ تنظیم کنید.

جدول ۶-۱۰ خصوصیات کنترل‌ها

| کنترل \ خصوصیت | VScrollBar | HScrollBar |
|----------------|------------|------------|
| Name | vsbimage | hsbimage |
| LargeChange | ۱۰۰ | ۱۰۰ |
| SmallChange | ۲۰ | ۲۰ |

۴- اکنون رویدادهای Change و Scroll دو کنترل نوار پیمایش را به صورت زیر تنظیم نمایید:

```
Private Sub vsbimage_Change()  

    imgshow.Top = -vsbimage.Value  

End Sub  

Private Sub vsbimage_Scroll()  

    imgshow.Top = -vsbimage.Value  

End Sub  

Private Sub hsbimage_Change()  

    imgshow.Left = -hsbimage.Value  

End Sub  

Private Sub hsbimage_Scroll()  

    imgshow.Left = -hsbimage.Value  

End Sub
```

در این رویدادها مشاهده می‌کنید، برای آن‌که عمل پیمایش روی تصاویر امکان‌پذیر باشد، مقدار خصوصیت Value کنترل نوار پیمایش عمودی به خصوصیت Top کنترل تصویر نسبت داده شده است. این امر سبب می‌شود تا در صورت پیمایش به وسیله هر یک از روش‌های ممکن تصویر با توجه به مقدار خصوصیت‌های SmallChange و LargeChange به سمت بالا حرکت کند. استفاده از علامت منفی نیز به همین علت است در غیر این صورت با انجام عمل پیمایش، تصویر به پایین حرکت نخواهد کرد. به همین شکل کنترل نوار پیمایش افقی نیز با خصوصیت Left کنترل تصویر هماهنگ می‌شود تا با حرکت در نوار پیمایش افقی به سمت جلو، قسمت‌های جلوتر تصویر قابل مشاهده شود.

۵ - در این مرحله دستورات زیر را به رویداد Click گزینه‌های Open, Zoom, Zoom In, Zoom Out و Normal اضافه کنید. این دستورات سبب می‌شوند تا حداکثر میزان حرکت در نوارهای پیمایش (خصوصیت Max کنترل‌های نوار پیمایش) با توجه به ابعاد تصویر تنظیم شوند و با تغییر در میزان بزرگ‌نمایی یا باز شدن تصویر جدید، مقدار خصوصیت نوارهای پیمایش صفر شده تا تصویر از بالاترین بخش آن قابل رؤیت باشد.

```
hsbimage.Max = imgshow.Width
```

```
vsbimage.Max = imgshow.Height
```

```
hsbimage.Value = 0
```

```
vsbimage.Value = 0
```

۶ - پروژه و فرم را با نام Picture Viewer ذخیره کرده، سپس آن را اجرا کنید و روی گزینه Open در منوی File کلیک کرده، نام و مسیر یک فایل تصویر از نوع BMP یا JPG را در کادر ورود داده تایپ کنید سپس روی دکمه OK کلیک کنید.

۷ - گزینه‌های منوی View را آزمایش کنید و با استفاده از نوارهای پیمایش، تصویر را مرور کنید. سپس تصویر دیگری را باز کرده، مراحل ۶ و ۷ را مجدداً تکرار نمایید.

تمرین:



امکانات زیر را به برنامه Picture Viewer اضافه کنید.

الف - در صورتی که ابعاد تصویر کوچک‌تر از ابعاد فرم باشد، نوارهای پیمایش مخفی می‌شوند.

ب - گزینه‌هایی در منوی View اضافه کنید که به وسیله آن بتوان نوارهای پیمایش را مخفی و آشکار کرد.



مثال ۴: پروژه Picture Viewer را به گونه‌ای تنظیم کنید که بتوان با کلیک راست روی تصویر نیز به گزینه‌های منوی View دسترسی پیدا کرد (شکل ۱۷-۱۰). برای این کار عملیات بعد را به ترتیب انجام دهید:



شکل ۱۷-۱۰

۱- پروژه Picture Viewer را باز کنید و به ماژول فرم و رویداد imgshow_MouseDown بروید.

۲- سپس این رویداد را به صورت زیر تنظیم کنید:

```
Private Sub imgshow_MouseDown(Button As Integer, _  
Shift As Integer, X As Single, Y As Single)
```

```
If Button = vbRightButton Then PopupMenu mnview
```

```
End Sub
```

مشاهده می‌کنید که با استفاده از دستور PopupMenu می‌توانید منوهای موضوعی خود را با استفاده از کلیک راست ایجاد کنید. در این جا با استفاده از رویداد MouseDown و یک دستور If، فشرده شدن دکمه راست ماوس بررسی می‌شود؛ سپس دستور PopupMenu گزینه‌های منوی View را در زمان کلیک راست روی کنترل تصویر در اختیار شما قرار می‌دهد.

۳- تغییرات را ذخیره کرده، پروژه را اجرا کنید و یک تصویر دلخواه را نمایش دهید، سپس روی تصویر کلیک راست کنید.

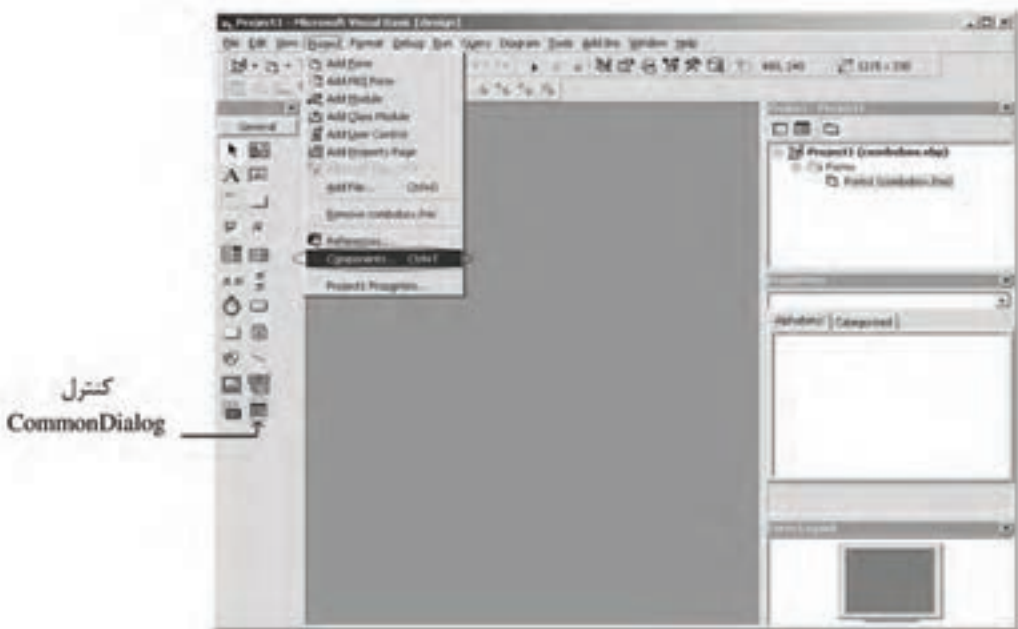
مطابق شکل ۱۷-۱۰ منوی View روی فرم ظاهر می‌شود، اکنون با استفاده از این منو عملکرد برنامه را مجدداً بررسی کنید.

۴- به اجرای پروژه خاتمه داده و به پنجره ویژوال بیسیک بازگردید.

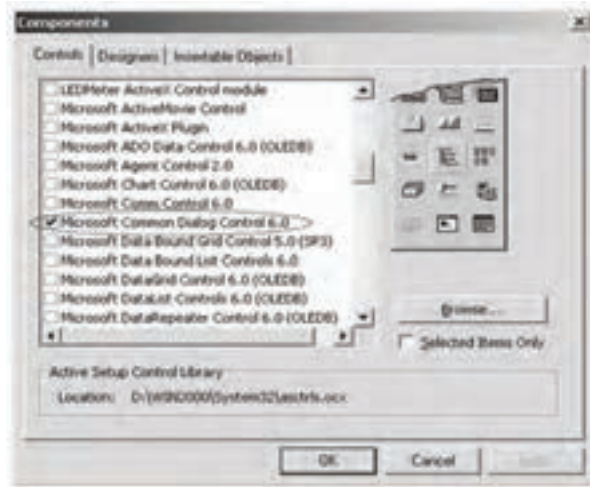
۴-۱۰ کنترل کادر محاوره (CommonDialog)

این کنترل امکان ایجاد کادرهای محاوره برای بازکردن و ذخیره‌سازی فایل‌ها، انجام عملیات چاپ، انتخاب رنگ‌ها و فونت‌ها و نمایش راهنما را فراهم می‌آورد. آیکن این کنترل به‌طور پیش‌فرض در جعبه ابزار دیده نمی‌شود؛ به منظور افزودن این کنترل به جعبه ابزار به ترتیب زیر عمل کنید:

- ۱ - روی منوی Project در نوار منوی ویژوال بیسیک کلیک کنید.
- ۲ - در منوی Project روی گزینه Components کلیک کنید.
- ۳ - پس از باز شدن پنجره Components روی زبانه Controls در این پنجره کلیک کنید.
- ۴ - گزینه Microsoft Common Dialog Control 6.0.0 را در لیستی که نمایش داده شده است انتخاب کنید (روی مربع ابتدای جمله کلیک کنید).
- ۵ - دکمه فرمان OK را بفشارید تا به محیط طراحی فرم بازگردید. آیکن کنترل مزبور در جعبه ابزار اضافه شده است (شکل‌های ۱۰-۱۸ و ۱۰-۱۹).



شکل ۱۰-۱۸ نحوه اضافه کردن کنترل CommonDialog به جعبه ابزار



شکل ۱۹-۱۰ پنجره Components برای اضافه کردن کنترل CommonDialog به جعبه ابزار

پس از اضافه شدن آیکن کنترل می‌توانید این کنترل را مانند سایر کنترل‌ها به فرم‌های خود اضافه کنید. این کنترل هنگام اجرای برنامه در روی فرم مشاهده نمی‌شود و فقط زمانی کادرهای محاوره نمایش داده می‌شوند که متدهای مربوط به آن‌ها فراخوانی شوند. این کنترل به وسیله متدهای زیر می‌تواند کادرهای محاوره مربوطه را نمایش دهد این متدها را در جدول ۷-۱۰ مشاهده می‌کنید.

جدول ۷-۱۰ انواع متدهای کنترل CommonDialog

| نام متد | نام کادر محاوره |
|-------------|--------------------------------|
| ShowOpen | کادر محاوره باز کردن فایل‌ها |
| ShowSave | کادر محاوره ذخیره‌سازی فایل‌ها |
| ShowColor | کادر محاوره رنگ‌ها |
| ShowFont | کادر محاوره فونت |
| ShowPrinter | کادر محاوره چاپگر |
| ShowHelp | کادر محاوره راهنما |

کنترل CommonDialog در این حالت می‌تواند خصوصیات زیر را در دسترس شما قرار دهد:
خصوصیت FileName: این خصوصیت نام و مسیر فایل را که توسط کاربر انتخاب شده است، نگهداری می‌کند. نوع این خصوصیت رشته‌ای است.
خصوصیت Filter: این خصوصیت می‌تواند نوع فایل‌هایی را که باید در کادر محاوره

نمایش داده شوند تعیین کند، مثلاً اگر بخواهید فایل‌هایی با پسوند txt و doc را در کادرمحاوره ببینید، خصوصیت Filter را به صورت زیر تنظیم کنید.

```
dlg.Filter="Text Files(*.txt)|*.txt|Documents(*.doc)|*.doc"
```

هر بخش در خصوصیت Filter شامل دو قسمت می‌شود؛ قسمت اول یک توضیح است که در لیست Files of types دیده می‌شود و قسمت دوم عبارتی است که می‌تواند نحوه نمایش فایل‌ها را تعیین کند.

خصوصیت FilterIndex: در صورتی که بیش از یک فایل را به وسیله خصوصیت Filter برای نمایش در کادر محاوره تعیین کرده باشید، به طور پیش فرض اولین نوع فایل در خصوصیت Filter در کادرمحاوره در نظر گرفته می‌شود. به وسیله خصوصیت FilterIndex می‌توانید گزینه پیش فرض را انتخاب کنید. خصوصیت FilterIndex می‌تواند مقادیر عددی صحیح از ۱ به بالا را کسب کند و بر اساس قرار گرفتن گزینه‌های مورد نظر در خصوصیت Filter محاسبه شود. مقدار پیش فرض این خصوصیت اولین گزینه یا مقدار ۱ است. شکل کلی نحوه تنظیم این خصوصیت به این صورت است:



مقدار عددی = FilterIndex . نام کنترل CommonDialog

خصوصیت FileName: به وسیله این خصوصیت می‌توان به نام فایل انتخاب شده دسترسی پیدا کرد. این خصوصیت فقط نام فایل را بدون مسیر آن نگهداری می‌کند.

خصوصیت InitDir: زمانی که کادر محاوره Open یا Save As نمایش داده می‌شود اسامی فایل‌ها و پوشه‌های مسیر جاری در دیسک نمایش داده می‌شود. در صورتی که بخواهید مسیر ویژه‌ای را برای کادرمحاوره در نظر بگیرید، می‌توانید مسیر مورد نظر را در این خصوصیت ذخیره کنید. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

مسیر = InitDir . نام کنترل CommonDialog

مسیر یک عبارت رشته‌ای است که موقعیت فایل مورد نظر را تعیین می‌کند.

نکته  خصوصیات فوق را می‌توانید از طریق کادر محاوره Property Pages نیز تنظیم کنید. برای دسترسی به پنجره مزبور، در پنجره خصوصیات، کنترل CommonDialog را انتخاب کرده و روی خصوصیت Custom آن کلیک کنید، در ادامه در روبه روی این خصوصیت روی دکمه  کلیک کنید، کادر محاوره Property Pages نمایش داده خواهد شد. در ادامه روی زبانه Open / Save As کلیک کنید، می‌توانید مقادیر مورد نظر را برای هر خصوصیت در این پنجره و در بخش مربوط به هر خصوصیت بنویسید.



شکل ۲۰-۱۰



مثال ۵: پروژه Picture Viewer را به گونه‌ای تنظیم کنید تا بتوان با استفاده از کادر محاوره باز کردن فایل‌ها، هر تصویر دلخواهی را انتخاب و مشاهده کرد. به این منظور عملیات زیر را به ترتیب انجام دهید:

- ۱ - پروژه Picture Viewer را باز کنید.
- ۲ - یک کنترل کادر محاوره روی فرم قرار دهید و خصوصیت نام آن را روی dlgopen تنظیم کنید.

۳ - به ماژول فرم و رویداد Click مnuopen_ بروید و رویدادهای زیر را تنظیم کنید:

```
Private Sub Form_Load()
```

```
    dlgopen.Filter = "BitMapFile (*.bmp)|*.bmp|AllFiles (*.*)|*.*"
```

```
    dlgopen.FilterIndex = 1
```

```
    dlgopen.InitDir = "c:\"
```

```
End Sub
```

```
Private Sub mnuopen_Click()
```


```
    dlgopen.DialogTitle = "Open"
```

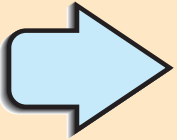
```
    dlgopen.ShowOpen
```

```
    imgshow.Stretch = False
```

```
    imgshow.Picture = LoadPicture(.dlgopen.FileName)
```

```
End Sub
```

- ۴ - پس از انجام، تغییرات فوق را ذخیره نمایید برنامه را اجرا کرده و روی گزینه Open از منوی File کلیک کنید، کادر محاوره Open باز می‌شود. در کادر محاوره Open روی دکمه  در لیست Files of type کلیک کنید تا لیست مربوطه باز شود. همان‌طور که ملاحظه می‌کنید توضیحات دو نوع فایل قابل مشاهده است؛ در صورتی که (BitMapFile (*.bmp)) انتخاب شود فقط فایل‌های با پسوند bmp و در صورتی که گزینه (AllFiles (*.*)) را انتخاب کنید، تمام فایل‌ها را خواهید دید. یک فایل گرافیکی به دلخواه انتخاب کرده و آن را نمایش دهید.



Learn in English

Assigning Access Keys to Menu Item

You can improve keyboard access to menu commands by defining access keys and shortcut keys.

Access Keys

Access keys allow the user to open a menu by pressing the ALT key and typing a designated letter. Once a menu is open, the user can choose a control by pressing the letter (the access key) assigned to it. For example, ALT+E might open the Edit menu, and P might select the Paste menu item. An access-key assignment appears as an underlined letter in the menu control's caption.

To assign an access key to a menu control in the Menu Editor

- 1 – Select the menu item to which you want to assign an access key.
- 2 – In the **Caption** box, type an ampersand (&) immediately in front of the letter you want to be the access key.

واژه نامه

| | |
|------------------------|-------------------------|
| Access Key | کلید دسترسی |
| Assignment | مأموریت |
| Compiler | مترجم |
| Immediately | بی‌درنگ، بلافاصله |
| Improve | اصلاح کردن، بهبود دادن |
| Interpreter | مفسر |
| Object | شیء |
| Source | برنامه مبدأ |
| Structural Programming | برنامه‌نویسی ساخت‌یافته |

خلاصه مطالب

- در ویژوال بیسیک از کادر محاوره Menu Editor برای ایجاد انواع منو استفاده می‌شود.
- از کنترل‌های نوار پیمایش افقی و عمودی برای مرور تصویر در کنترل‌های تصویر یا محتویات پنجره‌ها استفاده می‌شود.
- با استفاده از کنترل CommonDialog می‌توان انواع کادرهای محاوره باز و ذخیره کردن فایل، فونت، رنگ و غیره را ایجاد نمود.

آزمون نظری

۱ - کدام دستور برای ایجاد منوهای موضوعی به وسیله کلیک راست مناسب هستند؟

الف - MouseUp ب - PopupMenu ج - MouseMove د - Menu Editor

۲ - کدام رویداد در کنترل‌های نوار پیمایش در زمان کلیک روی مکان خالی از نوار

پیمایش اجرا می‌شود؟

الف - Change ب - Scroll ج - Click د - Move

۳ - در کنترل‌های نوار پیمایش کدام خصوصیت میزان تغییرات خصوصیت Value را

در زمان کلیک روی دکمه‌های مثلثی کنترل تعیین می‌کند؟

الف - LargeChange ب - SmallChange ج - Scroll د - Change

۴ - کدام خصوصیت در کنترل کادر محاوره می‌تواند نوع فایل‌هایی را که باید نمایش

داده شوند تعیین کند؟

الف - File Name ب - Filter ج - Filter Index د - InitDir

۵ - برای نمایش یک منوی موضوعی از کدام دستور استفاده می‌شود؟

الف - Menu ب - MenuPopup

ج - Popup د - PopupMenu

۶ - کدام رویداد در کنترل‌های نوار پیمایش در هنگام درگ کردن دکمه مستطیل شکل

اجرا می‌شود؟

الف - LargeChange ب - SmallChange

ج - Scroll د - MinScroll

۷ - کدام رویداد در کنترل‌های نوار پیمایش در زمان کلیک روی دکمه‌های مثلثی

شکل اجرا می‌شود؟

الف - Scroll ب - LargeChange ج - SmallChange د - Change

۸ - کدام متد نمی‌تواند یک کادر محاوره را نمایش دهد؟

الف - ShowOpen ب - ShowFile ج - ShowSave د - ShowColor

9 - Which of the following characters is required to create an access key for Menu item?

a- * b- ? c- - d- &

- ۱۰- انواع منو در رابط‌های گرافیکی را همراه با کاربرد هر یک توضیح دهید.
- ۱۱- انواع کادرهای محاوره و کاربرد هر یک از آن‌ها را توضیح دهید.
- ۱۲- انواع کنترل‌های نوار پیمایش و نحوه استفاده از آن‌ها را توضیح دهید.
- ۱۳- انواع منو و کاربرد هر یک از آن‌ها را توضیح دهید.
- ۱۴- خصوصیت Filter در کادر محاوره Open را توضیح دهید.

آزمون عملی

۱- یک برنامه نقاشی مطابق این شکل‌ها به گونه‌ای طراحی کنید تا کاربر توانایی رسم خطوط و مستطیل‌های دلخواه خود را با رنگ‌های مورد نظر داشته باشد.



۲- پروژه‌ای طراحی کنید که مطابق شکل‌های زیر توانایی محاسبه مساحت و محیط انواع اشکال هندسی را داشته باشد و به علاوه امکان تعیین واحد اندازه‌گیری نیز وجود داشته باشد.



توانایی استفاده از انواع آرایه‌ها در ویژوال بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کاراز فراگیر انتظار می‌رود که:

- ۱- توانایی ایجاد انواع آرایه‌های یک بعدی، با ابعاد ثابت و متغیر را داشته باشد.
- ۲- توانایی ذخیره‌سازی و بازیابی داده‌ها به وسیله آرایه‌های یک بعدی را داشته باشد.
- ۳- توانایی ارسال آرایه‌ها به رویه‌ها را داشته باشد.
- ۴- توانایی فراخوانی یک آرایه با تعداد آرگومان‌های نامعین را داشته باشد.
- ۵- توانایی ایجاد و استفاده از آرایه‌های دوبعدی را داشته باشد.
- ۶- نحوه استفاده از توابع Ubound و LBound را بداند.
- ۷- نحوه استفاده از توابع Split، Filter و Join را بداند.
- ۸- توانایی مرتب کردن اعضای یک آرایه با روش‌های زیر را داشته باشد.
 - الف - مرتب‌سازی با روش حبابی
 - ب - مرتب‌سازی با روش انتخابی
- ۹- توانایی جستجوی اطلاعات را در آرایه‌ها با روش خطی و دودویی داشته باشد.

کلیات

تاکنون با روش‌های مختلف ذخیره‌سازی اطلاعات در حافظه اصلی کامپیوتر آشنا شده‌اید که استفاده از انواع متغیرها و خصوصیات کنترل‌ها از نمونه‌های کاملاً مشخص آن است.

گاهی در برنامه‌نویسی‌های واقعی لازم است تعداد زیادی داده را مورد پردازش قرار دهیم که به ناچار باید به تعداد مورد نظر متغیر، تعریف کرد؛ اما این روش، همواره قابل اجرا نیست، مثلاً فرض کنید می‌خواهید برنامه‌ای را طراحی کنید که باید اسامی هزار نفر از دانشجویان یک دانشگاه را دریافت کند در چنین حالتی با توجه به دانسته‌های قبلی باید هزار متغیر با اسامی مختلف تعریف کنید. آیا این روش منطقی است؟ اگر تعداد داده‌ها باز افزایش پیدا کند، چطور؟ اگر بخواهید یک اسم را در میان مجموعه اسامی پیدا کنید، چه اتفاقی می‌افتد؟ این‌گونه عملیات با روش‌های معمول یا امکان‌پذیر نیست یا از نظر تکنیکی، منطقی نخواهد بود.

برای حل این مشکل و طراحی چنین برنامه‌هایی در تمام زبان‌های برنامه‌نویسی از مفهومی به نام آرایه (Array) استفاده می‌شود. یک آرایه در واقع یک سری از چندین متغیر با یک نام مشابه است که به وسیله یک اندیس (یک عدد صحیح مثبت) از یکدیگر متمایز می‌شوند.

استفاده از آرایه‌ها باعث می‌شود تا کدهای برنامه ساده‌تر و کوتاه‌تر شود زیرا شما می‌توانید با استفاده از انواع حلقه‌ها و شماره اندیس‌ها، به هر یک از اعضای آرایه دسترسی پیدا کنید. آرایه‌ها نیز مانند متغیرهای معمولی دارای نوع داده هستند و تمام اعضای یک آرایه از یک نوع داده هستند البته می‌توانید به وسیله استفاده از نوع داده Variant انواع مختلفی از داده‌ها را در اعضای یک آرایه ذخیره کنید.

استفاده از آرایه‌ها در برنامه‌های بزرگ اجتناب‌ناپذیر است و بدون استفاده از آن‌ها انجام عملیات مرتب‌سازی و جستجوی داده کار بسیار مشکلی خواهد بود. ابعاد یک آرایه به وسیله دامنه پایینی و بالایی آن معین می‌شود و اعضای آرایه به‌طور پیوسته و پشت سر هم داخل این محدوده قرار می‌گیرند. ویژوال بیسیک برای هر یک از اعضای یک آرایه فضای جداگانه‌ای را در حافظه اختصاص می‌دهد بنابراین استفاده از آرایه‌هایی بزرگ‌تر از اندازه مورد نیاز، باعث اشغال حافظه بدون استفاده خواهد شد.

۱-۱۱ تعریف انواع آرایه در ویژوال بیسیک

در ویژوال بیسیک دو نوع آرایه وجود دارد: آرایه با ابعاد ثابت آرایه ایستا یا (Static Array) و آرایه با ابعاد متغیر (آرایه پویا یا Dynamic Array). ابتدا به نحوه تعریف آرایه‌ها با ابعاد ثابت می‌پردازیم:


۱-۱-۱ آرایه ایستا (Static Array)

برای تعریف آرایه با ابعاد ثابت می‌توانید از تمام روش‌هایی که تاکنون برای تعریف متغیرها به کار گرفته‌اید، استفاده کنید تنها تفاوتی که بین تعریف متغیر و آرایه وجود دارد تعیین ابعاد یک آرایه است. آرایه‌ها را می‌توانید به وسیله کلمات کلیدی Public، Private، Static و Dim در یک رویه یا بخش تعاریف ماژول فرم یا ماژول کد تعریف کنید.

برای تعریف یک آرایه با ابعاد ثابت می‌توانید یکی از روش‌های زیر را استفاده کنید:

نوع داده As (دامنه بالایی) نام آرایه [Dim | Public | Private | Static]

نوع داده As (دامنه بالایی To دامنه پایینی) نام آرایه [Dim | Public | Private | Static]

 **نکته** با توجه به مکان تعریف آرایه و کاربرد آن می‌توانید یکی از کلمات کلیدی موجود در [] را انتخاب کنید.

مثلاً برای تعریف یک آرایه از نوع Integer و با تعداد ۱۵ عضو از فرمان زیر استفاده می‌شود:

```
Dim no (14) As Integer
```

در ویژوال بیسیک به طور پیش فرض اولین اندیس آرایه‌ها از شماره صفر آغاز می‌شود بنابراین در مثال قبل با توجه به مقدار دامنه بالایی، اندیس‌های آرایه از صفر تا ۱۴ در نظر گرفته می‌شوند و در نتیجه تعداد اعضا ۱۵ خواهد بود.
و در تعریف یک آرایه به صورت زیر:

```
Public counters(20) As Double
```

آرایه‌ای با تعداد ۲۱ عضو و از نوع Double در حافظه آدرس‌دهی خواهد شد. روش دوم در تعریف یک آرایه با ابعاد ثابت استفاده از دامنه بالایی و پایینی است مثلاً برای تعریف آرایه‌ای (با ۱۵ عضو) که اندیس اول آن از یک شروع شود و اندیس آخرین عضو در آن ۱۵ باشد از فرمان زیر استفاده می‌شود:

```
Dim counters (1 To 15) As Double
```

و در تعریف آرایه sums که به این صورت انجام شده است:

Private sums (100 To 120) As Variant

آرایه sums دارای اولین عضو با شماره اندیس ۱۰۰ و آخرین عضو با شماره اندیس ۲۰ خواهد بود به عبارت دیگر ۲۱ عضو خواهد داشت. برای آن که با نحوه کار آرایه‌ها بهتر آشنا شوید به ذکر مثالی در این رابطه می‌پردازیم:



مثال ۱: رویه‌ای بنویسید که ده عدد را به صورت تصادفی ایجاد کرده و در آرایه‌ای ذخیره کند، سپس آرایه را نمایش دهد.

به این منظور یک رویه با نام myrandom به صورت زیر بنویسید:

Sub myrandom()

Dim i As Integer, sngno(9) As Single

Randomize

For i = 0 To 9

sngno(i) = Rnd

Next i

For i = 0 To 9

Print , "number("; i + 1; ")="; sngno(i)

Next i

End Sub

در رویه myrandom ابتدا آرایه‌ای با دامنه بالایی ۹ (۱۰ عضو) تعریف شده است سپس به وسیله یک حلقه For که مقدار شمارنده آن (i) از صفر شروع می‌شود اولین عضو آرایه یعنی sngno(0) به وسیله تابع Rnd مقداردهی می‌شود برای نمایش اعدادی که در آرایه sngno ذخیره شده‌اند نیز از یک حلقه استفاده شده است. البته شما می‌توانید به جای For از حلقه‌های دیگر نیز استفاده کنید اما این کار با استفاده از حلقه For آسان‌تر خواهد بود. در رویه myrandom تابعی به نام Rnd وجود دارد، به وسیله این تابع می‌توان اعداد تصادفی بین صفر و یک را ایجاد کرد. این تابع یک عدد از نوع Single را برمی‌گرداند و یک آرگومان اختیاری دارد که می‌تواند یک عدد از نوع Single یا یک عبارت عددی باشد. توجه داشته باشید که هر بار تابع Rnd فراخوانی می‌شود یک عدد تصادفی تولید خواهد شد البته اگر برنامه مجدداً اجرا و تابع فراخوانی شود همان اعداد

به صورت تکراری به دست می‌آیند؛ برای جلوگیری از چنین حالتی و این‌که همواره اعداد تولید شده یکسانی به دست نیایند، می‌توانید قبل از تابع Rnd از تابع Randomize استفاده کنید. تابع Randomize دارای یک آرگومان اختیاری نیز بوده که می‌تواند یک عدد یا عبارت عددی باشد.

تمرین:



رویه مثال ۱ را به‌گونه‌ای تغییر دهید تا بزرگ‌ترین عضو آرایه را به‌دست آورده و همراه اعضای آرایه نمایش دهد. سپس در یک پروژه آن را فراخوانی کنید و نتیجه را مشاهده نمایید.

مثال ۲: رویه‌ای بنویسید که دو آرایه ۵ عضوی یک بعدی از اعداد تصادفی را ایجاد کرده و حاصل ضرب آن‌ها را محاسبه نماید و در آرایه دیگری ذخیره کند.



Sub mymatrix()

```
Dim i As Integer, no1(4) As Single
Dim no2(4) As Single, no3(4) As Single
Randomize
For i = 0 To 4
    no1(i) = Rnd
    no2(i) = Rnd
Next i
For i = 0 To 4
    no3(i) = no1(i) * no2(i)
Next i
```

End Sub

در این رویه از سه آرایه ۵ عضوی استفاده شده است و به‌وسیله اولین حلقه For اعضای دو آرایه no1 و no2 مقداردهی شده‌اند سپس با استفاده از حلقه For دوم حاصل ضرب اعضای متناظر دو آرایه در عضو متناظر آرایه حاصل ضرب یعنی no3 قرار می‌گیرد. علاوه بر آرایه‌های عددی می‌توانید داده‌های رشته‌ای را نیز به‌صورت آرایه ذخیره کنید. در واقع فرق زیادی بین آرایه‌های رشته‌ای و عددی وجود ندارد فقط در مورد آرایه‌های رشته‌ای طول هر عضو می‌تواند ثابت یا متغیر باشد.

تمرین:



پروژه‌ای طراحی کنید که با استفاده از یک آرایه، هر عددی از مبنای ۱۰ را به مبنای ۲ تبدیل کند.



مثال ۳: پروژه‌ای طراحی کنید که اسامی ۱۰ نفر را دریافت کرده و در یک آرایه رشته‌ای ذخیره کند به علاوه بتوان اسامی ذخیره شده را روی یک فرم جداگانه مشاهده نمود. به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ - برنامه ویژوال بیسیک را اجرا کرده و یک پروژه از نوع Standard EXE ایجاد کنید سپس یک فرم مطابق شکل ۱-۱۱ و جدول ۱-۱۱ ایجاد کنید و کنترل‌های آن را مطابق جدول ۲-۱۱ روی فرم قرار دهید. در این فرم از یک کنترل کادر متن برای دریافت داده‌ها و از دکمه Add برای ذخیره‌سازی اسامی در آرایه استفاده می‌شود به علاوه با دکمه Show می‌توان اسامی ذخیره شده در آرایه را در یک فرم دیگر مشاهده کرد.

جدول ۱-۱۱ خصوصیات فرم

| مقدار | خصوصیت |
|----------|---------|
| frmarray | Name |
| Array | Caption |



شکل ۱-۱۱

جدول ۲-۱۱ خصوصیات کنترل‌ها

| کنترل / خصوصیت | Command Button | Command Button | Command Button | Text Box | Label |
|----------------|----------------|----------------|----------------|----------|-------------|
| Name | cmdadd | cmdshow | cmdexit | txtname | lblen |
| Caption | Add& | Show& | Exit& | _____ | :Enter Name |

۲ - یک ماژول کد به پروژه اضافه کنید و در بخش تعاریف آن آرایه‌ای را با نام strname با ۱۰ عضو که هر عضو آن نیز توانایی دریافت ۲۰ کاراکتر را داشته باشد به صورت زیر تعریف کنید:

```
Public strname(9) As string* 20
```

رویداد Click کنترل دکمه Add را به این صورت تنظیم کنید:

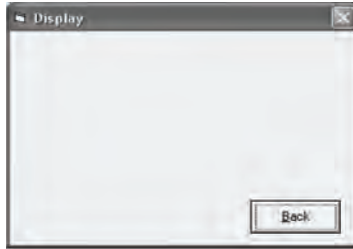
```
Private Sub cmdadd_Click()  
    Static i As Integer  
    If i < 10 Then  
        strname(i) = txtname.Text  
        i = i + 1  
    Else  
        MsgBox «Array is full»  
    End If  
End Sub
```

در این رویداد از یک متغیر i به‌عنوان شمارنده اندیس آرایه به‌صورت Static استفاده شده است تا به‌وسیله مقدار این متغیر بتوان در هر مرحله یک عضو در آرایه را با نامی که کاربر در کنترل کادر متن می‌نویسد پر کرد. اگر این متغیر به‌صورت محلی تعریف شود اسامی، همواره در اولین عضو آرایه ذخیره می‌شوند و در نتیجه، نامی که قبلاً در عضو اول آرایه ذخیره شده است از بین می‌رود. در ادامه اجرای رویداد، یک فرمان If مقدار i را کنترل می‌کند تا مقدار i از دامنه بالایی آرایه (یعنی ۹) تجاوز نکند در صورتی که مقدار i کنترل نشود در زمان رسیدن به مقدار 10 ، چون بالاترین مقدار اندیس آرایه ۹ است پیام خطای Subscript out of range نمایش داده می‌شود. بنابراین اگر $(i < 10)$ باشد آنگاه محتویات خصوصیت Text کنترل کادر متن در یکی از اعضای آرایه ذخیره می‌شود و سپس مقدار i یک واحد افزایش می‌یابد تا مرحله بعد، اندیس عضو بعدی آرایه آماده باشد، اما اگر کاربر 10 نام را وارد کند در هنگام ورود نام یازدهم با پیامی که به‌وسیله یک تابع MsgBox نمایش داده می‌شود از کامل شدن روند عملیات ورود داده مطلع می‌شود زیرا در رویداد Click دکمه فرمان Add نتیجه بررسی شرط $(i < 10)$ نادرست بوده و در نتیجه تابع MsgBox فراخوانی می‌شود.

۴ - یک فرم جدید با نام frmdisplay و عنوان Display مطابق شکل ۲-۱۱ به پروژه اضافه کرده سپس یک دکمه با نام cmdback و عنوان Back& روی آن قرار دهید. از این فرم برای نمایش اسامی ذخیره شده در آرایه استفاده می‌شود.

۵ - رویداد Click دکمه Show را در فرم frmarray به صورت زیر تنظیم کنید:

```
Private Sub cmdshow_Click()  
    frmarray.Hide  
    frmdisplay.Show  
End Sub
```



شکل ۱۱-۲

در این رویداد با استفاده از متد Hide فرم frmarray مخفی شده و با استفاده از متد Show فرم frmdisplay نمایش داده می شود.

۶ - رویداد دکمه cmdback در فرم frmdisplay به صورت زیر تنظیم کنید:

```
Private Sub cmdback_Click()  
    unload frmdisplay  
    frmarray.Show  
End Sub
```

در این رویداد با استفاده از دستور unload فرم frmdisplay بسته شده و با متد Show امکان بازگشت به فرم اول فراهم می شود.

۷ - پس از تنظیم رویداد دکمه فرمان cmdback در فرم frmdisplay دستورات زیر را در رویداد Activate همین فرم به صورت زیر تنظیم کنید:

```
Private Sub Form_Activate( )  
    Dim j  
    For j = 0 To 9  
        Print "NAME ( "; j ; " ) = "; strname(j)  
    Next j  
End Sub
```


دستورات در این رویداد باعث خواهند شد تا پس از فعال شدن فرم محتویات آرایه به وسیله یک حلقه نمایش داده شود.

- ۸ - پروژه و فرم frmarray را با نام Array و فرم frmdisplay را با نام Display ذخیره کنید سپس برنامه را اجرا نمایید و تعداد ۰ نام را وارد کرده و بعد از ورود هر نام، دکمه Add را کلیک کنید و بعد از پایان ورود داده‌ها به وسیله دکمه Show محتویات آرایه را مشاهده نمایید در پایان با کلیک روی دکمه Back به فرم اول بازگردید.
- ۹ - روی دکمه Exit کلیک کنید و به پنجره ویژوال بیسیک بازگردید.


تمرین:

برنامه قبل را به گونه‌ای تغییر دهید که در صورت عدم استفاده از تمام اعضای آرایه (خالی ماندن بعضی از اعضا) در هنگام نمایش اطلاعات در فرم cmddisplay، فقط تا آخرین اندیسی از آرایه که حاوی نام است، نمایش داده شود مثلاً اگر کاربر ۴ نام را وارد کرده است فقط همان ۴ نام نمایش داده شوند و از نمایش عناصر بعدی آرایه یعنی اندیس‌های بزرگ‌تر از ۳ خودداری شود.

در این جا لازم است که به ذکر نکته مهمی در رابطه با شماره اندیس اولین عضو در آرایه پردازیم. تاکنون وقتی یک آرایه را با ذکر مقدار اندیس بالایی آن تعریف می‌کنید شماره اندیس اولین عضو در آرایه از صفر شروع می‌شود، اما گاهی لازم است که این مقدار را با توجه به نیاز تغییر دهید، با استفاده از فرمان Option Base در بخش تعاریف می‌توانید مقدار اندیس اولین عضو را در آرایه‌ها تعیین کنید. شکل کلی فرمان به این صورت است:

Option Base n

n می‌تواند صفر یا یک باشد در صورت استفاده از مقدار صفر یا عدم استفاده از فرمان فوق اندیس آغازین در آرایه‌های برنامه صفر خواهد بود و اگر بخواهید اندیس آغازین در آرایه‌های برنامه از یک شروع شود مقدار n را در فرمان مزبور ۱ انتخاب کنید.

 **نکته** • از این فرمان فقط یک بار و در بخش تعاریف یکی از ماژول‌ها استفاده کنید.
• برای تعیین دامنه پایینی آرایه‌ها به جای استفاده از فرمان Option Base بهتر است در تعریف آرایه‌ها دامنه پایینی و بالایی آرایه را تعیین کنید.

به عنوان مثال فرض کنید می خواهیم یک آرایه عددی با 10° عضو را به وسیله اعداد تصادفی مقداردهی کنیم (این مسأله قبلاً حل شده است اما از آن، برای بررسی عملکرد دستور Option Base استفاده می شود).

Sub myrandom()

Dim i As Integer, sngno(10) As Single

Randomize

For i = 1 To 10

sngno(i) = Rnd

Next i

End Sub

اگر دستور Option Base 1 استفاده شود در رویه myrandom جدید و در تعریف آرایه sngno به جای عدد ۹ عدد 10° استفاده می شود و مقدار پایانی در حلقه For نیز از ۹ به 10° تغییر می یابد و مقدار شروع شمارنده نیز از صفر به ۱ تبدیل می شود. توجه داشته باشید که در ذخیره سازی مقادیر هر دو حالت، مشابه هم عمل می شود؛ فقط در نحوه تعریف آرایه ها و استفاده از حلقه باید دقت کافی داشته باشید.

تمرین:



پروژه ای طراحی کنید که ده نمره درس های یک دانش آموز را در یک آرایه ذخیره کند سپس مجموع و معدل نمرات وی را محاسبه کرده و نمایش دهد (اندیس آرایه از عدد ۱ شروع شود).

۲-۱-۱۱ آرایه پویا (Dynamic Array)

گاهی اوقات ممکن است تعداد داده ها نامشخص باشد در نتیجه نمی توان تعداد اعضا آرایه را در زمان طراحی برنامه تعیین کرد بنابراین لازم است ابعاد آرایه در زمان اجرا تنظیم شود.

به این منظور ویژوال بیسیک نوع دیگری از آرایه ها را با نام آرایه پویا در اختیار شما قرار می دهد. ابعاد آرایه ای از این نوع را می توانید در زمان اجرای برنامه با توجه به نیازتان مکرراً تغییر دهید.

برای تعریف یک آرایه دینامیک، بهتر است مانند آرایه‌های ثابت عمل کرده اما از ذکر ابعاد آرایه خودداری کنید.
به‌عنوان مثال به دستور زیر توجه کنید:

```
Dim dynnumber ( ) AS Integer
```

اما در صورت استفاده از آرایه در این مرحله پیام خطا نمایش داده خواهد شد. برای قابل استفاده شدن آرایه‌های پویا، پس از تعریف آن باید ابعاد آن را با استفاده از دستور ReDim تعیین کنید. به‌عنوان مثال پس از تعریف آرایه dynnumber از نوع Integer ابعاد آن به صورت زیر تعیین می‌شود:

```
ReDim dynnumber (10)
```

اگر Option Base ۰ باشد آرایه دارای یازده عضو و اگر Option Base ۱ باشد آرایه دارای ۱۰ عضو (از ۱ تا ۱۰) خواهد بود.



- **نکته** در صورت استفاده از دستور ReDim مقادیر موجود در تمام اعضای آرایه از بین خواهد رفت بنابراین در استفاده مجدد از دستور ReDim با دقت کافی اقدام کنید.
- در صورتی که بخواهید مقادیر موجود در آرایه در زمان تغییر ابعاد آن حفظ شوند از کلمه کلیدی Preserve همراه با دستور ReDim استفاده کنید.
- آرایه‌های پویا را نیز می‌توانید به‌صورت Public، Private یا Static تعریف کنید.

به‌عنوان مثال به دستورات زیر توجه کنید: (با فرض این که Option Base 1 است).

```
Dim myarray( ) As Integer, i As Integer
```

```
ReDim myarray(5)
```

```
Next i
```

```
ReDim myarray(10)
```

```
For i = 1 To 10
```

```
Print myarray(i)
```

```
Next i
```

با استفاده از تعریف آرایه پویا آرایه myarray با ۵ عضو تعریف شده است سپس به‌وسیله یک حلقه For مقادیر ۱ تا ۵ در آرایه قرار گرفته‌اند. پس از حلقه For اول مجدداً دستور ReDim به‌کار گرفته شده است تا تعداد اعضای آرایه دو برابر شود پس از تغییر

ابعاد آرایه، حلقه For دوم مقادیر موجود در آرایه را نمایش می‌دهد اما استفاده دوباره از دستور ReDim، تمام مقادیر قبلی در آرایه را از بین می‌برد، در نتیجه فقط مقادیر صفر توسط حلقه نمایش داده می‌شود.

حال اگر به جای دستور ReDim myarray (10) از دستور ReDim Preserve myarray (10) استفاده کنید، حلقه For دوم پس از نمایش مقادیر ۱ تا ۵ برای اعضای قبلی، مقدار صفر را هم برای ۵ عضو جدید که اضافه شده‌اند، نمایش می‌دهد.



تمرین:

یک پروژه جدید طراحی کنید و دستورات فوق را در رویداد Click یک دکمه فرمان قرار داده و نتیجه را در دو حالت بحث شده بررسی کنید.



نکته • در صورت کاهش ابعاد یک آرایه به وسیله دستور ReDim مقادیر مربوط به اعضای حذف شده از بین می‌روند.

• به وسیله دستور ReDim نیز می‌توانید یک آرایه پویا Dynamic تعریف کنید شکل کلی این دستور برای تعریف یک آرایه پویا به صورت زیر است:

نوع داده As (دامنه بالایی) نام آرایه ReDim
مثلاً دستور زیر یک آرایه با ۲۰ عضو و از نوع رشته‌ای تعریف می‌کند.

ReDim fam (19) As String

• در صورت استفاده از دستور ReDim برای تغییر ابعاد یک آرایه ثابت در هنگام اجرای برنامه، پیام خطای Array already dimensioned نمایش داده می‌شود.



تمرین:

می‌خواهیم پروژه‌ای که در مثال ۳ طراحی شده است به گونه‌ای تغییر کند تا تعداد اسامی با توجه به نیاز و درخواست کاربر دریافت و در آرایه ذخیره شود. به این منظور پروژه Array را باز کنید و عملیات زیر را به ترتیب انجام دهید:

۳-۱-۱۱ دستور Erase

از این دستور برای حذف اعضا در آرایه‌های پویا و از بین بردن محتویات اعضا در آرایه‌های ثابت استفاده می‌شود. شکل کلی دستور Erase به صورت زیر است:

نام آرایه Erase

اگر این دستور روی یک آرایه پویا اجرا شود تمام مقادیر آن از بین خواهد رفت و شما می‌توانید مجدداً با استفاده از دستور ReDim از آرایه استفاده کنید اگر پس از حذف اعضای یک آرایه پویا، بدون استفاده از دستور ReDim سعی در دستیابی به اعضای آن داشته باشید پیام خطای Subscript out of range نمایش داده می‌شود.

اگر این دستور روی یک آرایه ثابت اجرا شود مقادیر موجود در آرایه حذف خواهند شد، اما فضاهای مربوط به اعضای آرایه در حافظه از بین نخواهند رفت.

به عنوان مثال به دستورات زیر توجه کنید (با فرض این که ۱ Option Base است):

```
Dim myarray1() As Integer, myarray2(5) As Integer
```

```
Dim i As Integer
```

```
ReDim myarray1(5)
```

```
For i = 1 To 5
```

```
    myarray1(i) = i
```

```
    myarray2(i) = i
```

```
Next i
```

```
Erase myarray1, myarray2
```

```
ReDim myarray1(5)
```

```
For i = 1 To 10
```

```
    Print myarray1(i)
```

```
    Print myarray2(i)
```

```
Next i
```

در این مجموعه دستورات از یک آرایه پویا (myarray1) و یک آرایه‌ای با ابعاد ثابت (myarray2) استفاده شده است. در اولین حلقه For، آرایه‌های myarray1 و myarray2 به ترتیب مقداردهی شده‌اند سپس به وسیله دستور Erase، آرایه myarray1 از حافظه کاملاً پاک شده اما آرایه myarray2 فقط مقادیر خود را از دست می‌دهد و در نتیجه حلقه For دوم فقط مقادیر صفر را برای هر دو آرایه نمایش می‌دهد. البته اگر دستور ReDim myarray1(5) پس از حذف آرایه مزبور استفاده نشود آرایه در حلقه For قابل شناسایی نبوده و پیام خطای Subscript out of range نمایش داده می‌شود.



تمرین:

پروژه Array را به گونه‌ای تنظیم کنید که شرایط زیر را داشته باشد:
الف- امکان افزایش اسامی پس از پرشدن آرایه وجود داشته باشد.
ب- در صورت لزوم بتوان اعضای آرایه را حذف کرده و اسامی جدید در آن ذخیره نمود.

۲-۱۱ آرایه‌های چند بعدی

تاکنون با نحوه تعریف و چگونگی استفاده از آرایه‌های یک بعدی آشنا شدید اما در برنامه‌نویسی پروژه‌های واقعی، گاهی اوقات آرایه‌های یک بعدی نیز گره‌گشا نیستند و برنامه‌نویسی را با مشکلات متعددی روبه‌رو می‌کنند.

آرایه‌های دو بعدی از دو اندیس برای شناسایی اعضای خود استفاده می‌کنند. اندیس اول شماره سطر و اندیس دوم شماره ستون متناظر با آرایه را مشخص می‌کنند. برای روشن شدن بهتر موضوع به آرایه زیر توجه کنید، این آرایه دارای ۴ سطر و ۳ ستون است

جدول ۱۱-۳


| شماره ستون \ شماره سطر | 0 | 1 | 2 |
|------------------------|-----------|-----------|-----------|
| 0 | $a_{0,0}$ | $a_{0,1}$ | $a_{0,2}$ |
| 1 | $a_{1,0}$ | $a_{1,1}$ | $a_{1,2}$ |
| 2 | $a_{2,0}$ | $a_{2,1}$ | $a_{2,2}$ |
| 3 | $a_{3,0}$ | $a_{3,1}$ | $a_{3,2}$ |

در نوع از آرایه‌ها برای دسترسی به هر یک از عناصر آرایه از دو اندیس استفاده می‌شود مثلاً عنصر $a_{2,1}$ ، عضوی است که در سطر شماره ۲ و ستون شماره ۱ قرار دارد. برای تعریف آرایه دو بعدی می‌توانید یکی از این روش‌ها را به کار ببرید:

(دامنه بالایی ستون‌ها، دامنه بالایی سطرها) نام آرایه [Dim| Public| Private| Static]

نوع داده As (دامنه بالایی ستون‌ها، دامنه بالایی سطرها) نام آرایه [Dim| Public| Private| Static]

نوع داده As (دامنه بالایی ستون‌ها To دامنه پایینی ستون‌ها، دامنه بالایی سطرها To دامنه پایینی سطرها) نام آرایه [Dim| Public| Private| Static]

نکته  با توجه به مکان تعریف و کاربرد آرایه می‌توانید یکی از کلمات کلیدی موجود در [] را انتخاب کنید.


به عنوان مثال فرض کنید می‌خواهید نمرات سه درس دانش‌آموزان یک کلاس ۵ نفره را به صورت یک آرایه دو بعدی بنویسید.


| شماره ستون \ شماره سطر | ریاضی | فیزیک | شیمی | |
|------------------------|-------|-------|------|-----------------|
| | ۰ | ۱ | ۲ | |
| ۰ | ۱۷ | ۱۴ | ۱۶ | دانش‌آموز اول |
| ۱ | ۱۲ | ۱۰ | ۸ | دانش‌آموز دوم |
| ۲ | ۲۰ | ۱۸ | ۱۵ | دانش‌آموز سوم |
| ۳ | ۱۴ | ۱۳ | ۱۹ | دانش‌آموز چهارم |
| ۴ | ۱۷ | ۲۰ | ۱۱ | دانش‌آموز پنجم |

اکنون می‌خواهیم این آرایه دو بعدی را تعریف کنیم بنابراین می‌توان یکی از موارد زیر را برای انجام این کار استفاده کرد:

Dim sng student (4,2) As Single

Dim sng student (0 To 4, 0 To 2) As Single

نکته  توجه داشته باشید که قوانین دستور Option Base برای آرایه‌های دو بعدی نیز صادق است.

مثال ۴:  رویه‌ای بنویسید که آرایه دو بعدی ۴×۴ را با اعداد تصادفی مقداردهی کرده سپس اعضای آن را نمایش دهد (با فرض Option Base 0).

Sub mymatrix()

Dim A(4, 4) As Integer, i As Integer, j As Integer

Randomize

For i = 0 To 3

For j = 0 To 3

A(i, j) = Int(100 * Rnd + 1)

Next j

Next i

For i = 0 To 3

For j = 0 To 3

Print " "; A("; i"; "; j; ")="; A(i, j)

Next j

Print

Next i

End Sub

رویه فرعی فوق در صورت اجرا، خروجی مشابه شکل زیر را خواهد داشت:

A(0,0)=0 A(0,1)=20 A(0,2)=17 A(0,3)=9

A(1,0)=2 A(1,1)= 0 A(1,2)=52 A(1,3)=53

A(2,0)=69 A(2,1)=44 A(2,2)=0 A(2,3)=31

A(3,0)=28 A(3,1)=87 A(3,2)=3 A(3,3)=0

در این رویه ابتدا آرایه دو بعدی A را تعریف کرده‌ایم سپس با استفاده از دو حلقه تو در تو و تابع Rnd عناصر آرایه مقداردهی شده‌اند. حلقه For اول به وسیله شمارنده i شماره سطر و حلقه دوم به وسیله شمارنده j شماره ستون را در آرایه کنترل می‌کنند بنابراین با هر بار اجرای حلقه اول، حلقه دوم سه بار تکرار می‌شود تا اعضای هر سطر را جهت مقداردهی به ترتیب آماده کند. به عنوان مثال زمانی که مقدار i برابر با صفر است حلقه دوم مقادیر z را به ترتیب روی مقادیر ۰، ۱، ۲ و ۳ تنظیم می‌کند. با خاتمه اجرای حلقه دوم، حلقه اول مقدار i را یک واحد افزایش می‌دهد و حلقه دوم مجدداً از z=۰ شروع شده و اعضای سطر دوم نیز به ترتیب مورد دسترسی قرار می‌گیرند. در پایان پس از مقداردهی آرایه، با استفاده از دو حلقه For تودرتو اعضای آرایه نمایش داده می‌شوند

تمرین:



پروژه‌ای طراحی کنید که بتواند نام، نام خانوادگی و نمره سه درس، پنج دانش‌آموز را در آرایه ذخیره کند و در ضمن بتوان هر لحظه اطلاعات دانش‌آموزان را مشاهده کرد.

۱۱-۳ توابع LBound و UBound

از این توابع برای دستیابی به دامنه پایینی و بالایی یک آرایه استفاده می‌شود. تابع LBound می‌تواند با دریافت نام یک آرایه، دامنه پایینی آن را به صورت یک عدد از نوع Long بازگرداند. به عبارت دیگر به وسیله این تابع می‌توانید شماره اولین اندیس را در آرایه به دست آورید. شکل کلی این تابع به این صورت است:

LBound (arrayname, dimension)

تابع LBound یک آرگومان اجباری و یک آرگومان اختیاری دارد، آرگومان arrayname در واقع نام آرایه‌ای است که می‌خواهید اندیس آغازین آن را به دست آورید. آرگومان دوم (dimension) در آرایه‌های دو و چند بعدی استفاده می‌شود و بعدی از آرایه که می‌خواهید اندیس آغازین آن را پیدا کنید به صورت یک عدد صحیح تعیین می‌شود. مقدار ۱ برای بعد اول (سطر)، ۲ برای بعد دوم (ستون) و الی آخر. در صورت عدم استفاده از این آرگومان، مقدار پیش فرض ۱ خواهد بود. به عنوان مثال به این دستورات توجه کنید:

```
Dim A(1 To 100, 10 To 13) As Integer
```

```
Dim B(10) As Integer
```

```
Print LBound(A, 1)
```

```
Print LBound(A, 2)
```

```
Print LBound(B)
```

در دستورات قبل یک آرایه دو بعدی (A) و یک آرایه یک بعدی (B) تعریف شده‌اند. نتیجه اجرای اولین و دومین دستور Print به ترتیب ۱ و ۱۰ خواهد بود و در صورت اجرای دستور آخر با توجه به عدم استفاده از آرگومان اختیاری و یک بعدی بودن آرایه مقدار صفر نمایش داده می‌شود. توجه داشته باشید که فرض بر این باشد که : Option Base ۰ است.

تابع UBound می‌تواند با دریافت نام یک آرایه دامنه بالایی آن را به صورت یک عدد از نوع Long بازگرداند به عبارت بهتر عددی را که در تعریف آرایه به عنوان دامنه بالایی آرایه تعریف کرده‌ایم در اختیار شما قرار می‌دهد. شکل کلی این تابع به صورت زیر است:

UBound (arrayname ,dimension)

این تابع نیز یک آرگومان اجباری و یک آرگومان اختیاری دارد و عملکردی را مشابه آنچه در تابع LBound دارند به عهده می‌گیرند. به عنوان مثال به دستورات زیر توجه کنید:

```
Dim A ( 1 To 100, 10 To 13 ) As Integer
```

```
Dim B (10) As Integer
```

```
Print UBound(A, 1)
```

```
Print UBound(A, 2)
```

```
Print UBound(B)
```

اولین و دومین دستور Print به ترتیب مقادیر 10^0 و 13 را نمایش می‌دهند و دستور آخر نیز مقدار 10^0 را نمایش خواهد داد.

البته باید توجه داشت که دستور Option Base تأثیری روی تابع UBound نمی‌گذارد. به عنوان مثال به این دستورات توجه کنید:

```
Dim A(5) As Integer
```

```
For i = LBound(A) To UBound(A)
```

```
Print i
```

```
Next i
```

در دستورات قبل اگر Option Base 0 باشد حلقه For از مقدار صفر تا ۵ را نمایش می‌دهد و در واقع آرایه ۶ عضو خواهد داشت ولی اگر Option Base 1 باشد حلقه For از مقدار ۱ تا ۵ را نمایش می‌دهد و در نتیجه آرایه ۵ عضو خواهد داشت پس همان‌طور که دیدید دستور Option Base روی تابع UBound تأثیر نمی‌گذارد.

۴-۱۱ تابع Split

به وسیله این تابع می‌توان محتویات یک متغیر رشته‌ای را به صورت زیر رشته‌های جداگانه‌ای در یک آرایه یک بعدی ذخیره کرد. مقدار بازگشتی این تابع یک آرایه یک بعدی است. شکل کلی این تابع به صورت زیر است

Split (expression, delimiter, limit, compare)


این تابع دارای یک آرگومان اجباری است. expression می‌تواند یک عبارت رشته‌ای باشد. آرگومان delimiter یک آرگومان اختیاری است و به‌طور پیش‌فرض یک کاراکتر فاصله " " است. در واقع این آرگومان کاراکتری را که برای جدا کردن زیر رشته‌ها در آرگومان expression مورد استفاده قرار می‌گیرد معین می‌کند.

آرگومان limit دومین آرگومان اختیاری است و تعداد زیر رشته‌های مورد نظر را که باید بازگشت داده شوند تعیین می‌کند در صورت عدم استفاده از این آرگومان مقدار آن ۱- خواهد بود که به معنی بازگشت دادن تمام زیررشته‌هاست.

آرگومان compare نیز اختیاری بوده و تعیین می‌کند که آیا تابع بین حروف بزرگ و کوچک آرگومان delimiter با کاراکترهای expression تفاوت قائل شود یا خیر. مقادیری که این آرگومان می‌تواند دریافت کند در جدول ۴-۱۱ آرایه شده‌اند.

جدول ۴-۱۱ مقادیر مربوط به آرگومان Compare

| ثابت رشته‌ای | ثابت عددی | توضیح |
|-----------------|-----------|---|
| vbBinaryCompare | ۰ | تابع بین حروف کوچک و بزرگ تفاوت قائل می‌شود. |
| vbTextCompare | ۱ | تابع بین حروف کوچک و بزرگ تفاوت قائل نمی‌شود. |

نکته مقدار پیش‌فرض برای آرگومان compare صفر است. 

برای روشن شدن بهتر مطلب به رویه رویداد زیر توجه کنید:

```
Private Sub cmdsplit_Click ()
```

```
    Dim strname As String, stresult() As String
```

```
    Dim i As Integer
```

```
    strname = "microsoft visual basic 6"
```

```
    stresult = Split(strname)
```

```
    (For i = 0 To UBound(stresult)
```

```
        Print stresult(i)
```

Next i

End Sub

در دستورات فوق دستور Split براساس فضاهاى خالى « موجود در رشته strname, کلمات موجود در متغیر رشته‌ای strname را به صورت جداگانه در آرایه strresult ذخیره می کند که به وسیله یک حلقه For مطابق شکل ۱۱-۳ نمایش داده می شود.

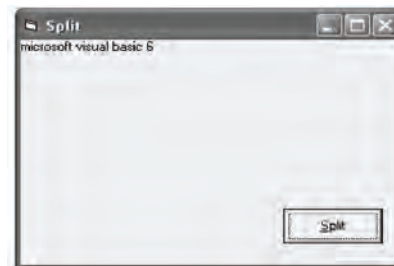


شکل ۱۱-۳

اکنون فرض کنید تابع Split را به صورت زیر تغییر دهید:

strresult = Split (strname , "A")

در این صورت کاراکتری که برای جدا کردن زیررشته‌ها استفاده می شود کاراکتر A خواهد بود. اما همان طور که در شکل ۱۱-۴ مشاهده می کنید رشته strname بدون توجه به کاراکتر A بدون تغییر باقی می ماند چون در این دستور از آرگومان compare استفاده نشده است مقدار پیش فرض صفر است در نتیجه تابع بین کاراکتر A در آرگومان دوم با کاراکتر a در رشته strname تفاوت قائل شده است.



شکل ۱۱-۴

حال اگر دستور قبل را به صورت (1) «A», Split (strname, strresult = تغییر دهید خروجی برنامه به صورت شکل ۱۱-۵ خواهد بود یعنی تابع بین کاراکتر A در آرگومان دوم با کاراکتر a در رشته strname تفاوتی قائل نمی شود و strname به سه زیر رشته تقسیم می شود.

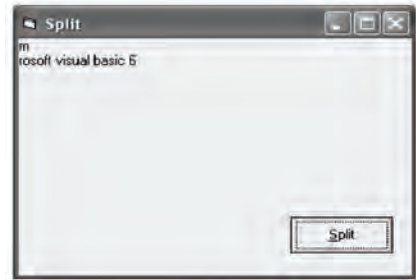
در صورتی که بخواهید تعداد معینی از زیررشته‌ها را جدا کنید می‌توانید از آرگومان سوم در تابع Split استفاده کنید. مثلاً اگر مثال قبل را به این صورت تغییر دهید:

```
strresult = Split(strname,"ic",2,1)
```

خروجی برنامه مطابق شکل ۱۱-۶ خواهد بود.



شکل ۱۱-۵



شکل ۱۱-۶

نکته در صورتی که آرگومان limit در تابع Split، ۱ باشد تمام رشته expression به صورت یک زیررشته بازگشت داده خواهد شد.

تمرین:



پروژه‌ای طراحی کنید که یک رشته و کاراکتر دلخواه را دریافت کند و رشته را براساس کاراکتر موردنظر به چند زیررشته تقسیم کرده و نمایش دهد.

۱۱-۵ تابع Join

عملکرد این تابع برعکس تابع Split است. به وسیله این تابع می‌توانید اعضای یک آرایه رشته‌ای را به صورت یک مقدار رشته‌ای واحد درآورید. شکل کلی این تابع به صورت زیر است:

Join (sourcearray, delimiter)

این تابع دارای یک آرگومان اجباری و یک آرگومان اختیاری است. آرگومان sourcearray، نام آرایه رشته‌ای مورد نظر است که می‌خواهید مقادیر موجود در آن را به صورت یکپارچه در یک متغیر رشته‌ای ذخیره کنید.

آرگومان delimiter یک آرگومان اختیاری است که کاراکتر جداکننده را بین کلمات در رشته‌ای که تابع بازگشت می‌دهد تعیین می‌کند. در صورت عدم استفاده از این آرگومان، کاراکتر فضای خالی " " مورد استفاده قرار می‌گیرد.
به‌عنوان مثال به دستورات زیر توجه کنید:

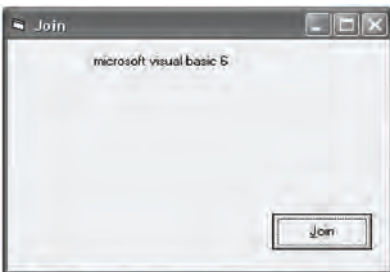
```
Private Sub cmdjoin_Click()  
    Dim strname(3) As String, stresult As String  
    strname(0) = "microsoft"  
    strname(1) = "visual"  
    strname(2) = "basic"  
    strname(3) = "6"  
    stresult = Join(strname)  
    Print  
    Print , stresult  
End Sub
```

در صورت اجرای دستورات فوق خروجی مشابه شکل ۷-۱۱ به دست می‌آید همان‌طور که می‌بینید مقادیر موجود در آرایه strname به‌وسیله تابع Join در متغیر stresult به‌صورت یک رشته واحد ذخیره شده‌اند و از کاراکتر فاصله خالی " " برای ایجاد فاصله بین مقادیر آرایه استفاده شده است.

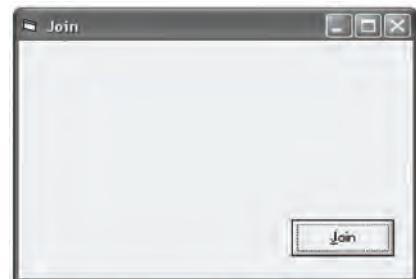
اکنون تابع Join را در مثال قبل به‌صورت زیر تغییر دهید:

```
stresult = Join ( strname , " # " )
```

در این صورت خروجی دستورات به‌صورت شکل ۸-۱۱ خواهد بود.



شکل ۷-۱۱



شکل ۸-۱۱



تمرین:

پروژه‌ای طراحی کنید که تعدادی کلمه را به صورت جداگانه دریافت کند سپس با استفاده از تابع Join آن‌ها را به یک رشته واحد تبدیل نموده و نمایش دهد.

۶-۱۱ نحوه ارسال آرایه‌ها به رویه‌ها

تاکنون با نحوه تعریف آرایه و چگونگی استفاده از آن‌ها آشنا شده‌اید اما گاهی اوقات لازم است تا آرایه‌ها را به یک رویه فرعی یا تابع ارسال کنید. در این جا با ذکر یک مثال با چگونگی انجام این کار آشنا می‌شوید.



مثال ۵: می‌خواهیم تابعی بنویسیم که با دریافت یک آرایه عددی از نوع Integer با ۱۰ عضو، کوچک‌ترین عضو را پیدا کرده و بازگرداند. ابتدا به تعریف تابع می‌پردازیم، در اینجا لازم است آرگومان ورودی به صورت آرایه استفاده شود بنابراین در هنگام تعریف آرگومان از دو پرانتز () استفاده می‌شود.

```
Public Function minelement(myarray() As Integer) As Integer
```

```
Dim i As Integer, min As Integer
```

```
min = myarray(0)
```

```
For i = 1 To 9
```

```
    If min > myarray(i) Then min = myarray(i)
```

```
Next i
```

```
minelement = min
```

```
End Function
```

همان‌طور که در تعریف تابع می‌بینید برای تعریف یک آرگومان از نوع آرایه فقط کافی است دو پرانتز در ابتدای نام آرایه ذکر کنید. برای فراخوانی این تابع نیز می‌توانید از دستور زیر استفاده کنید:

```
Print minelement (myarray ( ) )
```

نکته آرایه‌ها به‌طور پیش فرض به صورت ارسال با مرجع (Call By Reference) به رویه ارسال می‌شوند.



۷-۱۱ روش‌های مرتب‌سازی آرایه‌ها

در برنامه‌نویسی واقعی معمولاً لازم است تا اطلاعات دریافت شده یا نتیجه محاسبات را به صورت مرتب شده در اختیار کاربران قرار دهید. به این منظور می‌توان از آرایه‌ها استفاده کرد.

تاکنون روش‌های متعددی برای مرتب کردن داده‌های موجود در یک آرایه طراحی و معرفی شده‌اند که هر یک نقاط ضعف و قدرتی نیز دارند. از انواع روش‌های معروف در مرتب‌سازی اطلاعات می‌توان به روش مرتب‌سازی حبابی (Bubble Sort)، روش مرتب‌سازی Shell Sort، روش مرتب‌سازی سریع (Quick Sort)، روش مرتب‌سازی انتخابی (Selection Sort) و مرتب‌سازی به روش درج (Insertion Sort) اشاره کرد. بعضی از روش‌های نامبرده دارای الگوریتم پیچیده‌ای هستند که از حوصله این کتاب خارج است بنابراین دو روش مرتب‌سازی حبابی و مرتب‌سازی انتخابی (Selection Sort) ارائه می‌شود.

۱-۷-۱۱ روش مرتب‌سازی حبابی (Bubble Sort)

این روش ساده‌ترین روش مرتب‌سازی است و هر عضو از آرایه با اعضای بعدی آرایه مقایسه می‌شود و در صورت نیاز، عمل تعویض صورت می‌گیرد. در این روش چون هر عضو با سایر اعضای بعد از آن مقایسه می‌شود در آرایه‌های بزرگ زمان زیادی برای مرتب شدن داده‌ها مصرف می‌شود بنابراین توصیه می‌شود در صورت کوچک بودن ابعاد یک آرایه از این روش استفاده کنید. در زیر دستوراتی را می‌بینید که می‌تواند یک آرایه عددی از نوع صحیح با ۱۰ عضو را که قبلاً پر شده است به روش حبابی و به صورت صعودی مرتب کند (با فرض این که 1 Option Base است).

```
Private Sub cmdsort_Click()
```

```
Dim i As Integer
```

```
Dim j As Integer
```

```
Dim temp As Integer
```

```
For i = 1 To 9
```

```
For j = 1 To 9
```

```
If myarray(j) > myarray(j + 1) Then
```



```
temp = myarray(j)
myarray(j) = myarray(j + 1)
myarray(j + 1) = temp
```

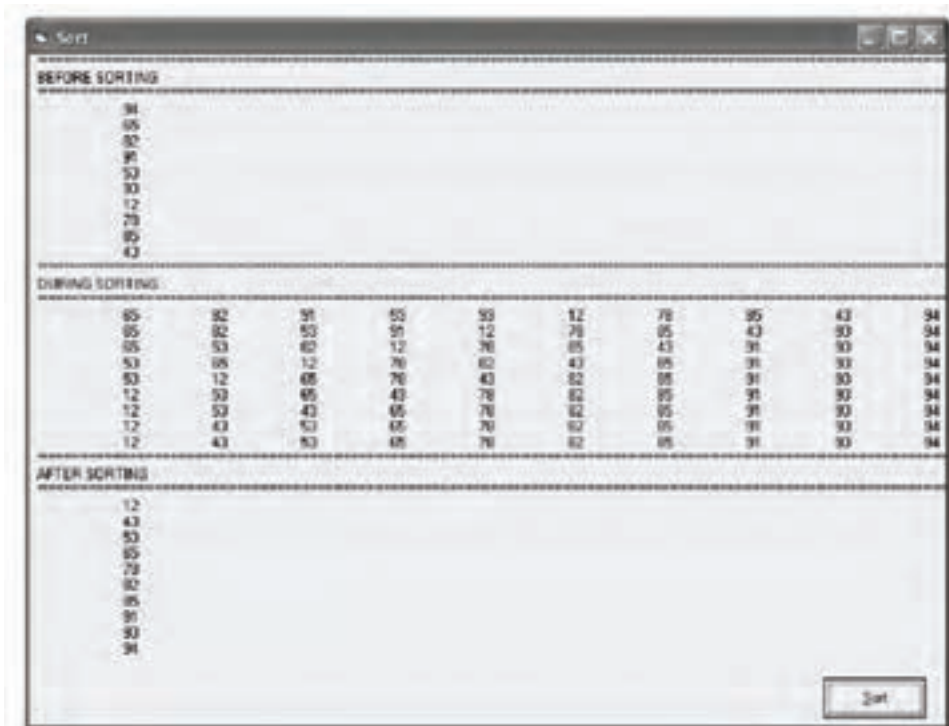
End If

Next j

Next i

End Sub

در این رویداد عمل مرتب‌سازی با دو حلقه که $n-1$ بار تکرار می‌شوند (n تعداد اعضای آرایه است) انجام می‌شود. در واقع به‌وسیله یک If در داخل حلقه دوم مقدار هر عضو با عضو بعدی مقایسه می‌شود و در صورت بزرگ‌تر بودن عضو که اندیس آن j است مقدار آن با مقدار عضو که اندیس آن $j+1$ است تعویض می‌شود. با ادامه روند عملیات، این کار آنقدر تکرار می‌شود تا آرایه مرتب شود؛ می‌توانید روند انجام عملیات را در شکل ۹-۱۱ مشاهده کنید: در ابتدا، مقادیر موجود در آرایه را قبل از مرتب شدن و در انتها، همان آرایه را به‌صورت مرتب شده می‌بینید در بخش میانی نیز اعضای آرایه را در هر بار اجرای حلقه اول (حلقه با شمارنده i) مشاهده می‌کنید. اگر به روند تغییر مکان اعداد توجه کنید نحوه مرتب‌سازی را کاملاً درک خواهید کرد.



شکل ۹-۱۱

تمرین:



پروژه‌ای طراحی کنید که ده عدد را از کاربر دریافت کند و پس از ذخیره‌سازی در یک آرایه، آرایه را به صورت نزولی مرتب کرده و نمایش دهد.

۲-۷-۱۱ روش مرتب‌سازی انتخابی

در این روش ابتدا کوچک‌ترین عضو آرایه تعیین و با عضو اول آرایه جابه‌جا می‌شود. در مرحله بعد کوچک‌ترین عضو آرایه در بین عضو دوم تا آخرین عضو آرایه پیدا شده و با عضو دوم آرایه جابه‌جا می‌شود و به همین ترتیب الی آخر تا آرایه مرتب شود.

```
Private Sub cmdsort_Click()
```

```
Dim length As Integer
```

```
Dim index As Integer
```

```
Dim min As Long
```

```
length = UBound(myarray)
```

```
For i = 1 To length
```

```
min = myarray(i)
```

```
index = i
```

```
For j = i + 1 To length
```

```
If min > myarray(j) Then
```

```
min = myarray(j)
```

```
index = j
```

```
End If
```

```
Next j
```

```
If index <> i Then
```

```
temp = myarray(i)
```

```
myarray(i) = myarray(index)
```

```
myarray(index) = temp
```

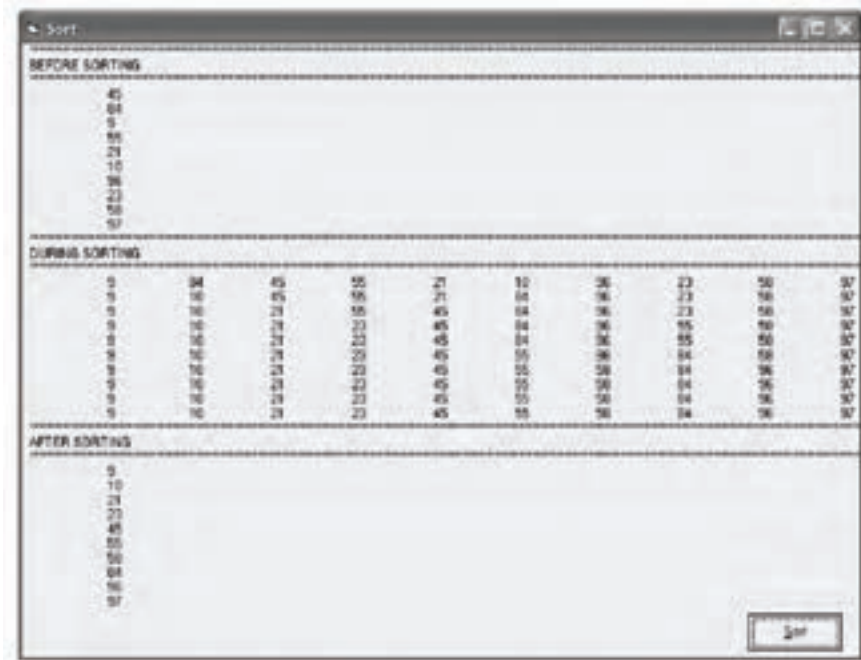
```
End If
```

```
Next i
```

```
End Sub
```

در این رویداد با استفاده از دو حلقه For تودرتو عملیات مرتب‌سازی انجام می‌شود در ابتدا نخستین حلقه For اولین عضو آرایه را به عنوان کوچک‌ترین مقدار در متغیر min ذخیره می‌کند. سپس حلقه For دوم این مقدار را با سایر اعضا یعنی اعضای دوم تا آخرین عضو مقایسه می‌کند اگر مقدار عضوی از مقدار min کمتر باشد مقدار کوچک‌تر را در متغیر min و اندیس آن را در متغیر index ذخیره می‌کند و پس از خاتمه اجرای این حلقه اگر مقدار کوچک‌تری پیدا شده باشد (index < i) آنگاه با استفاده از یک دستور If جای عضو اول با عضوی کوچک‌تر که اندیس آن در متغیر index ذخیره شده است عوض می‌شود.

در این مرحله با رسیدن Next i اجرای دستورات به ابتدای حلقه اول منتقل می‌شود. در این مرحله همان عملیات برای عضو دوم آرایه اجرا خواهد شد و به همین شکل آرایه مرتب می‌شود (شکل ۱۰-۱۱).



شکل ۱۰-۱۱

۸-۱۱ روش‌های جستجوی داده‌ها در آرایه‌ها

گاهی اوقات لازم است تا داده‌ای را میان مجموعه‌ای از اطلاعات موجود جستجو

کرده و مورد دستیابی قرار دهیم. یکی از دلایل استفاده از آرایه‌ها جستجوی سریع‌تر و راحت‌تر داده‌ها در میان انبوهی از اطلاعات است.

جستجوی داده در یک آرایه مانند مرتب‌سازی از روش‌های مختلفی امکان‌پذیر است که از مهم‌ترین آن‌ها می‌توان روش جستجوی خطی و روش جستجوی دودویی را نام برد.

۱-۸-۱ روش جستجوی خطی (Linear Search)

این روش یکی از ساده‌ترین روش‌های جستجو است در این روش برای پیدا کردن یک مقدار، مقدار مربوطه را یک به یک با اعضای آرایه مورد مقایسه قرار داده و در صورت پیدا شدن اطلاعات مورد نظر جستجو خاتمه می‌یابد و اگر مقدار مورد نظر در هیچ یک از اعضای آرایه پیدا نشود در آن صورت نتیجه جستجو منفی خواهد بود و مقدار مربوطه در آرایه وجود نخواهد داشت.

به‌عنوان مثال فرض کنید می‌خواهیم یک عدد را در آرایه‌ای با ۲۰ عضو جستجو کنیم. تابع mysearch می‌تواند با دریافت یک عدد و آرایه مربوطه، عدد مربوط را در آرایه جستجو کند و در صورت وجود عدد در آرایه، مقدار Yes و در غیر این صورت مقدار No را برگرداند.

```
Function mysearch(myarray() As Integer, mynumber As Integer) _ As Boolean
```

```
Dim i As Integer
```

```
For i = 0 To 19
```

```
    If mynumber = myarray(i) Then
```

```
        mysearch = True
```

```
    Exit Function
```

```
End If
```

```
mysearch = False
```

```
Next i
```

```
End Function
```

در این دستورات با استفاده از حلقه For و یک دستور If تک‌تک اعضای آرایه با مقدار متغیر mynumber مقایسه می‌شوند و در صورت پیدا شدن مقدار مشابه، مقدار درست (True) برگشت داده می‌شود و پس از آن به وسیله Exit Function خروج از تابع اتفاق می‌افتد.

اما اگر تمام اعضای آرایه بررسی شوند و مقداری برابر با mynumber پیدا نشود با بازگشت دادن مقدار نادرست (False) این مسأله را مشخص می‌کند.
به‌عنوان مثالی دیگر فرض کنید می‌خواهیم یک نام را در یک آرایه رشته‌ای با ۲۰ عضو مورد جستجو قرار دهیم، در این صورت تابع mysearch به این صورت در می‌آید:

```
Function mysearch(myarray() As String, myname As String) As Boolean
```

```
    Dim i As Integer
```

```
    For i = 0 To 19
```

```
        If StrComp(myarray(i), myname, 1) Then
```

```
            mysearch = True
```

```
            Exit Function
```

```
        End If
```

```
    mysearch = False
```

```
    Next i
```

```
End Function
```

در تابع فوق نیز نحوه انجام عملیات مانند تابع قبل است و فقط برای پیدا کردن نام مورد نظر (myname) در آرایه از تابع StrComp استفاده می‌شود.



تمرین:

پروژه‌ای طراحی کنید که ده عدد را که به صورت تصادفی تولید شده‌اند در آرایه قرار دهد و به کاربر نشان دهد سپس کاربر عدد دلخواهی را در یک کنترل کادر متن وارد کند و با کلیک روی دکمه فرمان Search به روش جستجوی خطی به جستجوی عدد مورد نظر پردازد.

۲-۸-۱۱ روش جستجوی دودویی (Binary Search)

روش جستجوی خطی در آرایه‌های بزرگ، بسیار کند و وقت‌گیر است، روش جستجوی دو دویی با استفاده از این راه‌حل ساده، تعداد مقایسه‌ها را به مقدار زیادی کاهش می‌دهد.

در این روش ابتدا آرایه را مرتب کرده سپس عضو میانی آرایه با مقدار مورد جستجو، مقایسه می‌شوند، اگر مقدار مورد جستجو کوچک‌تر از عضو میانی باشد جستجو در

قسمت پایینی عضو میانی صورت می‌گیرد و اگر مقدار مورد جستجو بزرگ‌تر از عضو میانی باشد جستجو در قسمت بالایی عضو میانی انجام می‌گیرد سپس همین اعمال برای نیمه پایینی و یا بالایی در آرایه انجام می‌شود و بدین صورت تا زمان پیدا شدن مقدار مورد نظر یا با نصف شدن تعداد اعضای باقیمانده عملیات ادامه می‌یابد.
بنابراین می‌توانید تابع Binary Search را به این صورت بنویسید:

Public Function mysearch(myarray() As Integer, mynum As Integer) As Boolean

Dim i As Integer, intlow As Integer

Dim inthigh As Integer, intmid As Integer

intlow = LBound(myarray)

inthigh = UBound(myarray)

Do While (inthigh >= intlow)

intmid = Int((intlow + inthigh) / 2)

If mynum = myarray(intmid) Then

mysearch = True

Exit Function

ElseIf myarray(intmid) < mynum Then

intlow = intmid + 1

Else

inthigh = intmid - 1

End If

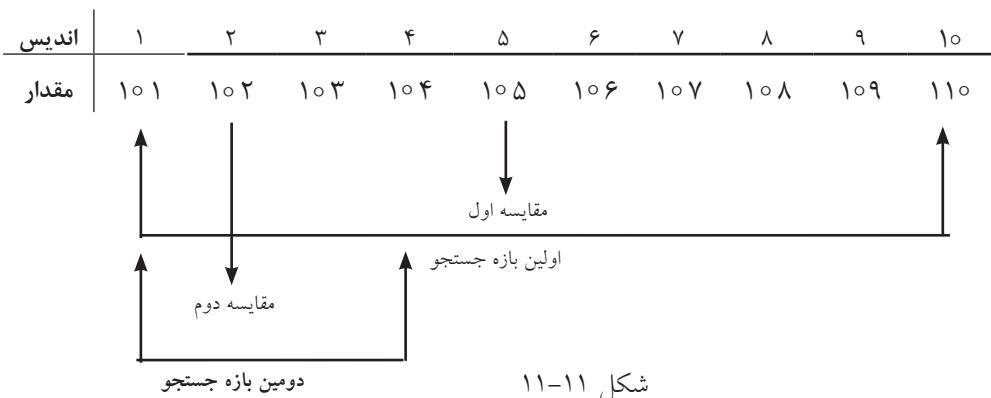
Loop

mysearch = False

End Function

این تابع با دریافت یک آرایه و عدد مورد نظر برای جستجو تعریف می‌شود پس از تعریف متغیرهای مورد نظر، مقدار اندیس بالایی و پایینی در آرایه را به وسیله توابع LBound, UBound در متغیرهای inthigh و intlow قرار می‌دهد. سپس یک حلقه Do while عملیات جستجو را انجام می‌دهد. ابتدا مقدار اندیس میانی در آرایه محاسبه می‌شود برای این کار اندیس پایینی با بالایی جمع شده و مقدار صحیح تقسیم آن‌ها بر عدد ۲ توسط تابع Int به دست می‌آید.

فرض کنید که آرایه دارای ۲۰ عضو است و اندیس پایینی ۱ است. با این شرایط مقدار `intmid` برابر با ۵ خواهد بود. پس از محاسبه مقدار اندیس عضو میانی به وسیله فرمان `if` تساوی عضو میانی (`myarray(5)` با عدد مورد جستجو `mynum` بررسی می‌شود، در صورت مساوی بودن آن‌ها با یکدیگر مقدار `True` بازگشت داده خواهد شد و به وسیله فرمان `Exit Function` اجرای برنامه به محل فراخوانی منتقل می‌شود اما اگر تساوی برقرار نباشد شرط موجود در `ElseIf` بررسی می‌شود. اگر در این مرحله مقدار عضو (`myarray(5)` کوچک‌تر از مقدار `mynum` باشد چون آرایه مرتب است بنابراین مقدار اندیس پایینی به وسیله فرمان `intlow = intmid + 1` به مقدار بعد از اندیس عضو میانی (یعنی ۶) تغییر پیدا خواهد کرد و حلقه در اجرای دوباره دستورات، عناصر موجود در بین اندیس ۶ و ۱۰ را مورد جستجو قرار می‌دهد؛ اما اگر شرط موجود در `ElseIf` غلط باشد بنابراین عنصر مورد جستجو در نیمه پایینی آرایه قرار دارد و با استفاده از بخش `Else` مقدار اندیس بالایی یکی کمتر از اندیس میانی (یعنی ۴) خواهد شد و در اجرای دوباره حلقه، عناصر موجود در بین اندیس ۱ و ۴ مورد جستجو قرار می‌گیرند، به این ترتیب با کوچک‌تر شدن بازه جستجو، در صورتی که مقدار مورد نظر در آرایه کشف نشود مقادیر `intlow` و `inhigh` به گونه‌ای تغییر می‌کنند که نتیجه شرط `inhigh >= intlow` نادرست شود که در نتیجه اجرای حلقه خاتمه یافته و مقدار `False` بازگشت داده می‌شود مثلاً فرض کنید که می‌خواهیم عدد ۲۰ را در آرایه ۱۰ عضوی (مقدار اعضای آن از ۱۰۱ تا ۱۰ هستند) جستجو کنیم. عملکرد تابع به این صورت نمایش داده می‌شود:

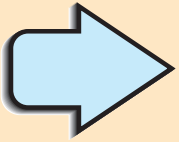


شکل ۱۱-۱۱



تمرین:

پروژه تمرین قبل را با روش جستجوی دودویی انجام دهید.



Learn in English

Array Variables

Much of the time, you just want to assign a single value to a variable you've declared.

A variable containing a single value is a scalar variable.

Other times, it's convenient to assign more than one related value to a single variable.

Then you can create a variable that can contain a series of values.

This is called an array variable. Array variables and scalar variables are declared in the same way,

except that the declaration of an array variable uses parentheses () following the variable name.

In the following example, a single-dimension array containing 11 elements is declared:

```
Dim A(10) As Integer
```


واژه‌نامه

| | |
|----------------|-------------------|
| Array | آرایه |
| Binary Search | جستجوی دودویی |
| Bound | مرز، محدوده |
| Bubble Sort | مرتب‌سازی حبابی |
| Convenient | راحت، مناسب |
| Delimiter | حائل |
| Dimension | اندازه، بعد |
| Dynamic | پویا |
| Element | عنصر |
| Erase | پاک کردن |
| Expression | عبارت |
| Join | متصل کردن |
| Limit | حد، اندازه |
| Linear Search | جستجوی خطی |
| Random | تصادفی |
| Related | مربوط |
| Scalar | عددی |
| Selection Sort | مرتب‌سازی انتخابی |
| Split | از هم جدا کردن |
| Subscript | اندیس، زیرنویس |

خلاصه مطالب

- از آرایه‌ها برای ذخیره‌سازی و نگهداری آسان‌تر داده‌ها، مرتب‌سازی و جستجوی سریع اطلاعات استفاده می‌شود.
- آرایه‌ها به دو دسته کلی، آرایه‌ها با ابعاد ثابت و آرایه‌های پویا یا با ابعاد متغیر تقسیم می‌شوند.
- به وسیله فرمان Option Base می‌توان شماره اولین عنصر آرایه را تعیین کرد.
- در آرایه‌های پویا (Dynamic) تعداد اعضای آرایه با توجه به تعداد مورد نیاز معین می‌شوند اما در آرایه‌های ثابت تعداد اعضای آرایه بعد از تعریف آن قابل تغییر نیست.
- برای قابل استفاده شدن یک آرایه پویا پس از تعریف آرایه از دستور ReDim برای تعیین ابعاد آن استفاده می‌شود.
- در صورتی که بخواهید مقادیر موجود در یک آرایه پویا در زمان تغییر ابعاد آن از بین

نرود از کلمه کلیدی Preserve استفاده کنید.

- به وسیله فرمان Erase می توان آرایه های پویا را حذف کرد.
- مقادیر موجود در یک آرایه با ابعاد ثابت به وسیله فرمان Erase از بین خواهند رفت اما فضاهای مربوط به اعضای آرایه در حافظه باقی می ماند.
- هنگام ارسال آرایه ها به رویه ها به طور پیش فرض از حالت ارسال با مرجع استفاده می شود.
- روش های مرتب سازی آرایه ها عبارتند از: مرتب سازی حبابی، مرتب سازی به روش Shell Sort، روش مرتب سازی Quick Sort، روش مرتب سازی انتخابی (Selection Sort) و روش مرتب سازی Insertion Sort.
- برای جستجوی اطلاعات در یک آرایه می توان از روش جستجوی خطی یا دودویی استفاده کرد.
- توابع UBound و LBound به ترتیب دامنه بالایی و پایینی آرایه مورد نظر را محاسبه می کنند.
- تابع Split می تواند یک عبارت رشته ای را به صورت زیر رشته های جداگانه در یک آرایه یک بعدی ذخیره کند.
- تابع Join برخلاف تابع Split، می تواند مقادیر رشته ای موجود در یک آرایه رشته ای را به صورت یک عبارت رشته ای واحد در آورد.

آزمون نظری

۱- در ویژوال بیسیک اندیس اول یک آرایه به‌طور پیش فرض از چه شماره‌ای آغاز می‌شود؟

الف- ۱ ب- صفر ج- ۱ د- ۲

۲- پس از تعریف یک آرایه پویا به‌وسیله کدام دستور آرایه قابل استفاده می‌شود؟

الف- Dim ب- Static ج- Erase د- ReDim

۳- به‌وسیله کدام فرمان می‌توان شماره اندیس اولین عضو آرایه را تعیین کرد؟

الف- Option Compare Text ب- Option Explicit

ج- Option Base د- Option Keyword

۴- آرایه (۵, ۱۰) No چند عضو دارد؟ (Option Base 1)

الف- ۳۶ ب- ۵۰ ج- ۴۵ د- ۴۰۰

۵- فرمان Erase می‌تواند آرایه‌های را حذف کند.

الف- پویا ب- با ابعاد ثابت

ج- رشته‌ای د- گزینه‌های الف و ب صحیح هستند.

۶- اگر (۱۰) name باشد حاصل عبارت (name) UBound چیست؟

الف- ۸ ب- ۹ ج- ۱۰ د- ۱۱

۷- آرایه (Color (1 To 5, 3 To 6) چند عضو دارد؟

الف- ۲۰ ب- ۲۵ ج- ۳۰ د- ۳۵

۸- کدام تابع می‌تواند اعضای یک آرایه رشته‌ای را به یک عبارت رشته‌ای واحد تبدیل کند؟

الف- Split ب- Join ج- Ubound د- Preserve

۹- در صورتی که بخواهیم در زمان تغییر ابعاد یک آرایه پویا مقادیر قبلی آرایه از بین

نرود از کدام کلمه کلیدی همراه با دستور ReDim استفاده می‌کنیم؟

الف- Dim ب- ParamArray ج- Para د- Preserve

۱۰- کدام تابع می‌تواند یک عبارت رشته‌ای را به‌صورت کلمات جداگانه در یک آرایه

ذخیره کند؟

الف- Split ب- Join ج- Preserve د- Filter

۱۱- Which of the following answer can be use to create a single dimension array With 5 elements?

a- Dim no (5) As Integer

b- Dim no (4) As Integer

c- Dim no [5] As Integer

d- Dim no [4] As Integer

- ۱۲- آرایه را تعریف کنید و انواع آن را نام ببرید.
- ۱۳- دستور Erase, Split و Join را با ذکر مثال توضیح دهید.
- ۱۴- نحوه تعریف و استفاده از آرایه‌های استاتیک و پویا را با ذکر مثال بیان کنید.
- ۱۵- توابع UBound و LBound را با ذکر مثال توضیح دهید.
- ۱۶- نحوه مرتب سازی آرایه‌ها با روش حبابی و انتخابی را بیان کنید.
- ۱۷- کاربرد دستور Option Base را با ذکر مثال توضیح دهید.
- ۱۸- نحوه ارسال آرایه‌ها به رویه‌های تابعی و فرعی را بیان کنید.
- ۱۹- انواع روش‌های جستجوی داده‌ها در آرایه‌ها را توضیح دهید.

آزمون عملی

- ۱- رویه‌ای بنویسید که یک آرایه 5×5 را با اعضای تصادفی ایجاد کرده سپس مجموع هر سطر و هر ستون آرایه را به‌طور جداگانه به همراه خود آرایه نمایش دهد.
- ۲- رویه‌ای بنویسید که دو آرایه دلخواه را دریافت کرده و آرایه حاصل ضرب آن دو را به همراه دو آرایه نمایش دهد.
- ۳- رویه‌ای بنویسید که دو آرایه یک بعدی دلخواه را دریافت کرده و سپس آرایه حاصل جمع، حاصل ضرب و حاصل تفریق آن دو را در آرایه‌های جداگانه‌ای ذخیره کند.
- ۴- رویه‌ای بنویسید که یک آرایه عددی با ۲۰۰ عضو را به‌صورت نزولی مرتب کرده و نمایش دهد.

توانایی استفاده از جلوه‌های گرافیکی و چاپ در ویژگی‌های بیسیک

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

- ۱- مفاهیم مربوط به سیستم مختصات را بدانند.
- ۲- توانایی تغییر مختصات را به منظور انجام ترسیمات داشته باشد.
- ۳- توانایی به‌کارگیری متدهای گرافیکی زیر را داشته باشد:
PSet , Line , Circle , Point , Cls , Print , TextHeight , TextWidth
- ۴- توانایی استفاده از خصوصیات گرافیکی زیر را داشته باشد:
CurrentX , CurrentY , AutoRedraw , DrawMode , DrawStyle , DrawWidth , FillStyle
- ۵- توانایی استفاده از توابع QBColor و RGB را داشته باشد.
- ۶- توانایی استفاده از امکانات چاپ، خصوصیات و متدهای آن را در برنامه‌ها داشته باشد.
- ۷- توانایی استفاده از امکانات چندرسانه‌ای و کنترل MCI و MAM
- ۸- توانایی استفاده از شیء Picture و رویه‌های SavePicture و LoadPicture

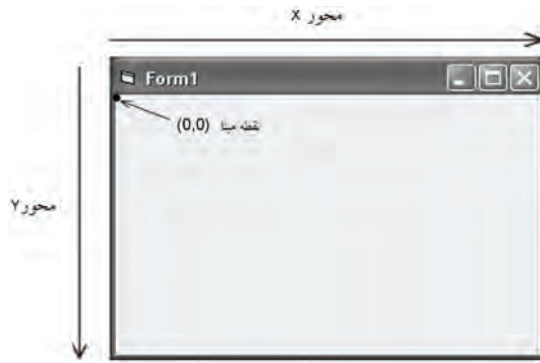
کلیات

تاکنون چگونگی طراحی و ساخت رابط‌های گرافیکی را با استفاده از فرم‌ها و بعضی از کنترل‌ها آموختید؛ اما گاهی اوقات وجود فرم‌ها و کنترل‌ها بدون جلوه‌های گرافیکی محیط نرم‌افزار را خسته‌کننده و غیر قابل استفاده می‌کند. با استفاده از جلوه‌های گرافیکی نظیر رنگ، نمودارهای گرافیکی و تصاویر متحرک می‌توانید به کاربر در مشاهده و درک بهتر گزارشات و نتیجه محاسبات کمک کنید.

ویژوال بیسیک قابلیت بالایی در استفاده از جلوه‌های گرافیکی در اختیار شما قرار می‌دهد و می‌توانید این جلوه‌های گرافیکی را با دو روش ایجاد کنید: کنترل‌های گرافیکی مثل Line, Shape و متدهای گرافیکی مثل Circle, PSet, Line و ...

۱-۱۲ مفهوم سیستم مختصات در ویژوال بیسیک

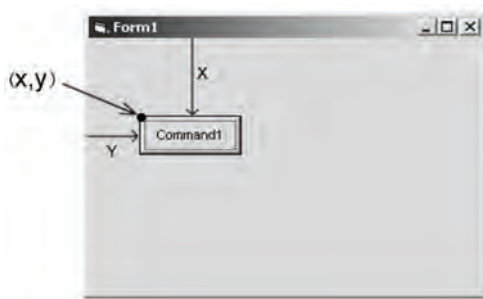
در ویژوال بیسیک نیز مانند ریاضیات و هندسه برای انجام هر نوع ترسیمی از سیستم مختصات استفاده می‌شود، هم‌چنین از دو بعد یا محور مختصات برای تعیین موقعیت ترسیمات استفاده می‌شود (شکل ۱-۱۲).



شکل ۱-۱۲

مختصات هر نقطه به صورت (X,Y) تعیین می‌شود مقدار X موقعیت نقطه را در طول محور X و مقدار Y موقعیت نقطه را در طول محور Y بیان می‌کند که مقدار شروع در هر یک از محورها صفر است. دقت داشته باشید که محورهای مختصات در ویژوال بیسیک با محورهای مختصات در ریاضیات متفاوت است. در شکل ۱-۱۲ نحوه قرار گرفتن محورها بر نقطه مبنا قابل مشاهده است. مختصات نقطه مبنا $(0,0)$ است که در گوشه بالا و چپ شیء مربوطه که معمولاً فرم است قرار دارد.

وقتی کنترل را جابه‌جا کرده یا اندازه آن را تغییر می‌دهید، از سیستم مختصات فرمی که کنترل در آن قرار دارد، استفاده می‌کنید. در واقع کنترل‌ها و هرگونه ترسیمات گرافیکی از سیستم مختصات شیء که در آن رسم می‌شوند، تبعیت می‌کنند. اگر کنترلی در روی فرم قرار گیرد موقعیت و اندازه کنترل از سیستم مختصات فرم پیروی می‌کند و اگر خطی در یک جعبه تصویر PictureBox رسم شود، موقعیت و اندازه خط از سیستم مختصات کنترل جعبه تصویر استفاده می‌کند.



شکل ۲-۱۲ تعیین موقعیت یک کنترل به وسیله سیستم مختصات

برای تعریف موقعیت ترسیمات گرافیکی به وسیله محورهای واحدهای اندازه‌گیری استفاده می‌شود که به آن مقیاس می‌گویند. در ویژوال بیسیک هر محور در سیستم مختصات می‌تواند مقیاس خاص خود را داشته باشد.

۲-۱۲ تغییر سیستم مختصات

می‌توانید سیستم مختصات یک شیء خاص مثل فرم را به وسیله خصوصیت Scale یا متد Scale روی مقادیر مورد نظرتان تنظیم کنید. برای انجام این کار می‌توانید از مقیاس پیش‌فرض استفاده کرده یا یکی از مقیاس‌های استاندارد را انتخاب کنید یا این که یک مقیاس جدید را ایجاد کنید. با تغییر مقیاس سیستم مختصات می‌توانید اندازه و موقعیت ترسیمات گرافیکی را روی فرم با توجه به نیازتان به آسانی تنظیم کنید.

هر فرم یا بعضی از کنترل‌ها مانند PictureBox چندین خصوصیت Scale، نظیر ScaleMode، ScaleWidth، ScaleHeight، ScaleTop، ScaleLeft و یک متد Scale دارند که به وسیله آن‌ها می‌توانید سیستم مختصات خود را تعریف کنید. مقیاس پیش‌فرض twip است.

همان‌طور که قبلاً هم اشاره کردیم هر ۵۶۷ twip برابر با یک سانتی متر است. برای انتخاب یک مقیاس استاندارد می‌توانید یکی از مقادیر موجود در جدول ۱-۱۲ را برای خصوصیت ScaleMode فرم یا کنترل مورد نظر خود در نظر بگیرید.

جدول ۱-۱۲ مقادیری که خصوصیت ScaleMode کسب می‌کند.

| ثابت رشته‌ای | ثابت عددی | توضیح (نوع مقیاس) |
|---------------|-----------|--|
| vbUser | ۰ | مقادیر تعریفی کاربر در خصوصیات ScaleWidth, ScaleHeight, ScaleLeft, ScaleTop استفاده می‌شوند. |
| vbTwips | ۱ | twip |
| vbPoints | ۲ | Point (۱ Inch = ۷۲Point) |
| vbPixels | ۳ | Pixel (یک کوچک‌ترین واحد نمایشی در صفحه نمایش یا چاپگر است و تعداد آن‌ها در هر اینچ به مقدار وضوح تصویر بستگی دارد.) |
| vbCharacters | ۴ | Character |
| vbInches | ۵ | Inch |
| vbMillimeters | ۶ | Milimeter |
| vbCentimeters | ۷ | Centimeter |

اگر بخواهید مختصات نقطه مبنا را تغییر دهید یا مقیاس جدید را در یک کنترل یا فرم ایجاد کنید می‌توانید از خصوصیات ScaleWidth, ScaleHeight, ScaleLeft و ScaleTop استفاده کنید. خصوصیات ScaleLeft و ScaleTop با دریافت مقادیر عددی، مختصات نقطه مبنا را در کنترل و فرم معین می‌کنند. مقدار پیش فرض برای این دو خصوصیت صفر است. مقدار این خصوصیات را می‌توانید از طریق پنجره خصوصیات تغییر دهید و یا با استفاده از نوشتن کد در رویدادها و رویه‌های مورد نظر این کار را انجام دهید. شکل کلی استفاده از این خصوصیات به صورت زیر است:

object.ScaleLeft = value

object.ScaleTop = value

منظور از object نام یک فرم یا کنترل است و استفاده از آن اختیاری است و در صورتی که از آن استفاده نشود فرمی که فوکوس دارد در نظر گرفته خواهد شد. value یک مقدار عددی است که مختصات نقطه مبنا را تعیین می‌کند در صورت

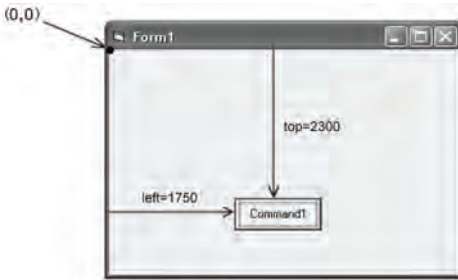
عدم استفاده از value می‌توانید مقدار فعلی مختصات نقطه مبنا را به دست آورید. به عنوان مثال فرض کنید در یک فرم همراه با یک کنترل دکمه فرمان این مقادیر تنظیم شده است:

```
Form1.ScaleMode = 1
Command1.Top = 2300
Command1.Left = 1750
```

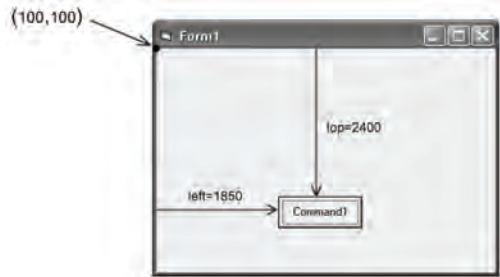
اگر بخواهیم در این فرم مختصات نقطه مبنا را (100, 100) قرار دهیم در این صورت خصوصیات مربوطه را به صورت زیر تنظیم می‌کنیم:

```
Form1.ScaleTop = 100
Form1.ScaleLeft = 100
```

در این صورت مقدار خصوصیت ScaleMode در Form1 به طور خودکار روی مقدار صفر قرار می‌گیرد و در ضمن خصوصیت Top و Left در دکمه فرمان به طور خودکار به ترتیب روی مقادیر 2400° و 1850° تنظیم می‌شوند و همان طور که از مقایسه مقادیر در دو حالت مشاهده می‌کنید مقادیر (x,y) برای کنترل دکمه فرمان با توجه به نقطه مبناي جدید (1000, 1000) تنظیم می‌شود. این دو حالت را می‌توانید در شکل‌های ۳-۱۲ و ۴-۱۲ مشاهده کنید.



شکل ۳-۱۲



شکل ۴-۱۲

به علاوه به وسیله این دو خصوصیت می‌توانید از مقدار مختصات نقطه مبنا مطلع شوید در رویه بعد، پس از تغییر مقدار این دو خصوصیت، مقادیر جدید در یک کادر پیغام نمایش داده می‌شوند.

```
Private Sub cmdshow_Click ()
```

```
Form1.ScaleLeft = 150
```

```
Form1.ScaleTop = 180
```

```
MsgBox " ScaleLeft = " + Str ( ScaleLeft ) + " ScaleTop = " + Str ( ScaleTop )
```

```
End Sub
```



نکته در صورت عدم استفاده از بخش object، فرمی که فوکوس دارد در نظر گرفته خواهد شد.

علاوه بر تغییر مختصات نقطه مبنا، می‌توانید مقیاس سیستم مختصات را نیز تغییر دهید. خصوصیات ScaleWidth و ScaleHeight واحد اندازه‌گیری در محورهای X و Y را تعیین می‌کنند. مقدار این خصوصیات را می‌توانید از طریق پنجره خصوصیات یا با نوشتن کد مناسب تغییر دهید. شکل کلی استفاده از این خصوصیات به صورت زیر است:

```
object.ScaleHeight = value
```

```
object.ScaleWidth = value
```

منظور از object نام یک فرم یا کنترل است و value یک مقدار عددی است که واحد اندازه‌گیری را در محورها تعیین می‌کند. در صورتی که از object استفاده نشود، فرمی که فوکوس دارد در نظر گرفته خواهد شد و در صورت عدم استفاده از بخش value می‌توانید مقادیر ذخیره شده در این دو خصوصیت را به دست آورید.
به عنوان مثال به رویه زیر توجه کنید:

```
Private Sub cmdscale_Click()
```

```
Form1.ScaleWidth = 1000
```

```
Form1.ScaleHeight = 500
```

```
End Sub
```

با اجرای رویه فوق معیار اندازه‌گیری در محور افقی (X)، یک هزارم ($\frac{1}{1000}$) عرض داخلی (Width) فرم و در محور عمودی (Y)، یک پانصدم ($\frac{1}{500}$) ارتفاع داخلی (Height) فرم خواهد بود.



نکته خصوصیات ScaleWidth و ScaleHeight واحدها را با توجه به ابعاد داخلی فرم یا شیء مربوطه تعیین می‌کنند. این ابعاد شامل حاشیه‌ها یا منوها یا نوار عنوان نمی‌شوند. این دو خصوصیت همواره در رابطه با بخش قابل ترسیم داخل فرم یا شیء مربوطه تعریف می‌شوند.

به عنوان مثال یک فرم همراه با یک دکمه فرمان با مشخصات زیر را ایجاد کنید:

```
Form1.BorderStyle = None
```

Form1.Height = 3000

Form1.Width = 4200

Form1.ScaleMode = twip

Command1.Height = 400

Command1.Width = 1200

اگر تنظیمات فوق را به ترتیب انجام دهید سپس مقدار خصوصیات ScaleHeight و ScaleWidth را ملاحظه کنید خواهید دید که مقدار این دو خصوصیت مانند مقدار ارتفاع و عرض فرم است زیرا فرم شما بدون حاشیه و نوار عنوان است و تمام فضای فرم، فضای قابل دسترس محسوب می‌شود. اما اگر مقدار خصوصیت BorderStyle را Sizeable قرار دهید مقدار دو خصوصیت ScaleHeight و ScaleWidth مقادیر کمتری خواهند بود زیرا بخشی از ابعاد فرم به نوار عنوان و حاشیه‌ها داده شده است که جزء فضاهای قابل دسترس نیستند. همچنین معیار اندازه‌گیری در محورها توسط خصوصیت ScaleMode تعیین می‌شود که از نوع twip است.

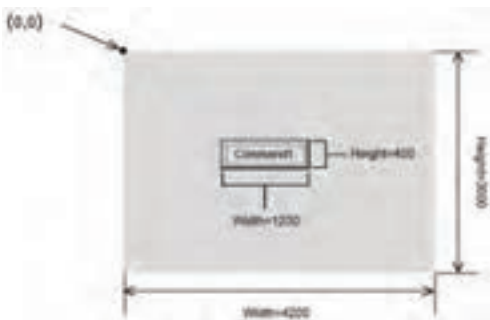
اکنون خصوصیات ScaleHeight و ScaleWidth فرم را روی ۵۰۰ و ۱۰۰۰ تنظیم کنید با تغییر یکی از این خصوصیات، خصوصیت ScaleMode به‌طور خودکار روی مقدار 0-User تنظیم می‌شود پس از تغییر دو خصوصیت فوق، خصوصیات Height و Width کنترل Command1 را مشاهده کنید. همان‌طور که می‌بینید مقدار آن‌ها به ترتیب به ۶۶/۶۶۷ و ۲۸۵/۷۱۴ تغییر کرده است. در واقع برای محاسبه هر یک از این مقادیر به این صورت عمل شده است (شکل ۵-۱۲ و ۶-۱۲).

$$400 \times \frac{500}{3000} = 66/667$$

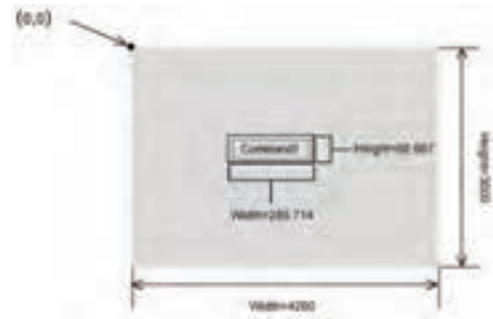
مقدار خصوصیت Command1.Height

$$1200 \times \frac{1000}{4200} = 285/714$$

مقدار خصوصیت Command1.Width



شکل ۵-۱۲



شکل ۶-۱۲



نکته • تنظیم هر یک از خصوصیات Scale، مقدار خصوصیت ScaleMode را به طور خودکار روی ۰ تنظیم می‌کند.

• انتخاب مقیاس بزرگ‌تر از صفر برای خصوصیت ScaleMode مقدار خصوصیات ScaleHeight و ScaleWidth را به طور خودکار روی مقادیر جدید تنظیم می‌کند و مقدار خصوصیات ScaleTop و ScaleLeft را روی صفر تنظیم می‌کند.

هر چهار خصوصیت Scale می‌توانند مقادیر اعشاری و حتی منفی باشند. در صورت استفاده از اعداد منفی برای خصوصیات ScaleWidth و ScaleHeight جهت محورها در سیستم مختصات تغییر می‌کند، مثلاً برای فرم و کنترلی با مشخصات زیر فرمی مطابق شکل ۱۲-۷ خواهید داشت:

```
Form1.ScaleWidth = - 1000
```

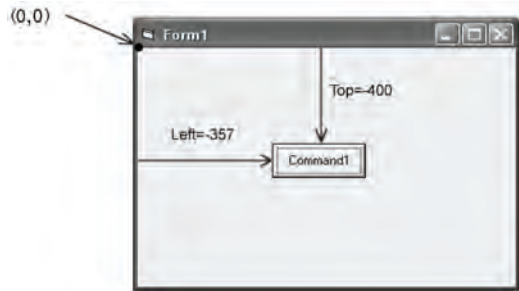
```
Form1.ScaleHeight = - 1000
```

```
Form1.ScaleLeft = 0
```

```
Form1.ScaleTop = 0
```

```
Command1.Left = - 357
```

```
Command1.Top = - 400
```



شکل ۱۲-۷

به‌عنوان آخرین مثال در رابطه با دو خصوصیت ScaleWidth و ScaleHeight به این رویه توجه کنید:

```
Private Sub cmdscale_Click()
```

```
Form1.ScaleMode = 1
```

```
Form1.Width = 4200
```

```
Form1.height = 3000
```

```
Print "Form1.Width = " Form1.Width, "Form1.Height = " Form1.Height
```

```
Print "Form1.ScaleWidth = " Form1.ScaleWidth
```

```
Print "Form1.ScaleHeight = " Form1.ScaleHeight
```

```
Print Form1.ScaleWidth = 1000
```

```
Print Form1.ScaleHeight = 1500
```

```
Print "Form1.Width = " Form1.Width, "Form1.Height = " Form1.Height;
```

```
Print "Form1.ScaleWidth = " Form1.ScaleWidth
```

```
Print "Form1.ScaleHeight = " Form1.ScaleHeight
```

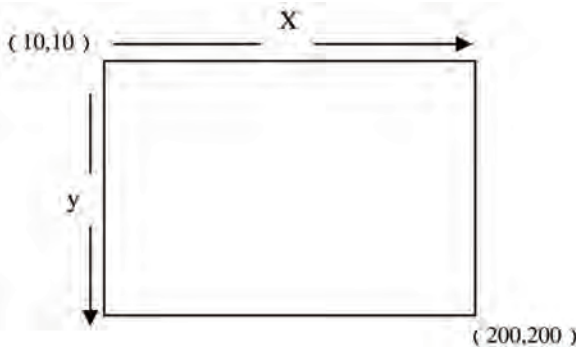
```
End Sub
```

در این رویه رویداد ابتدا نوع مقیاس اندازه‌گیری twip در نظر گرفته شده است سپس عرض و ارتفاع فرم مقداردهی می‌شود. اولین دستور Print ابتدا عرض و ارتفاع فرم یعنی 4200 و 3000 و دو فرمان بعدی نیز مقادیر 4080 و 2595 را برای خصوصیات ScaleWidth و ScaleHeight نمایش می‌دهد. در واقع این دو خصوصیت ابعاد فرم را بدون در نظر گرفتن حاشیه و نوار عنوان فرم نمایش می‌دهند. اما در خطوط بعدی مقدار دو خصوصیت ScaleWidth و ScaleHeight روی 1000 و 1500 تنظیم می‌شود در این حالت خصوصیت ScaleMode نیز به‌طور خودکار روی صفر تنظیم می‌شود. پس از مقداردهی دو خصوصیت ScaleWidth و ScaleHeight، سه دستور Print دیگر اجرا می‌شوند اولین Print همان ابعاد قبلی یعنی 4200 و 3000 را برای ابعاد فرم نمایش می‌دهند اما دو دستور Print بعد مقادیر جدید دو خصوصیت ScaleWidth و ScaleHeight یعنی 1000 و 1500 را نمایش خواهند داد.

روش دیگری که در رابطه با تغییر سیستم مختصات می‌توان به‌کار برد استفاده از متد Scale است. در واقع متد Scale راه‌حل مناسب و آسان‌تری برای تنظیم سیستم مختصات می‌باشد. شکل کلی متد Scale به‌صورت زیر است:

object. Scale (x1,y1) - (x2,y2)

مقادیر عددی $x1,y1$ مختصات گوشه بالایی و سمت چپ شیء و مقادیر عدد $x2,y2$ مختصات گوشه پایینی و سمت راست شیء را مشخص می‌کنند.



شکل ۸-۱۲

object در واقع نام شیء است که می‌خواهید سیستم مختصات آن را تعیین کنید و در صورت عدم استفاده از این قسمت، فرمی که فوکوس دارد به‌عنوان شیء مربوطه در نظر گرفته می‌شود. به‌عنوان مثال دستور زیر سیستم مختصات را در فرم به‌صورت شکل ۸-۱۲ درمی‌آورد.

Form1. Scale (10,10)-(200,200)

در واقع فرمان فوق چهار خصوصیت Scale را به این صورت تنظیم می‌کند:

ScaleWidth = 190

ScaleHeight = 190

ScaleTop = 10

ScaleLeft = 10

۳-۱۲ خصوصیات و متدهای گرافیکی

در این جا لازم است تا چگونگی انجام انواع ترسیمات گرافیکی را بیاموزید. تاکنون این کار را با استفاده از کنترل‌های گرافیکی انجام می‌دادید اما کنترل‌های معرفی شده همواره نیازهای گرافیکی را برطرف نمی‌کنند و در پاره‌ای از مواقع نیز کار را با مشکلات متعدد روبه‌رو می‌سازند. استفاده از متدها و تنظیم خصوصیات گرافیکی نیاز گرافیکی شما را در پروژه‌های برنامه‌نویسی برآورده می‌کند.

۳-۱۲-۱ متد PSet

به وسیله این متد می‌توانید نقاط مورد نظر خود را در مکان‌های مناسب قرار دهید شکل کلی این متد به صورت زیر است:

object. PSet Step (x,y) , color

منظور از object، شیئی است که نقطه روی آن رسم می‌شود. اگر از ذکر آن خودداری کنید فرمی که فوکوس را در اختیار دارد مکان رسم نقطه است. کلمه کلیدی Step نیز اختیاری است و در صورت استفاده از آن هنگام رسم نقطه، مکان ترسیم با توجه به موقعیت جاری در شیئی که در آن ترسیم انجام شده، انتخاب می‌شود.

x, y مقادیر عددی از نوع Single هستند که مختصات محل ترسیم نقطه را مشخص می‌کنند. علاوه بر این می‌توانید رنگ نقطه مورد نظر را به وسیله بخش color تعیین کنید در صورت عدم استفاده از این قسمت، رنگی که در خصوصیت ForeColor شیئی که نقطه در آن رسم می‌شود (به عنوان مثال فرم) در نظر گرفته خواهد شد. جدول مربوط به مقادیر رنگ‌ها در جدول ۲-۱۲ آورده شده است. می‌توانید از ثابت‌های رشته‌ای یا از ثابت‌های عددی در مبنای ۱۶ استفاده کنید.

جدول ۲-۱۲ مقادیر رنگ در ویژوال بیسیک

| ثابت رشته‌ای | ثابت عددی (مبنای ۱۶) | توضیح |
|--------------|----------------------|-----------|
| vbBlack | &H0 | سیاه |
| vbRed | &HFF | قرمز |
| vbGreen | &HFF00 | سبز |
| vbYellow | &HFFFF | زرد |
| vbBlue | &HFF0000 | آبی |
| vbMagenta | &HFF00FF | بنفش |
| vbCyan | &HFFFFFF00 | فیروزه‌ای |
| vbWhite | &8HFFFFFFF | سفید |

به عنوان مثال این دستورات یک نقطه به رنگ آبی روی فرم نمایش می‌دهد:

```
Form1.ForeColor = vbBlue
```

```
PSet ( 1000, 200 )
```

اکنون دستور زیر را در نظر بگیرید:

```
PSet ( 500 , 700 ) , vbCyan
```

این دستور نقطه‌ای را با رنگ فیروزه‌ای در مختصات ۵۰۰ و ۷۰۰ رسم می‌کند حال اگر بلافاصله دستور زیر اجرا شود:

```
PSet Step ( 500 , 700 ) , vbGreen
```

مختصات نقطه مربوطه با توجه به مختصات نقطه رسم شده قبلی محاسبه می‌شود بنابراین دو نقطه روی یکدیگر قرار نخواهند گرفت.

مثال ۱: می‌خواهیم پروژه‌ای طراحی کنیم که ده نقطه به صورت تصادفی روی یک



فرم ترسیم کند.

به این منظور عملیات زیر را به ترتیب انجام دهید:

۱ - برنامه ویژوال بیسیک را اجرا کرده و یک

پروژه از نوع Standard EXE ایجاد کنید.

۲ - نام فرم frmgraphic و عنوان آنرا روی

Geraphic تنظیم کنید سپس یک دکمه فرمان با نام

cmdpset و عنوان &Pset مطابق شکل ۹-۱۲ روی فرم

قرار دهید.



شکل ۹-۱۲

۳ - رویداد click دکمه PSet را به صورت زیر تنظیم کنید:

```
Private Sub cmdpset_Click()
```

```
Dim i As Integer
```

```
Randomize
```

```
For i = 1 To 10
```

```
    PSet (Int(Rnd * 1000 ), Int(Rnd * 2000 ))
```

```
Next i
```

```
End Sub
```

- ۴ - پروژه و فرم را با نام Point ذخیره کنید سپس برنامه را اجرا و روی دکمه PSet کلیک کنید تا ۱۰ نقطه به صورت تصادفی روی فرم ترسیم شوند (شکل ۹-۱۲).
- ۵ - از برنامه خارج شوید و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه Point را به گونه‌ای تغییر دهید که نقاط با رنگ‌های مختلف به طور پیوسته روی فرم نمایش داده شوند.

۲-۳-۱۲ متد Line

به وسیله متد Line می‌توانید انواع خطوط و مستطیل‌های توپر و توخالی را رسم کنید. شکل کلی این متد به صورت زیر است:

```
object.Line Step (x1,y1) - ( x2,y2 ), color, B F
```

عملکرد گزینه‌های object و Step مانند متد PSet است. مقادیر x_1 و y_1 مختصات نقطه ابتدای خط و مقادیر x_2 و y_2 مختصات نقطه انتهایی را در خط تعیین می‌کنند. به وسیله بخش Color نیز می‌توانید رنگ خط را مشخص کنید.

استفاده از حرف B یک مستطیل خالی و استفاده از حرف F به همراه حرف B (BF) یک مستطیل توپر ایجاد می‌کند، البته استفاده از این دو کاراکتر اختیاری است. به عنوان مثال به دستورات زیر توجه کنید:

```
Line (500,800) - (1500 , 1500 )
```

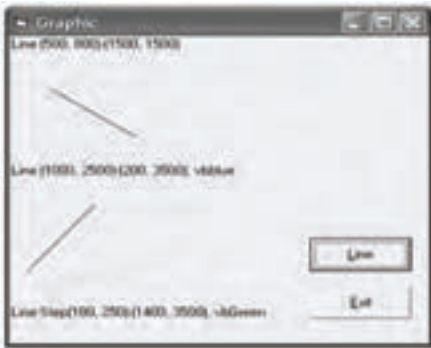
```
Line (1000 , 2500) - ( 200,3500 ),vbBlue
```


Line Step (100,250) – (1400,3500) , vbGreen

Line (410,870) – (1500,1800), vbBlue, B

Line (410,2500) – (2000,3700), vbRed, BF

در این مجموعه دستورات، دستور اول یک خط با رنگی که در خصوصیت ForeColor فرم تعیین شده است ترسیم می‌کند، دستور دوم یک خط با رنگ آبی و دستور سوم نیز یک خط با رنگ سبز ترسیم می‌کنند اما در دستور سوم به دلیل استفاده از Step نقطه شروع از انتهای نقطه‌ای که خط دوم تعیین می‌کند استفاده می‌شود. دستور چهارم و پنجم نیز به ترتیب یک مستطیل توخالی با رنگ آبی و یک مستطیل توپر با رنگ قرمز رسم می‌کنند. در شکل ۱۰-۱۲ و ۱۱-۱۲ نتیجه اجرای این دستورات را مشاهده می‌کنید.



شکل ۱۰-۱۲



شکل ۱۱-۱۲

۱۲-۳-۳ مند Circle

به وسیله این متد می‌توانید انواع دایره، بیضی و کمان را رسم کنید. شکل کلی این متد به صورت زیر است:
 object. Circle step (x,y), radius , color , start , end , aspect
 بخش object و step و color مانند توضیحات ارائه شده در متد PSet است. مقادیر عددی x و y از نوع Single بوده و مختصات مرکز دایره یا بیضی را با توجه به مقدار ScaleMode تعیین می‌کند.

مقدار عددی radius نیز از نوع Single است و مقدار شعاع دایره را براساس مقدار ScaleMode معین می‌کند. مقادیر عددی start و end از نوع Single و اختیاری بوده و موقعیت شروع و خاتمه کمان را برای ترسیم معین می‌کند. مقدار مجاز برای این دو مقدار از -2π رادیان تا 2π رادیان است. در صورت عدم استفاده از این دو مقدار، کمان ترسیمی از صفر تا 2π رادیان در نظر گرفته می‌شود.



نکته • جهت ترسیم کمان، خلاف جهت حرکت عقربه‌های ساعت است. مقدار عددی aspect از نوع Single است و نسبت دو قطر عمودی و افقی را در بیضی معین می‌کند اگر این مقدار ۱ باشد دایره و در غیر این صورت برای مقادیر بزرگ‌تر از ۱، بیضی‌های عمودی و برای مقادیر کوچک‌تر از ۱، بیضی‌های افقی ایجاد می‌شود.

- منظور از عدد π ، ثابت 3.14159265358979 است.
- برای رسم قطاع‌های بیضی یا دایره همراه با خطوط شعاع آن‌ها از مقادیر منفی استفاده کنید.

به عنوان مثال به دستورات زیر توجه کنید:

Circle (700 , 1000),500, vbRed

Circle (600,2000), 400, Pi/2,3* Pi/2

Circle (600,3500), 400, -Pi/6, -Pi/3

Circle (1700,1200), 500, vbRed,,2

Circle (1700,2500), 500, vbRed,,0.2

Circle (1000,4000), 1500,, -0.001, Pi/2,0.4

نتیجه اجرای این دستورات را در شکل‌های ۱۲-۱۲ و ۱۲-۱۳ مشاهده می‌کنید.



شکل ۱۲-۱۲



شکل ۱۲-۱۳



شکل ۱۲-۱۴

تمرین:



پروژه‌ای طراحی کنید که شکل ۱۲-۱۴ را روی یک فرم ترسیم کند.

۴-۳-۱۲ متد Point

این متد با دریافت مختصات یک نقطه، شماره رنگ آن را به صورت یک عدد صحیح از نوع Long بازمی‌گرداند. شکل کلی این متد به صورت زیر است:

object. Point (x,y)

به عنوان مثال فرض کنید که با استفاده از متد Line، یک مستطیل توپر با رنگ قرمز رسم شده است:

```
Line (500,500)-(2000,2500) , vbRed , BF
```

```
Print . Point (700,800)
```

در دستور دوم مختصات نقطه‌ای را که در مستطیل قرمز قرار دارد، به متد Point می‌دهد و در نتیجه مقدار ۲۵۵ که بیانگر رنگ قرمز است توسط متد Point بازگشت می‌یابد و نمایش داده می‌شود.

۵-۳-۱۲ خصوصیات CurrentX و CurrentY

به وسیله این دو خصوصیت می‌توانید موقعیت جاری مکان نما در صفحه ترسیمات را تغییر دهید. خصوصیت CurrentX مختصات مکان‌نما را در جهت محور X و خصوصیت CurrentY مختصات مکان‌نما را در جهت محور Y تعیین می‌کنند. شکل کلی نحوه استفاده از این دو خصوصیت به صورت زیر است:

```
object. CurrentX = x
```

```
object. CurrentY = y
```

مقادیر x و y مقادیر عددی هستند که موقعیت مکان‌نما را در سطح شیء که ترسیمات گرافیکی روی آن انجام می‌گیرد مشخص می‌کنند. اگر مقادیر x و y استفاده نشود می‌توان مختصات فعلی مکان‌نما را به دست آورد. به دستورات زیر توجه کنید:

```
Form1.CurrentX = 2000
```

```
Form1.CurrentY = 1000
```

```
PSet (CurrentX, CurrentY)
```

در این دستورات ابتدا مختصات مکان‌نما به نقطه (۱۰۰۰ و ۲۰۰۰) تغییر پیدا می‌کند سپس به وسیله مقدار این دو خصوصیت نقطه‌ای در همین مختصات رسم می‌شود. به عنوان مثالی دیگر به رویه زیر توجه کنید:

Private Sub cmdshow_Click()

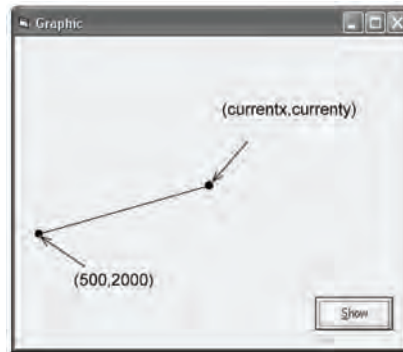
Form1.CurrentX = ScaleWidth / 2

Form1.CurrentY = ScaleHeight / 2

Line (CurrentX, CurrentY)-(500, 2000)

End Sub

با استفاده از خصوصیات ScaleWidth و ScaleHeight ابعاد فرم به دست آمده و با تقسیم آن‌ها بر عدد ۲ مختصات نقطه میانی در فرم محاسبه شده است و فرمان Line، خطی را از این نقطه تا نقطه (۵۰۰، ۲۰۰۰) رسم می‌کند (شکل ۱۵-۱۲).



شکل ۱۵-۱۲



تمرین:

پروژه‌ای طراحی کنید که مختصات مرکز یک دایره و شعاع و رنگ آن را دریافت کرده و دایره موردنظر را رسم کند.

۶-۳-۱۲ متد Cls

شکل کلی این متد به صورت زیر است:

object. Cls

این متد صفحه را کاملاً پاک کرده و مکان نما را به مختصات (۰ و ۰) انتقال می‌دهد. قبلاً بانحوه استفاده از این متد آشنا شده‌اید. به عنوان مثال در رویه زیر ابتدا یک مستطیل توپر با رنگ بنفش و یک دایره با رنگ قرمز رسم می‌شود و بعد از رسم آن‌ها یک کادر پیغام (MessageBox) با جمله Choose ok to clear this background نمایش داده می‌شود که در صورت فشردن دکمه

فرمان OK در کادر پیغام، سطح فرم پاک می‌شود و مقدار صفر برای دو خصوصیت CurrentX و CurrentY در کادر پیغام دیگری به نمایش درمی‌آید که نشان‌دهنده عملکرد متد Cls در تغییر موقعیت جاری مکان‌نماست.

```
Private Sub cmdcls_Click()
```

```
Dim pi, msg
```

```
pi = 3.14159265358979
```

```
Line (200, 150)-(850, 600), vbMagenta, BF
```

```
Circle (1600, 1800), 400, vbRed
```


```
msg = "Choose OK to clear this background"
```

```
MsgBox msg
```

```
Cls
```

```
MsgBox "CurrentX=" + Str(CurrentX) + "CurrentY=" + Str(CurrentY)
```

```
End Sub
```

 **مثال ۲:** پروژه‌ای طراحی کنید که به وسیله آن کاربر بتواند اشکال هندسی مانند نقطه، خط و انواع مستطیل را رسم کند. به این منظور مراحل بعد را انجام دهید:

۱- یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم با عرض و ارتفاع

۸۰۰۰ و ۶۰۰۰ باشد. سپس نام و عنوان آن را به ترتیب روی FrmDrawing و Drawing

تنظیم کنید و خصوصیت Appearance آن را روی Flat- تنظیم کنید. (شکل ۱۶-۱۲).

۲- در این مرحله بخش تعاریف فرم و رویدادهای ماوس را به صورت زیر تنظیم کنید:

```
Option Explicit
```

```
Dim intx As Integer, inty As Integer
```

```
Private Sub Form_MouseDown(Button As Integer, Shift As _
```

```
Integer, X As Single, Y As Single)
```

```
If Button = vbLeftButton Then
```

Pset (X,Y)

intx = X

inty = Y

End If

End Sub

Private Sub Form_MouseUp(Button As Integer,Shift As Integer , X As Single, Y As Single)

If Button = vbLeftButton Then

Select Case Shift

Case vbCtrlMask: Line (intx, inty)-(X,Y)

Case vbShiftMask: Line (intx, inty)-(X,Y), , B

Case vbAltMask: Line (intx, inty)-(X, Y), , BF

End Select

End If

End Sub

Private Sub Form_MouseMove(Button As Integer,Shift As_
Integer , X As Single, Y As Single)

If Button = vbRightButton Then

PSet (X,Y)

End If

End Sub



شکل ۱۶-۱۲

در رویداد MouseDown با استفاده از دستور If مقدار آرگومان Button بررسی می‌شود. اگر این مقدار برابر با vbLeftButton باشد به این معنی است که کلید سمت چپ ماوس فشرده شده است و با استفاده از دستور PSet یک نقطه در موقعیتی که اشاره‌گر ماوس قرار دارد، ترسیم می‌شود و همین مقادیر یعنی X و Y در دو متغیر عمومی ذخیره می‌شوند تا برای رویداد MouseUp مورد استفاده قرار گیرند. متد PSet می‌تواند یک نقطه در مختصات x و y با رنگ color ایجاد کند، color می‌تواند یکی از ثابت‌های رنگ ویژوال بیسیک باشد. شکل کلی استفاده از این دستور به صورت $PSet(x,y,color)$ می‌باشد. استفاده از آرگومان color اختیاری است.

از رویداد MouseUp برای ترسیم انواع خطوط و مستطیل‌های توپر و توخالی استفاده شده است. در این رویداد ابتدا رها شدن کلید سمت چپ ماوس به وسیله دستور If کنترل می‌شود، سپس با استفاده از یک دستور Select Case فشرده شدن کلیدهای Ctrl، Shift و Alt در زمان رها شدن کلید چپ ماوس بررسی می‌شود و اگر در زمان رها شدن کلید سمت چپ ماوس، کلید Ctrl نگه داشته شود، متد $(X,Y)-(Line(intx,inty))$ در اولین Case اجرا شده و یک خط بین نقطه با مختصات $(intx,inty)$ یعنی محل اشاره‌گر در زمان فشرده شدن کلید چپ ماوس و نقطه با مختصات (X,Y) یعنی محل اشاره‌گر در زمان رها شدن کلید چپ ماوس رسم می‌شود. اگر در زمان رها شدن کلید چپ ماوس، کلید Shift نگه داشته شود، متد $(X,Y,B)-(Line(intx,inty))$ در Case دوم اجرا شده و یک مستطیل بین نقاط $(intx,inty)$ و (X,Y) رسم می‌کند. پارامتر B در متد Line سبب رسم یک مستطیل

خواهد شد. اما اگر در زمان رها شدن کلید چپ ماوس، کلید Alt نگه داشته شود، متد $Line(x1,y1)-(x2,y2),color,BF$ در Case آخر اجرا شده و یک مستطیل توپر بین نقاط مربوطه رسم خواهد شد.

متد Line می تواند یک خط را با دریافت موقعیت نقاط ابتدا و انتهای آن ترسیم کند. شکل کلی این متد به صورت $Line(x1,y1)-(x2,y2),color$ است که $x2,x1$ فاصله نقاط ابتدا و انتهای خط از سمت چپ فرم و $y1$ و $y2$ فاصله نقاط ابتدا و انتهای خط از بالای فرم می باشد، آرگومان color رنگ خط ترسیمی را مشخص می کند. استفاده از این آرگومان اختیاری است.

برای رسم یک مستطیل توخالی از کاراکتر B در متد Line به صورت $Line(x1,y1)-(x2,y2),color,B$ استفاده کنید و برای رسم یک مستطیل توپر از عبارت BF در متد Line به صورت $Line(x1,y1)-(x2,y2),color,BF$ استفاده کنید.

آخرین رویداد در ماژول فرم، رویداد MouseMove فرم است که در این رویداد با بررسی آرگومان Button فشرده شدن دکمه راست ماوس بررسی می شود. اگر در زمان حرکت اشاره گر ماوس کلید راست ماوس نگه داشته شود (Right Drag) متد PSet اجرا شده و یک نقطه در محل اشاره گر ترسیم خواهد کرد.

۳ - پروژه را با نام drawing ذخیره کرده و سپس آن را اجرا کنید.

۴ - روی فرم، عمل کلیک انجام دهید. مشاهده می کنید که نقاط کوچکی با رنگ سیاه روی فرم ایجاد خواهد شد.

۵ - روی فرم عمل درگ انجام دهید و هم زمان کلید Ctrl را پایین نگه دارید، سپس کلید چپ ماوس را رها کنید. مشاهده می کنید که یک خط با رنگ سیاه ترسیم می شود. همین عمل را با کلیدهای Shift و Alt انجام دهید و نتیجه را بررسی کنید.

۶ - در این مرحله با کلید راست ماوس عمل درگ را انجام دهید، می بینید که با حرکت ماوس نقاط ترسیم می شوند.

۷ - از برنامه خارج شده و به پنجره ویژوال بیسیک بازگردید.

۷-۳-۱۲ متد Print

تاکنون بارها از این متد استفاده کرده‌اید. این متد می‌تواند هرگونه اطلاعات اعم از متن، مقادیر عددی، مقادیر مربوط به خصوصیات و نظایر آن‌ها را روی فرم نمایش دهد. شکل کلی این متد به صورتی است که در ادامه می‌آید:

```
object.Print outputlist
```

object نام شیئی است که اطلاعات روی آن نمایش داده می‌شوند. outputlist اطلاعاتی است که توسط متد Print روی شیء object نمایش داده می‌شود. استفاده از outputlist و object اختیاری است و در صورتی که outputlist استفاده نشود یک خط خالی نمایش داده خواهد شد.



نکته:

- استفاده از کاراکتر " ; " در متد Print سبب می‌شود تا اطلاعات به صورت چسبیده به هم و بدون فاصله از هم نمایش داده شوند.
- استفاده از کاراکتر " , " در متد Print سبب می‌شود تا اطلاعات با فاصله ۱۴ ستون از یکدیگر نمایش داده شوند.

به‌عنوان مثال، رویداد زیر عبارت Visual Basic 6 را به شکل‌های مختلف نمایش می‌دهند:

```
Private Sub cmdprint_Click( )
```

```
Dim str1 As String
```

```
Dim str2 As String
```

```
Dim str3 As String
```

```
str1 = "Visual"
```

```
str2 = "Basic"
```

```
str3 = "6"
```

```
Print str1; str2; str3
```

```
Print str1, str2, str3
```

```
Print str1;
```

```
Print str2,
```

```
Print str3
```

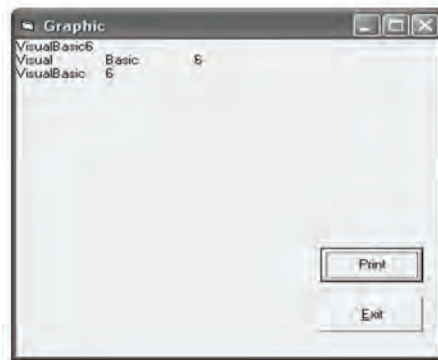
```
End Sub
```

نتیجه اجرای این دستورات به صورت زیر خواهد بود:

```
Visual Basic 6
```

```
Visual Basic 6
```

```
VisualBasic 6
```



شکل ۱۷-۱۲

برای نمایش فضاهای خالی توسط متد Print، می توانید از تابع Space استفاده کنید. مثلاً دستور زیر قبل از نمایش کلمه BASIC، ۵ فضای خالی ایجاد می کند.

```
Print Space (5); " BASIC "
```

در ضمن می توانید به وسیله تابع Tab اطلاعات خود را در ستون های مورد نظر نمایش دهید. شکل کلی تابع Tab به این صورت است:

```
Tab (n)
```

که در آن n یک مقدار عددی است که شماره ستون مورد نظر را معین می کند. مثلاً فرمان زیر، کلمه VISUAL را از ستون دوم به بعد و کلمه BASIC را از ستون دهم به بعد نمایش می دهد.

```
Print Tab (2) ; " VISUAL " ; Tab (10) ; " BASIC"
```

اما اگر این فرمان به صورت زیر استفاده شود:

```
Print Tab (2) ; " VISUAL " ; Tab (5) ; " BASIC "
```

کلمه VISUAL از ستون دوم خط جاری نمایش داده می‌شود اما چون پس از نمایش کلمه VISUAL، مکان نما در ستون هشتم قرار می‌گیرد و Tab دوم به ستون پنجم اشاره می‌کند بنابراین کلمه BASIC در ستون پنجم خط بعد نمایش داده خواهد شد.

 **نکته** در صورتی که مقدار n در تابع Tab عددی منفی باشد، ستون شماره ۱ در نظر گرفته خواهد شد.

۸-۳-۱۲ متدهای TextWidth و TextHeight

متد TextWidth عرض متن مورد نظر و متد TextHeight ارتفاع متن مورد نظر برای نمایش را معین می‌کنند. شکل کلی این دو متد به صورت زیر است:

```
object. TextWidth ( string )
```

```
object. TextHeight ( string )
```

در string یک عبارت رشته‌ای است. هر دو متد یک مقدار عددی از نوع Single را بازمی‌گردانند مثلاً برای نمایش عبارت VISUAL BASIC در وسط صفحه می‌توانید به این صورت عمل کنید:

```
Private Sub cmdtext_Click ( )
```

```
CurrentX = (ScaleWidth - TextWidth ("VISUAL BASIC")) / 2
```

```
CurrentY = (ScaleHeight - TextHeight ("VISUAL BASIC")) / 2
```

```
Print "VISUAL BASIC"
```

```
End Sub
```

نتیجه اجرای این رویه را در شکل ۱۸-۱۲ مشاهده کنید:

در این رویه به وسیله دو متد TextWidth و TextHeight و محاسبه عرض و ارتفاع متن و با کم کردن این مقادیر از عرض و ارتفاع فرم که به وسیله خصوصیات ScaleWidth و ScaleHeight به دست می‌آیند و تقسیم مقدار به دست آمده بر عدد ۲، محل نمایش متن را به گونه‌ای که در وسط صفحه قرار گیرد محاسبه شده‌اند. مزیت این کار این است که حتی اگر ابعاد فرم تغییر یابد و رویه مجدداً اجرا شود متن مورد نظر با توجه به ابعاد جدید باز

شده و در وسط صفحه قرار می‌گیرد.



شکل ۱۸-۱۲

۹-۳-۱۲ خصوصیت AutoRedraw

شاید تاکنون در هنگام استفاده از دستورات گرافیکی به این نکته دقت کرده‌اید که در بعضی از مواقع ترسیمات انجام شده در روی فرم از بین می‌رود یا در اثر قرار گرفتن پنجره‌های دیگر روی پنجره‌ای که ترسیمات در آن انجام شده است ترسیمات شما مخدوش می‌شود. به‌عنوان مثال یک فرم با یک دکمه فرمان به گونه‌ای طراحی کنید که با فشردن دکمه فرمان یک مستطیل توپر در روی فرم رسم شود سپس برنامه را اجرا کرده و روی دکمه فرمان کلیک کنید و بعد پنجره برنامه را به حداقل اندازه برسانید (به‌وسیله دکمه‌کنترلی Minimize) و مجدداً روی آیکن پنجره در نوار وظیفه کلیک کنید تا پنجره برنامه به حالت قبل بازگردد. در این صورت اثری از شکل رسم شده مشاهده نخواهد شد.

فرم‌ها و بعضی از کنترل‌ها مانند PictureBox خصوصیتی به نام AutoRedraw دارند که در حالت پیش‌فرض مقدار این خصوصیت False است اگر این خصوصیت روی مقدار True تنظیم شود در هنگام تغییر مکان یا تغییر اندازه فرم و نظایر آن، ترسیمات موجود از بین نمی‌روند و با نمایش مجدد فرم یا کنترل نمایش داده می‌شوند.

در مثالی که در این قسمت انجام دادید خصوصیت AutoRedraw را برای فرم برنامه روی مقدار True تنظیم کنید و برنامه را مجدداً اجرا کنید و روی دکمه فرمان کلیک کنید. فرم را به حالت Minimize درآورده و دوباره آن را روی دسکتاپ نمایش دهید. مشاهده می‌کنید در این حالت مشکل قبلی از بین رفته است.

شکل کلی استفاده از این خصوصیت به این صورت است:

`object.AutoRedraw = Boolean`

object یک مقدار اختیاری است که می‌تواند نام فرم یا کنترل باشد و در صورت عدم استفاده از آن، نام فرمی که فوکوس دارد استفاده می‌شود. مقدار Boolean یک مقدار منطقی است که می‌تواند True یا False باشد. در صورت عدم استفاده از مقدار Boolean، مقدار خصوصیت بازگشت داده می‌شود. مقدار پیش‌فرض این خصوصیت False است.



تمرین:

از خصوصیت AutoRedraw در پروژه‌هایی که تاکنون طراحی کرده‌اید استفاده کرده و نتیجه را با حالت قبل از استفاده این خصوصیت بررسی کنید.

۱۰-۳-۱۲ خصوصیت DrawMode

به وسیله این خصوصیت می‌توان شکل ظاهری ترسیماتی (از نظر رنگ) را که به وسیله متدهای گرافیکی نظیر Circle، Line، PSet و غیره انجام می‌شوند تعیین کرد. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

`object.DrawMode = value`

مقدار value نیز اختیاری بوده و شکل ظاهری ترسیمات را مشخص می‌کند. در صورت عدم استفاده از این مقدار، مقدار خصوصیت بازگشت داده می‌شود. مقادیری که بخش value می‌تواند کسب کند در جدول ۳-۱۲ آورده شده‌اند.

توجه داشته باشید که در جدول ۳-۱۲ منظور از رنگ pen، رنگی است که متد گرافیکی برای رسم شکل گرافیکی از آن استفاده می‌کند. در واقع این خصوصیت به ویژوال بیسیک می‌گوید که چگونه رنگ یک pixel نمایشی روی صفحه را تعیین کند به عبارت دیگر ویژوال بیسیک بر اساس رنگ فعلی pixel و رنگ نقطه‌ای که در این pixel به وسیله متد گرافیکی ایجاد می‌شود، ترکیب رنگی مناسب را با توجه به مقدار خصوصیت ScaleMode تعیین می‌کند. به عنوان مثال دستورات زیر نحوه استفاده از این خصوصیت را نشان می‌دهند:

Form1.DrawMode = vbInvert

Form1.BackColor = vbRed

Line (500 , 500) - (2000 , 2500) vbGreen, BF

در حالت عادی این دستورات یک مستطیل توپر به وسیله متد Line با رنگ سبز رسم می کنند اما چون مقدار خصوصیت DrawMode روی vbInvert تنظیم شده، از رنگ معکوس رنگ قرمز (یعنی رنگ فعلی نقاطی که مستطیل به وسیله آن‌ها نمایش داده می شود) استفاده می شود و مستطیل با این رنگ دیده خواهد شد. بنابراین اگر رنگ دیگری نیز در دستور Line استفاده شود تغییری در رنگ مستطیل ایجاد نمی شود.

جدول ۳-۱۲ مقادیر مربوط به خصوصیت DrawMode

| ثابت رشته‌ای | ثابت عددی | توضیح |
|----------------|-----------|---|
| vbBlackness | ۱ | رنگ سیاه |
| vbNotMergePen | ۲ | عکس حالت 15-vbMergePen |
| vbMaskNotPen | ۳ | ترکیب رنگ بر اساس رنگ زمینه و رنگ معکوس pen |
| vbNotCopyPen | ۴ | عکس حالت 13-CopyPen |
| vbMaskPenNot | ۵ | ترکیب رنگ‌های عمومی با رنگ pen و رنگ معکوس، رنگی که نمایش داده شده است. |
| vbInvert | ۶ | رنگ معکوس، رنگی که نمایش داده شده است. |
| vbXorPen | ۷ | ترکیب رنگ‌ها بر اساس رنگ pen یا رنگی که نمایش داده شده است. |
| vbNotMaskPen | ۸ | عکس حالت 9-MaskPen |
| vbMaskPen | ۹ | ترکیب رنگ‌های عمومی با رنگ pen و رنگی که نمایش داده شد. |
| vbNotXorPen | ۱۰ | عکس حالت 7-vbXorPen |
| vbNop | ۱۱ | ترسیمی انجام نمی‌شود. |
| vbMergeNotePen | ۱۲ | ترکیب رنگ معکوس، رنگ pen و رنگی که نمایش داده شده است. |
| vbCopyPen | ۱۳ | به‌طور پیش‌فرض رنگ pen و اگر رنگ pen تعیین نشود رنگ ForeColor استفاده می‌شود. |
| vbMergePenNot | ۱۴ | ترکیب رنگ pen و رنگ معکوس، رنگی که نمایش داده شده است. |
| vbMergePen | ۱۵ | ترکیب رنگ pen و رنگی که نمایش داده شده است. |
| vbWhiteness | ۱۶ | رنگ سفید |



نکته مقدارپیش فرض خصوصیت DrawMode، 13-vbCopyPen است که سبب می‌شود از رنگی که در متد گرافیکی تعیین می‌شود استفاده شود و در صورتی که رنگ توسط متد تعیین نشود از رنگی که در خصوصیت ForeColor تعیین شده استفاده شود.

۱۱-۳-۱۲ خصوصیت DrawStyle

به وسیله این خصوصیت می‌توان نوع و حالت خطوط را در ترسیمات گرافیکی تعیین کرد. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

object. DrawStyle = value

مقدار value اختیاری بوده و نوع و حالت ترسیمات را مشخص می‌کند. در صورت عدم استفاده از این مقدار، مقدار خصوصیت بازگشت داده می‌شود. مقادیری که value می‌تواند کسب کند در جدول ۴-۱۲ ارایه شده است.

جدول ۴-۱۲ مقادیر مربوط به خصوصیت DrawStyle

| ثابت رشته‌ای | ثابت عددی | حالت نمایشی |
|---------------|-----------|---|
| vbSolid | ۰ | _____ |
| vbDash | ۱ | — — — — |
| vbDot | ۲ | · · · · · |
| vbDashDot | ۳ | — · — · — |
| vbDashDotDot | ۴ | — · · — · · — |
| vbInvisible | ۵ | خط نامرئی |
| vbInsideSolid | ۶ | لبه بیرونی حاشیه بر لبه بیرونی شکل منطبق است. |

رویداد زیر، حالت‌های مختلف ترسیم را برای مقادیر صفر تا ۴ مورد استفاده قرار می‌دهد.

```
Private Sub cmdshow_Click( )
```

```
    DrawStyle = vbSolid
```

```
    Line (500, 200)-(5000, 200), vbBlue
```

```
    DrawStyle = vbDash
```

```
    Circle (900,1000), 400, vbRed
```

```
DrawStyle = vbDot
```

```
Line (2000, 500)-(5000, 1500), vbBlue, B
```

```
DrawStyle = vbDashDot
```

```
Circle (900, 2500),400, vbRed
```

```
DrawStyle = vbDashDotDot
```

```
Line (2000 ,2000)-( 5000-3000), vbBlue, B
```

```
End Sub
```



شکل ۱۹-۱۲

۱۲-۳-۱۲ خصوصیت DrawWidth

به وسیله این خصوصیت می‌توانید ضخامت خطوط را برای ترسیمات گرافیکی معین کنید. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

object.DrawWidth = size

مقدار size اختیاری بوده و یک عبارت عددی است که اندازه قلم را برای ترسیمات معین می‌کند و در صورت عدم استفاده از این مقدار، مقدار خصوصیت بازگشت داده می‌شود. مقدار size می‌تواند مقداری بین ۱ تا ۳۲۷۶۷ باشد.

در این رویداد حالت‌های مختلف ترسیم را با اندازه‌های مختلف قلم مشاهده می‌کنید (شکل ۲۰-۱۲).

```
Private Sub cmdshow_Click( )
```

```
DrawStyle = vbDot
```

```
DrawWidth = 1
```

```
Line (500, 150)-(5000, 150), vbBlue
```

```
DrawWidth = 2
```


Circle (900, 1000), 400, vbRed

DrawWidth = 3

Line (2000, 500)-(5000, 1500), vbBlue, B

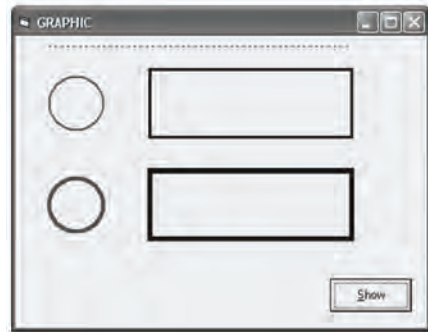
DrawWidth = 4

Circle (900, 2500), 400, vbRed

DrawWidth = 5

Line (2000, 2000)-(5000, 3000), vbBlue, B

End Sub



شکل ۲۰-۱۲

نکته در صورتی که خصوصیت DrawWidth روی عدد بزرگ‌تر از یک تنظیم شود مقادیر ۱ تا ۴ برای خصوصیت DrawStyle در زمان اجرای متدهای گرافیکی عملکردی از خود نشان نمی‌دهند و مانند مقدار vbSolid عمل می‌کنند.

۱۳-۳-۱۲ خصوصیت FillStyle

به وسیله این خصوصیت می‌توانید ترسیماتی نظیر دایره، بیضی و مستطیل را با حالت‌های مختلف پر کنید. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

object.FillStyle = number

مقدار number اختیاری بوده و عددی صحیح است که حالت موردنظر را برای پر کردن ترسیمات معین می‌کند. در صورت عدم استفاده از این بخش، مقدار خصوصیت بازگشت داده می‌شود. مقادیر مجاز برای number در جدول ۵-۱۲ ارایه شده است:

جدول ۵-۱۲ مقادیر مربوط به خصوصیت FillStyle

| ثابت رشته‌ای | ثابت عددی | توضیح |
|--------------------------|-----------|--|
| vbFsWithSolid | ۰ | داخل شکل با رنگی که در خصوصیت FillColor تعیین شده پر می‌شود. |
| vbFsWithTransparent | ۱ | داخل شکل با رنگ زمینه پر می‌شود. |
| vbFsWithHorizontalLine | ۲ | داخل شکل با خطوط افقی پر می‌شود. |
| vbFsWithVerticalLine | ۳ | داخل شکل با خطوط عمودی پر می‌شود. |
| vbFsWithUpwardDiagonal | ۴ | داخل شکل با خطوط مایل پر می‌شود (از چپ به راست). |
| vbFsWithDownwardDiagonal | ۵ | داخل شکل با خطوط مایل پر می‌شود (از راست به چپ). |
| vbFsWithCross | ۶ | داخل شکل با خطوط عمودی و افقی پر می‌شود (حالت شطرنجی). |
| vbFsWithDiagonalCross | ۷ | داخل شکل با خطوط عمودی و افقی مایل پر می‌شود (حالت شطرنجی مایل). |

در این رویداد حالت‌های مختلف ترسیم را برای مقادیر متفاوتی از خصوصیت FillStyle نشان می‌دهد:

```
Private Sub cmdshow_Click()
    DrawStyle = vbSolid
    DrawWidth = 1
    FillColor = vbGreen
    FillStyle = 0
    Circle (900, 1000), 400, vbRed
    FillColor = vbBlack
    FillStyle = 1
    Line (2000, 500)-(5000, 1500), vbBlue, B
    FillStyle = 2
    Circle (900, 2200), 400, vbRed
    FillStyle = 3
    Line (2000, 1700)-(5000, 2700), vbBlue, B
    FillStyle = 4
    Circle (900, 3500), 400, vbRed
    FillStyle = 5
```

Line (2000, 3000)-(5000, 4000), vbBlue, B

FillStyle = 6

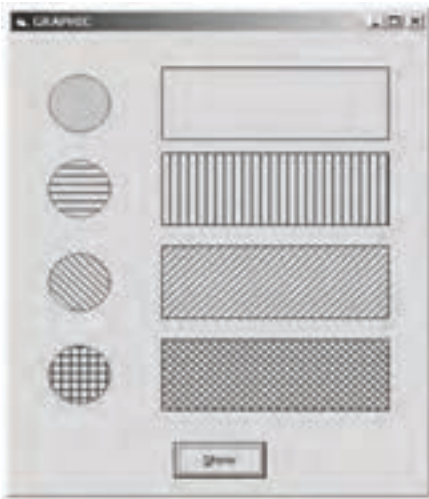
Circle (900, (4800, 400, vbRed

FillStyle = 7

Line (2000, 4300)-(5000, 5300), vbBlue, B

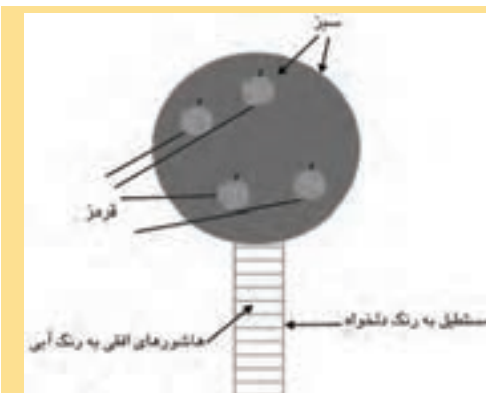
End Sub

در این رویداد پس از تعیین مقادیر مورد نظر برای خصوصیات DrawStyle و DrawWidth، مقدار خصوصیت FillColor روی سبز تنظیم شده است. خصوصیت FillColor رنگ قلم را برای ترسیماتی که به وسیله خصوصیت FillStyle ایجاد می‌شود معین می‌کند.



شکل ۲۱-۱۲

به عنوان مثال در دایره اول چون مقدار خصوصیت FillStyle صفر است دایره با رنگ FillColor یعنی سبز پر می‌شود. در مورد سایر مقادیر FillStyle نیز خصوصیت FillColor رنگ خطوط عمودی، افقی، مایل و غیره که شکل را پر می‌کند تعیین می‌کند. نتیجه اجرای این رویداد را می‌توانید در شکل ۲۱-۱۲ مشاهده کنید.



شکل ۲۲-۱۲

تمرین: 

پروژه‌ای طراحی کنید که این شکل را روی یک فرم نمایش دهد:

۴-۱۲ تابع QBcolor

این تابع با دریافت یک عدد بین صفر و ۱۵، یک عدد از نوع Long را که بیانگر رنگ معادل عدد دریافتی است بازمی‌گرداند. شکل کلی این تابع به صورت زیر است:

QBColor (color)

آرگومان color یک عدد از نوع صحیح است که براساس جدول ۶-۱۲ قابل استفاده خواهند بود.

جدول ۶-۱۲ مقادیر قابل استفاده برای آرگومان color

| مقدار عددی | رنگ | مقدار عددی | رنگ |
|------------|-----------|------------|----------------|
| ۰ | سیاه | ۸ | خاکستری |
| ۱ | آبی | ۹ | آبی روشن |
| ۲ | سبز | ۱۰ | سبز روشن |
| ۳ | فیروزه‌ای | ۱۱ | فیروزه‌ای روشن |
| ۴ | قرمز | ۱۲ | قرمز روشن |
| ۵ | بنفش | ۱۳ | بنفش روشن |
| ۶ | زرد | ۱۴ | زرد روشن |
| ۷ | سفید | ۱۵ | سفید روشن |

به عنوان مثال در دستورات زیر با استفاده از تابع QBColor، ۵۰۰۰ نقطه با رنگ‌های متفاوت نمایش داده خواهد شد.

```
Dim i As Integer
```

```
Dim XPos As Single, YPos As Single
```

```
DrawWidth = 4
```

```
Randomize
```

```
For i = 1 To 5000
```

```
    XPos = Rnd * ScaleWidth
```

```
    YPos = Rnd * ScaleHeight
```

```
    PSet (XPos, YPos), QBColor(Rnd * 15)
```

```
Next i
```

۵-۱۲ تابع RGB

در ویژوال بیسیک تابع دیگری به نام RGB وجود دارد که می‌تواند ترکیبات رنگی را با توجه به نیاز ایجاد کند. این تابع می‌تواند با دریافت سه مقدار عددی برای سه رنگ اصلی تمام ترکیبات مورد نظر را ایجاد کند. شکل کلی این تابع به صورت زیر است:

RGB (red , green , blue)

این تابع سه آرگومان اجباری دارد که می‌توانند اعداد صحیح از صفر تا ۲۵۵ را کسب کنند. آرگومان red مقدار رنگ قرمز، آرگومان green مقدار رنگ سبز و آرگومان blue مقدار رنگ آبی را معین می‌کنند. مقدار بازگشتی این تابع یک عدد از نوع Long است که بیانگر ترکیب رنگی درخواستی است. مقادیر سه آرگومان فوق برای رنگ‌های استاندارد در جدول ۷-۱۲ ارائه شده است.

جدول ۷-۱۲ مقادیر سه رنگ اصلی برای رنگ‌های استاندارد

| رنگ | مقدار آرگومان red | مقدار آرگومان green | مقدار آرگومان blue |
|-----------|-------------------|---------------------|--------------------|
| سیاه | ۰ | ۰ | ۰ |
| آبی | ۰ | ۰ | ۲۵۵ |
| سبز | ۰ | ۲۵۵ | ۰ |
| فیروزه‌ای | ۰ | ۲۵۵ | ۲۵۵ |
| قرمز | ۲۵۵ | ۰ | ۰ |
| بنفش | ۲۵۵ | ۰ | ۲۵۵ |
| زرد | ۲۵۵ | ۲۵۵ | ۰ |
| سفید | ۲۵۵ | ۲۵۵ | ۲۵۵ |

به عنوان مثال دستورات زیر پانصد دایره با ابعاد و مختصات و رنگ‌های تصادفی ایجاد می‌کند.

```
Dim i As Integer
```

```
Dim XPos As Single, YPos As Single
```

```
DrawWidth = 4
```

Randomize

For i = 1 To 500

XPos = Rnd * ScaleWidth

YPos = Rnd * ScaleHeight

(Circle (XPos, YPos), Rnd * 800, RGB(Rnd * _255, Rnd * 255, Rnd * 255

Next i

۶-۱۲ شی چاپگر (Printer Object)

تاکنون کلیه عملیاتی که انجام داده‌اید روی فرم و صفحه نمایش انجام شده است اما گاهی اوقات لازم است تا اطلاعات مورد نیاز خود را به وسیله چاپگر روی کاغذ چاپ کنید. ویژوال بیسیک در این زمینه نیز امکانات لازم را مهیا کرده است. شما با استفاده از شی چاپگر، علاوه بر انجام عملیات چاپ می‌توانید به وسیله خصوصیات این شی عملیات چاپ را به نحو مناسبی مدیریت کنید. در این بخش به معرفی متدهای چاپ و معرفی خصوصیات شی چاپگر می‌پردازیم.

۱-۶-۱۲ متدهای چاپ

تاکنون متدهای مختلفی را برای نمایش اطلاعات و ترسیمات آموختید. در این متدها می‌توانید به جای پارامتر object از شی چاپگر استفاده کنید؛ بنابراین به آسانی می‌توانید از متدهای Circle، Line، PSet، Scale، TextHeight، TextWidth و Print استفاده کنید. فقط کافی است برای معرفی شی چاپگر از کلمه Printer استفاده کنید. به‌عنوان مثال نتایج حاصل از اجرای دستورات زیر روی کاغذ چاپ خواهند شد.

```
Printer.Print " IN THE NAME OF GOD. "
```

```
(Printer.Line (100,100) - (800,800
```

```
Printer.Circle (150,150), 500
```

متد EndDoc

این متد سبب می‌شود تا عملیات چاپ متوقف شده و تا زمانی که چاپگر آماده چاپ شود اطلاعات مربوط به چاپ، روی دیسک یا حافظه کامپیوتر ذخیره می‌شود. شکل کلی

نحوه استفاده از این متد به صورت زیر است:

Printer.EndDoc

متد KillDoc

این متد می‌تواند در زمان چاپ اطلاعات، عملیات چاپ را خاتمه دهد. شکل کلی نحوه استفاده از این متد به صورت زیر است:

Printer.KillDoc

متد NewPage

متد Newpage می‌تواند عملیات چاپ صفحه جاری را خاتمه داده و چاپگر، چاپ را از صفحه بعدی انجام دهد. شکل کلی نحوه استفاده از این متد در ادامه می‌آید:

Printer.NewPage

۲-۶-۱۲ خصوصیات شیء چاپگر

تنظیمات مربوط به چاپگرها نیز مانند اشیای دیگر به وسیله تعدادی از خصوصیات قابل دستیابی و تغییر هستند. در این بخش مهم‌ترین خصوصیات شیء چاپگر را مورد بررسی قرار می‌دهیم. بعضی از خصوصیات نیز قبلاً توضیح داده شده‌اند مانند: DrawMode, Style, CurrentY, CurrentX, FillStyle, FillColor, DrawWidth, خصوصیات مربوط به قلم‌ها (Fonts) و ...

خصوصیت ColorMode

به وسیله این خصوصیت می‌توان نوع چاپگر را از نظر چاپ رنگی یا سیاه سفید تعیین کرد. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

Printer.ColorMode = value

value یک ثابت عددی یا رشته‌ای است که نوع چاپ را معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۸-۱۲ باشد. در صورت عدم استفاده از بخش value، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

جدول ۸-۱۲ مقادیر مربوط به خصوصیت ColorMode

| توضیح | ثابت عددی | ثابت رشته‌ای |
|---------------|-----------|------------------|
| چاپ سیاه سفید | ۱ | vbPRCMMonochrome |
| چاپ رنگی | ۲ | vbPRCMColor |



نکته استفاده از چاپ رنگی یا سیاه سفید به امکانات چاپگر بستگی دارد.

خصوصیت Copies

به وسیله این خصوصیت می‌توانید تعداد نسخه‌هایی که چاپگر چاپ می‌گیرد تعیین کنید. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

Printer.Copies = number

Number یک عبارت عددی از نوع صحیح است که تعداد نسخه‌ها را برای چاپ معین می‌کند و در صورت عدم استفاده از آن، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

خصوصیت DeviceName

این خصوصیت نام دستگاه چاپگر پیش فرض را باز می‌گرداند. نام چاپگرها در زمان نصب آن‌ها از طریق برنامه Control Panel توسط کاربر تعیین می‌شود. مثلاً اگر یک چاپگر EPSON LQ 300 را با نام myprinter و به صورت پیش فرض نصب کرده باشید فرمان زیر نام چاپگر یعنی myprinter را نمایش می‌دهد.

Print Printer.DeviceName

خصوصیت DriverName

این خصوصیت نام راه‌انداز (driver) دستگاه چاپگر پیش فرض را باز می‌گرداند. به عنوان مثال اگر چاپگر EPSON LQ 500 را با نام myprinter نصب کرده باشید فرمان زیر نام راه‌انداز نصب شده یعنی EPSON LQ 500 را نمایش خواهد داد.

Print Printer.DriverName

خصوصیت Orientation

به وسیله این خصوصیت می‌توانید جهت انجام عملیات چاپ را روی صفحه کاغذ تعیین کنید. عملیات چاپ می‌تواند به صورت portrait یا landscape باشد. در شکل ۲۳-۱۲ تفاوت این دو حالت نمایش داده شده است.



شکل ۲۳-۱۲

شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

Printer.Orientation = value

value یک ثابت عددی یا رشته‌ای است که جهت چاپ اطلاعات را روی کاغذ معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۹-۱۲ باشد. در صورت عدم استفاده از مقدار value، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

جدول ۹-۱۲ مقادیر مربوط به خصوصیت Orientation

| ثابت رشته‌ای | ثابت عددی | توضیح |
|-----------------|-----------|-----------------------|
| vbPRORPortrait | ۱ | چاپ به صورت portrait |
| vbPRORLandscape | ۲ | چاپ به صورت landscape |

خصوصیت Page

این خصوصیت شماره صفحه در حال چاپ را در اختیار برنامه قرار می‌دهد. شکل کلی نحوه استفاده از این خصوصیت به این صورت است:

Printer.Page

خصوصیت PaperSize

به وسیله این خصوصیت می‌توانید نوع و ابعاد کاغذ چاپ را تنظیم کنید. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

Printer.PaperSize = value

value یک ثابت عددی یا رشته‌ای است که ابعاد کاغذ را معین می‌کند. این مقدار می‌تواند یکی از مقادیر موجود در جدول ۱۰-۱۲ باشد. در صورت عدم استفاده از مقدار value، مقدار فعلی خصوصیت بازگشت داده خواهد شد.

جدول ۱۰-۱۲ مقادیر مربوط به اندازه کاغذ در خصوصیت PageSize

| ثابت رشته‌ای | ثابت عددی | ابعاد کاغذ |
|-------------------|-----------|--|
| vbPRPSLetter | ۱ | $8 \frac{1}{2} \times 11$ Inch |
| vbPRPSLetterSmall | ۲ | $8 \frac{1}{4} \times 11$ Inch (اندازه کوچک) |
| vbPRPSTabloid | ۳ | 11×17 Inch |
| vbPRPSLedger | ۴ | 17×11 Inch |
| vbPRPSLegal | ۵ | $8 \frac{1}{2} \times 14$ Inch |
| vbPRPSStatement | ۶ | $5 \frac{1}{2} \times 8 \frac{1}{2}$ Inch |
| vbPRPSExecutive | ۷ | $7 \frac{1}{2} \times 11$ Inch |
| vbPRPSA3 | ۸ | A3 (۲۹۷× mm) |
| vbPRPSA4 | ۹ | A4 (۲۹۷× mm) |
| vbPRPSA۴Small | ۱۰ | A4 (اندازه کوچک) |
| vbPRPSA5 | ۱۱ | A5 (۱۴۸×mm) |

خصوصیت Port

به وسیله این خصوصیت می‌توان نام پورت مربوط به چاپگری را که چاپ به آن ارسال می‌شود به دست آورد. شکل کلی نحوه استفاده از این خصوصیت به این صورت است:

Printer.Port

چاپگرها معمولاً از پورت‌های موازی LPT1 و LPT2 استفاده می‌کنند. 

خصوصیت PrintQuality

این خصوصیت کیفیت وضوح چاپ را در چاپگر معین می‌کند. شکل کلی نحوه استفاده از این خصوصیت به صورت زیر است:

Printer.PrintQuality = value

value یک ثابت عددی یا رشته‌ای است که میزان وضوح چاپ را معین می‌کند در صورت عدم استفاده از مقدار value مقدار فعلی خصوصیت بازگشت داده خواهد شد.

مقدار value می‌تواند یکی از مقادیر موجود در جدول ۱۱-۱۲ باشد.

جدول ۱۱-۱۲ مقادیر مربوط به کیفیت چاپ در خصوصیت PrintQuality

| ثابت رشته‌ای | ثابت عددی | توضیح |
|--------------|-----------|--------------------|
| vbPRPQDraft | -۱ | چاپ با کیفیت Draft |
| vbPRPQLow | -۲ | چاپ با کیفیت پایین |
| vbPRPQMedium | -۳ | چاپ با کیفیت متوسط |
| vbPRPQHigh | -۴ | چاپ با کیفیت بالا |

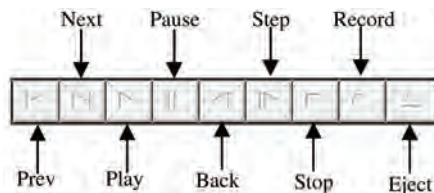
۱۲-۷ چند رسانه‌ای (Multimedia)

یکی از ویژگی‌های دیگر زبان برنامه‌نویسی ویژوال بیسیک توانایی استفاده از امکانات صوتی و تصویری در آن است. با استفاده از کنترل‌های ارائه شده در زبان برنامه‌نویسی ویژوال بیسیک می‌توان انواع فایل‌های ویدئویی، صوتی و موسیقی را مورد استفاده قرار داد.

۱۲-۷-۱ کنترل MCI (Media Control Interface)

با استفاده از این کنترل می‌توان انواع فایل‌های صدا، موسیقی، ویدئویی و سی‌دی‌های صوتی را پخش کرد.

این کنترل به‌طور پیش‌فرض در جعبه ابزار مشاهده نمی‌شود. برای اضافه کردن این کنترل گزینه Components را از منوی Project انتخاب کنید سپس در کادر محاوره Components که نمایش داده می‌شود کادر علامت Microsoft Multimedia Control 6.0 را انتخاب کنید و روی دکمه OK کلیک کنید. در این مرحله کنترل MCI به کنترل جعبه ابزار اضافه می‌شود. کنترل MCI شامل ۹ دکمه است که وظایف مختلفی را به عهده دارند. عملکرد این دکمه‌ها در شکل ۱۲-۲۴ و مطابق جدول ۱۲-۱۴ قابل مشاهده است.



شکل ۱۲-۲۴



مثال ۳: می‌خواهیم پروژه‌ای طراحی کنیم که به وسیله آن بتوان فایل‌های ویدئویی

از نوع avi را مشاهده کرد برای این کار عملیات بعد را به ترتیب انجام دهید:

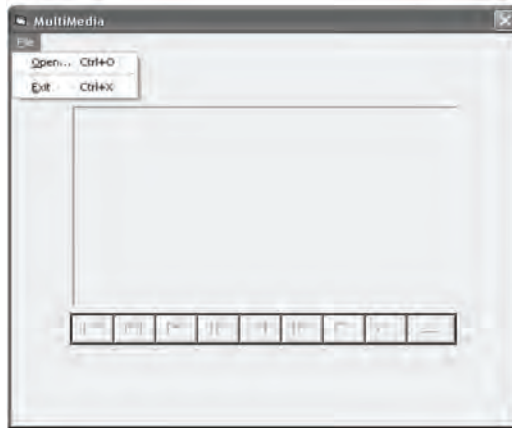
۱- یک پروژه از نوع StandardEXE ایجاد کنید سپس خصوصیت‌های فرم را مطابق جدول

۱۲-۱۲ تنظیم کنید.

جدول ۱۲-۱۲ خصوصیات فرم

| مقدار | خصوصیت |
|---------------|---------|
| frmmultimedia | Name |
| Multimedia | Caption |

۲- یک کنترل منو و کادر تصویر با نام PicVideo مطابق شکل ۱۲-۲۵ روی فرم قرار دهید.



شکل ۱۲-۲۵

۳- کنترل MCI را به جعبه ابزار اضافه کنید. در جعبه ابزار روی آیکن MMControl

دابل کلیک کنید تا کنترل MCI روی فرم قرار گیرد، سپس خصوصیت Name آن را



روی مقدار mcivideo و اندازه و موقعیت آن را مطابق شکل ۱۲-۲۵ تنظیم نمایید.

۴- به ماژول فرم بروید و دستورات زیر را در رویداد Load فرم تایپ کنید.

```
mcivideo.DeviceType = "aviVideo"
```

```
mcivideo.hWndDisplay = picvideo.hWnd
```

دستور اول با استفاده از خصوصیت DeviceType کنترل MCI را برای نمایش فایل‌ها با قالب‌بندی از نوع AVI آماده می‌کند. با استفاده از این خصوصیت می‌توان انواع رسانه‌های مالتی‌مدیا را برای استفاده تنظیم کرد. این خصوصیت می‌تواند مقادیر ارائه‌شده در جدول ۱۲-۱۳ را کسب کند.

جدول ۱۲-۱۳

| مقدار | توضیح |
|--------------|------------------------------|
| AVIVideo | پخش فایل‌های صوتی تصویری AVI |
| CDAudio | پخش سی‌دی صوتی |
| DigitalVideo | پخش فایل‌های ویدئویی دیجیتال |
| Videodisc | پخش دیسک‌های ویدئویی |
| WaveAudio | پخش فایل‌های صوتی از نوع WAV |

دستور دوم در این رویداد نیز سبب می‌شود تصویر فایل ویدئویی در کنترل جعبه تصویر Picvideo نمایش داده شود.

۵ - رویداد کلیک گزینه mnuopen را به صورت زیر تنظیم کنید:

```
Private Sub mnuopen_Click()
```

```
Dim strpath As String
```

```
strpath = InputBox("Enter Path and Filename :", "Input Data")
```

```
mcivideo.FileName = strpath
```

```
mcivideo.Command = "Open"
```

```
End Sub
```

در این رویداد ابتدا با استفاده از یک کادر ورود داده، مسیر و نام فایل ویدئویی دریافت شده و در متغیر strpath ذخیره می‌شود سپس محتویات strpath در خصوصیت Filename کنترل MCI قرار می‌گیرد تا کنترل MCI بتواند به فایل ویدئویی دسترسی پیدا کند سپس خصوصیت Command کنترل MCI روی مقدار Open تنظیم می‌شود تا کنترل آماده پخش فایل ویدئویی شود. این خصوصیت می‌تواند فرمان‌های مختلفی را برای اجرا در کنترل مشخص کند. این فرمان‌ها در جدول ۱۲-۱۴ قابل مشاهده است.

جدول ۱۴-۱۲

| مقدار | توضیح |
|--------|---|
| Open | ابزار مورد نیاز برای کنترل MCI را باز می کند. |
| Close | ابزار مورد نیاز برای کنترل MCI را می بندد. |
| Play | فایل ویدئویی، صدا یا موسیقی را پخش می کند. |
| Pause | عملیات پخش یا ضبط را موقتاً قطع می کند. |
| Stop | عملیات ضبط یا پخش را قطع می کند. |
| Back | از موقعیت جاری کمی به عقب حرکت می کند. |
| Step | از موقعیت جاری کمی به جلو حرکت می کند. |
| Prev | به ابتدای شیار (Track) جاری حرکت می کند. اگر ۳ ثانیه از شیار جاری پخش شده باشد به ابتدای شیار قبلی باز می گردد. |
| Next | یک ترک به جلو حرکت می کند. |
| Record | عملیات ضبط را آغاز می کند. |
| Eject | سی دی را به درایو وارد یا از آن خارج می کند. |

۶ - پروژه و فرم را با نام playvideo ذخیره کرده سپس برنامه را اجرا کنید.

۷ - روی گزینه Open از منوی File کلیک کرده، نام و مسیر فایل ویدئویی خود را که دارای پسوند avi است در کادر ورود داده تایپ کنید و روی دکمه OK کلیک نمایید تا به فرم برنامه بازگردید.

۸ - در این مرحله بعضی از دکمه های کنترل MCI فعال می شوند روی دکمه پخش (Play) کلیک کنید تا فایل ویدئویی مورد نظرتان در کنترل جعبه تصویر نمایش داده شود سپس عملکرد سایر دکمه های کنترل MCI را بررسی نمایید.

۹ - از برنامه خارج شده و به پنجره ویژوال بیسیک بازگردید.


تمرین:

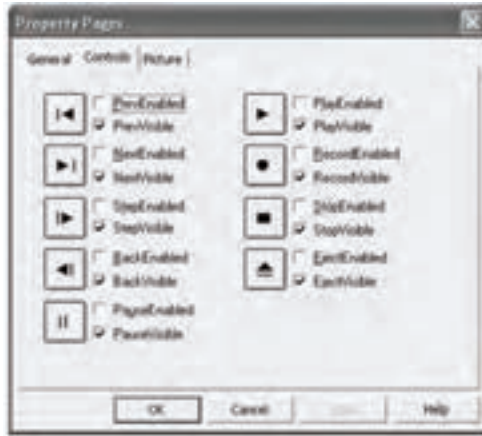


پروژه playvideo را به گونه ای تغییر دهید که به وسیله آن بتوان فایل های صوتی با پسوند WAV را پخش کرد.

نکته در کنترل MCI با استفاده از خصوصیت Orientation می توان کنترل را به صورت افقی (mciOrientHorz) یا عمودی (mciOrientVert) تنظیم نمود.



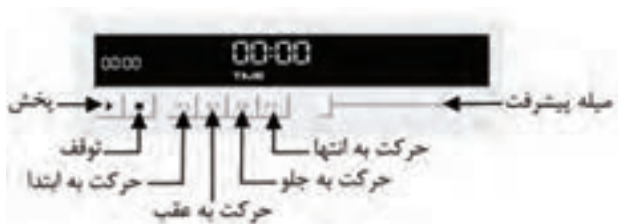
نکته  کلیه دکمه‌های موجود در کنترل MCI را می‌توان با استفاده از صفحه‌های خصوصیت (Property Pages) مرئی یا مخفی، فعال یا غیرفعال نمود. به منظور دسترسی به صفحه‌های خصوصیت در پنجره خصوصیات روی دکمه **...** روبه‌روی خصوصیت Custom کلیک کنید سپس زبانه Controls را انتخاب کنید (شکل ۱۲-۲۶).



شکل ۱۲-۲۶

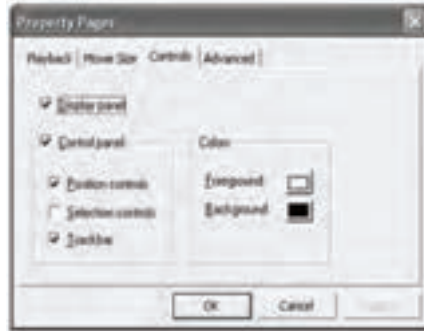
۱۲-۷-۲ کنترل Microsoft Active Movie (MAM)

با استفاده از این کنترل می‌توان انواع فایل‌های صوتی و موسیقی مانند WAV و MIDI را پخش کرد و فایل‌های تصویری مانند AVI، MPEG و MOV را نمایش داد. این کنترل به طور عادی در جعبه ابزار مشاهده نمی‌شود. برای اضافه کردن این کنترل گزینه Components را از منوی Project انتخاب کنید سپس در کادر محاوره Components که نمایش داده می‌شود کادر علامت Microsoft Active Movie Control را انتخاب کنید و روی دکمه OK کلیک کنید تا کنترل MAM به جعبه ابزار اضافه شود. این کنترل نیز مانند کنترل MCI دکمه‌های متعددی را شامل می‌شود که وظایف متفاوتی را بر عهده دارند (شکل ۱۲-۲۷).



شکل ۱۲-۲۷

در این کنترل به طور پیش فرض فقط دکمه‌های پخش (Play) و توقف (Stop) قابل استفاده می‌باشد. به منظور دسترسی به سایر دکمه‌ها می‌توانید پس از انتخاب کنترل، در پنجره خصوصیات روی دکمه **...** روبه‌روی خصوصیت Custom کلیک کنید. در کادر محاوره Property Pages روی زبانه Controls کلیک کنید (شکل ۲۸-۱۲). با استفاده از کادرهای علامت موجود در این بخش می‌توان دکمه‌ها و سایر بخش‌های کنترل را مخفی یا قابل مشاهده کرد.




شکل ۲۸-۱۲

مثال ۴: می‌خواهیم پروژه‌ای طراحی کنیم که به وسیله آن بتوان فایل‌های صوتی، موسیقی و ویدئویی را پخش کرد. برای این کار عملیات زیر را به ترتیب انجام دهید:

۱ - یک پروژه از نوع Standard EXE ایجاد کنید. خصوصیت‌های فرم را مطابق جدول ۱۵-۱۲ تنظیم کنید.

جدول ۱۵-۱۲

| مقدار | خصوصیت |
|---------|---------|
| frmmpeg | Name |
| Movie | Caption |

۲ - یک کنترل منو مطابق شکل ۲۹-۱۲ روی فرم قرار دهید سپس کنترل MAM را به جعبه ابزار اضافه کنید. در جعبه ابزار روی آیکن **Active Movie**  دابل کلیک کنید تا کنترل روی فرم قرار گیرد سپس خصوصیت Name آن را روی mammovie تنظیم کنید.

۳ - کنترل mammovie را انتخاب کرده و خصوصیت AutoStart آن را روی مقدار True تنظیم کنید تا در زمانی که فایل صوتی یا تصویری باز می‌شود کنترل MAM آن را پخش کند.



شکل ۲۹-۱۲

- ۴ - خصوصیت ShowPositionControls را در کنترل mammovie روی مقدار True تنظیم کنید تا چهار دکمه دیگر نیز قابل مشاهده و استفاده شوند.
- ۵ - رویداد Click گزینه Open را به صورت زیر تنظیم کنید:

```
Private Sub mnuopen_Click()
```

```
Dim strpath As String
```

```
strpath = InputBox("Enter Path and Filename :", "Input Data")
```

```
mammovie.FileName = strpath
```

```
End Sub
```

در این رویداد پس از دریافت نام و مسیر فایل موسیقی یا ویدئویی و ذخیره‌سازی آن در متغیر strpath، محتویات این متغیر در دستور سوم محتویات این متغیر در خصوصیت FileName کنترل mammovie قرار داده می‌شود تا آن را پخش کند.

۶ - پروژه و فرم را با نام activemovie ذخیره کرده سپس برنامه را اجرا کنید.

۷ - با استفاده از گزینه Open در منوی File، مسیر و نام یک فایل mpg را تعیین کنید. سپس در هنگام پخش فایل مورد نظرتان، عملکرد دکمه‌ها و سایر بخش‌ها در کنترل mammovie را بررسی نمایید.

۸ - از برنامه خارج شده و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه activemovie را برای انواع دیگر فایل‌ها آزمایش کنید.

۸-۱۲ شی تصویر (Picture)

با استفاده از شی تصویر نیز می‌توانید یک تصویر را روی فرم یا در کنترل کادر تصویر نمایش دهید. به منظور ایجاد یک شی تصویر از نوع داده Picture استفاده می‌شود. با شی تصویر می‌توان انواع فایل‌های گرافیکی مانند BMP، GIF، ICO و JPG را نمایش داد.



مثال ۵: می‌خواهیم رویه‌ای بنویسیم که بتواند یک تصویر را با استفاده از شی تصویر نمایش دهد.

به این منظور می‌توان از یک شی تصویر همراه با تابع LoadPicture استفاده کرد به این ترتیب رویه به صورت زیر تعریف می‌شود:

```
Public Sub showpicture(strpath As String)
```

```
Dim objpicture As Picture
```

```
Set objpicture = LoadPicture(strpath)
```

```
Set picshow.Picture = objpicture
```

```
End Sub
```

این رویه با یک آرگومان (strpath) برای دریافت مسیر و نام فایل تعریف شده است سپس داخل رویه، یک شی از نوع Picture با نام objpicture با دستور Dim تعریف شده است در مرحله بعد با استفاده از دستور Set و تابع LoadPicture تصویر موردنظر در شی تصویر objpicture بارگذاری می‌شود و در پایان با دستور شی تصویر در کنترل کادر تصویر picshow نمایش داده می‌شود.

برای فراخوانی رویه ShowPicture می‌توان به صورت زیر عمل نمود:

```
Call showpicture (" C:\WinXP\Coffee Bean.Bmp")
```

اگر لازم باشد تصویر روی فرم نمایش داده شود می‌توان از متد PaintPicture استفاده کرد. به عنوان مثال دستور سوم در رویه showpicture را می‌توان به این صورت تغییر داد تا تصویر روی فرم نمایش داده شود.

```
PaintPicture objpic, 20, 50, 3000, 4000
```

این دستور شی تصویر را در مختصات $X = 20$ و $Y = 50$ با عرض 3000 و ارتفاع 4000 روی فرم نمایش می‌دهد.



• **نکته** • از متد PaintPicture می‌توان در کنترل کادر تصویر و شیء چاپگر نیز استفاده کرد.
 • متد LoadPicture از انواع فایل‌های گرافیکی مانند BMP، ICO، GIF، JPG و ... پشتیبانی می‌کند.



تمرین:

پروژه‌ای طراحی کنید که با استفاده از آن بتوان هر تصویر دلخواهی را با استفاده از یک کادر محاوره Open مشاهده کرد به علاوه امکان بزرگ یا کوچک کردن تصویر مورد نظر نیز وجود داشته باشد.



مثال ۶: می‌خواهیم پروژه‌ای طراحی کنیم که به وسیله آن بتوان ترسیمات رسم شده روی فرم را در یک فایل ذخیره کرد. برای این کار عملیات زیر را به ترتیب انجام دهید:
 ۱ - یک پروژه از نوع Standard EXE ایجاد کنید و فرم و کنترل‌های آن را مطابق شکل ۱۲-۳۰ و جدول ۱۲-۱۶ و ۱۲-۱۷ تنظیم کنید.



شکل ۱۲-۳۰

جدول ۱۲-۱۶ خصوصیات فرم

| مقدار | خصوصیت |
|--------------|---------|
| frmdrawing | Name |
| Save Picture | Caption |

جدول ۱۲-۱۷ خصوصیات کنترل‌ها

| کنترل / خصوصیت | Command Button | Command Button | Picture Box |
|----------------|----------------|----------------|-------------|
| Name | cmddrawline | cmdsave | picdrawing |
| Caption | &Line | &Save | _____ |

۲ - رویداد Click دکمه Line را به صورت زیر تنظیم کنید.

Private Sub cmddrawline_Click()

ScaleMode = vbPixels

AutoRedraw = True

Line (10, 10)-(200, 200), vbRed

End Sub

در این رویداد عملیات رسم یک خط انجام می‌گیرد.
۳ - دستور زیر را در رویداد Click دکمه Save تایپ کنید:

SavePicture Image, "C:\drawing.bmp"

رویه فرعی SavePicture می‌تواند ترسیمات انجام شده روی فرم یا کنترل کادر تصویر را در یک فایل گرافیکی از نوع BMP ذخیره کند. آرگومان اول در این رویه فرعی خصوصیت Image فرم یا کنترل کادر تصویری است که ترسیمات روی آن انجام شده است و آرگومان دوم مسیر و نام فایلی است که ترسیمات در آن ذخیره می‌شود.

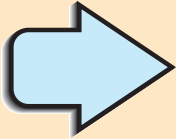
۴ - پروژه و فرم را با نام savepicture ذخیره کرده سپس آنرا اجرا کنید و ابتدا روی دکمه Line و سپس Save کلیک کنید. در پایان به مسیر: C بروید و فایل drawing.bmp را با برنامه نقاشی ویندوز باز کرده و محتویات آنرا بررسی کنید.

۵ - از برنامه خارج شوید و به پنجره ویژوال بیسیک بازگردید.

تمرین:



پروژه‌ای طراحی کنید که به وسیله آن بتوان هر نوع فایل گرافیکی مانند GIF، JPG، CUR، ICO و ... را در فایل جدیدی با قالب بندی BMP ذخیره کرد.



Learn in English

Circle Method

Draws a circle, ellipse, or arc on an object.

Syntax

object.Circle [*Step*] (x, y), radius, [*color, start, end, aspect*]

The **Circle** method syntax has the following object qualifier and parts:

| Part | Description |
|------------|---|
| object | Optional. Object expression that evaluates to an object in the Applies To list. If object is omitted, the Form with the focus is assumed to be object. |
| Step | Optional. Keyword specifying that the center of the circle, ellipse, or arc is relative to the current coordinates given by the CurrentX and CurrentY properties of object. |
| (x, y) | Required. Single values indicating the coordinates for the center point of the circle, ellipse, or arc. The ScaleMode property of object determines the units of measure used. |
| radius | Required. Single value indicating the radius of the circle, ellipse, or arc. The ScaleMode property of object determines the unit of measure used. |
| color | Optional. Long integer value indicating the RGB color of the circle's outline. If omitted, the value of the ForeColor property is used. You can use the RGB function or QBColor function to specify the color. |
| start, end | Optional. Single-precision values. When an arc or a partial circle or ellipse is drawn, start and end specify (in radians) the beginning and end positions of the arc. The range for both is -2π radians to 2π radians. The default value for start is \cdot radians; the default for end is $2 * \pi$ radians. |
| aspect | Optional. Single-precision value indicating the aspect ratio of the circle. The default value is 1.0, which yields a perfect (non-elliptical) circle on any screen. |

| واژه نامه | |
|-------------|-----------------|
| Circle | دایره |
| Component | جزء |
| Current | جاری |
| Draw | رسم کردن، کشیدن |
| Line | خط |
| Movie | فیلم |
| Multimedia | چند رسانه‌ای |
| Orientation | جهت |
| Point | نقطه |
| Quality | کیفیت |
| Radius | شعاع دایره |
| Scale | مقیاس |
| Style | سبک، روش |

خلاصه مطالب

- ترسیمات گرافیکی را می‌توان روی فرم یا کنترل PictureBox یا چاپگر ایجاد کرد.
- به وسیله خصوصیت ScaleMode می‌توان مقیاس را در محورهای مختصات تنظیم کرد.
- به وسیله خصوصیت ScaleLeft و ScaleTop می‌توان مختصات نقطه مبنا را تنظیم کرد.
- خصوصیات ScaleHeight و ScaleWidth مقیاس را می‌توان در سیستم مختصات فرم یا کنترل PictureBox تنظیم کرد.
- به وسیله خصوصیت Scale نیز می‌توان مختصات نقطه مبنا و مقیاس سیستم مختصات را در فرم یا کنترل مورد نظر تنظیم کرد.
- به وسیله متد PSet می‌توان هر نقطه دلخواهی را در موقعیت مورد نظر ترسیم کرد.
- به وسیله متد Line می‌توان انواع خطوط مستطیل و مستطیل توپر را در مکان مورد نظر رسم کرد.
- متد Circle می‌تواند انواع دایره، بیضی یا کمانی از دایره و یا بیضی را رسم کند.
- به وسیله متد Point می‌توان رنگ یک نقطه را به دست آورد. این متد رنگ نقطه مورد نظر را به صورت یک عدد از نوع Long باز می‌گرداند.
- متدهای CurrentX و CurrentY موقعیت جاری مکان‌نما را در صفحه ترسیمات معین می‌کنند. خصوصیت CurrentX موقعیت جاری مکان‌نما را در محور X و CurrentY موقعیت جاری مکان‌نما را در محور Y معین می‌کنند.

- متد Cls می‌تواند صفحه ترسیمات را پاک کند.
- به وسیله متد Print هر نوع عبارت، می‌توان مقدار متغیرها و خصوصیات را نمایش داد.
- با استفاده از تابع Spc در متد Print می‌توان به تعداد مورد نظر فضای خالی ایجاد کرد.
- با استفاده از تابع Tab در متد Print می‌توان اطلاعات نمایشی را در ستون مورد نظر نمایش داد.
- متدهای TextWidth و TextHeight می‌توانند به ترتیب عرض و ارتفاع عبارت رشته‌ای را که دریافت می‌کنند معین کنند.
- به وسیله خصوصیت AutoRedraw می‌توان از حذف ترسیمات موجود در یک پنجره جلوگیری به عمل آورد.
- خصوصیت DrawMode می‌تواند رنگ ترسیمات گرافیکی را تعیین کند.
- به وسیله خصوصیت DrawStyle می‌توان نوع خطوط را در متدهای گرافیکی نظیر Circle و Line تعیین کرد.
- با استفاده از خصوصیت DrawWidth می‌توان ضخامت خطوط و نقاط را در متد Circle، Line و PSet مشخص کرد.
- خصوصیت FillStyle، نوع و حالت خطوطی که سطح یک شکل گرافیکی نظیر مستطیل، دایره یا بیضی را پر می‌کند تعیین می‌کند.
- به وسیله خصوصیت FillColor می‌توان رنگ مورد نظر را برای پوشاندن سطح یک شکل گرافیکی نظیر مستطیل، دایره یا بیضی تعیین کرد.
- تابع QBColor می‌تواند با دریافت یک عدد صحیح، رنگ متناظر آن را به صورت یک عدد از نوع Long تعیین کند.
- به وسیله تابع RGB می‌توانید ترکیبات رنگی مورد نظرتان را بر اساس مقدار سه رنگ اصلی آبی، قرمز و سبز ایجاد کنید.
- با استفاده از شیء Printer و خصوصیات و متدهای این شیء، می‌توان اطلاعات مورد نظر را به وسیله چاپگر روی کاغذ چاپ کرده و بر نحوه انجام عملیات چاپ نظارت کرد.
- به وسیله کنترل MCI و MAM می‌توان انواع فایل‌های صدا، موسیقی و ویدئویی را پخش کرد.
- به وسیله رویه‌های LoadPicture و PaintPicture می‌توان فایل‌های گرافیکی را روی کنترل کادر تصویر یا فرم نمایش داد.
- به وسیله رویه SavePicture می‌توان ترسیمات روی فرم یا کنترل کادر تصویر را روی دیسک ذخیره کرد.

آزمون نظری

۱ - به وسیله کدام خصوصیت می توان یکی از مقیاس های استاندارد را در ویژوال بیسیک انتخاب کرد؟

الف - ScaleMode ب - thgScaleHei ج - ScaleWidth د - Scale

۲ - کدام گزینه برای رسم یک مستطیل توپر توسط متد Line مناسب است؟

الف - B ب - BF ج - F د - FB

۳ - کدام تابع در متد Print می تواند اطلاعات نمایشی را در ستون مشخصی نمایش دهد؟

الف - Spc ب - Point ج - Tab د - Spcb

۴ - خروجی فرمان Circle (150,200), 1000 2,,, به صورت است.

الف - بیضی افقی ب - بیضی عمودی

ج - نیمدایره د - ربع دایره

۵ - به وسیله کدام خصوصیت می توان ضخامت ترسیمات را تعیین کرد؟

الف - DrawMode ب - DrawStyle

ج - DrawWidth د - FillStyle

۶ - کدام متد می تواند عملیات چاپ را خاتمه دهد؟

الف - EndDoc ب - Port ج - NewPage د - KillDoc

۷ - کدام تابع می تواند ترکیبات رنگی را بر اساس رنگ های اصلی ایجاد کند؟

الف - QBColor ب - BackColor ج - RGB د - ColorMode

۸ - کدام خصوصیت، رنگ مورد نظر برای پر کردن یک مستطیل یا دایره را معین می کند؟

الف - FillStyle ب - FillColor

ج - ColorMode د - ForeColor

۹ - به وسیله کدام خصوصیت موقعیت جاری مکان نما در صفحه ترسیمات در جهت محور

افقی معین می شود؟

الف - CurrentX ب - CurrentY

ج - Point د - گزینه های الف و ب صحیح هستند.

۱۰ - واحد اندازه گیری پیش فرض در سیستم مختصات ویژوال بیسیک عبارت است

از:

الف - cm ب - inch ج - pixel د - twip

۱۱ - رویه SavePicture ترسیمات را به صورت فایل‌های گرافیکی از نوع ذخیره می‌کند.

الف - JPG ب - BMP ج - ICO د - CUR

۱۲ - کدام یک از کنترل‌ها امکان پخش فایل‌های ویدئویی با پسوند MPG را دارند؟

الف - MCI ب - Image ج - MAM د - Picture

۱۳ - Which of the following answer can be used in circle method to draw an ellipse?

a- radius b- start c- aspect d- end

۱۴ - سیستم مختصات در ویژوال بیسیک را توضیح داده و نحوه تنظیم آن را بیان کنید.

۱۵ - نحوه ساخت و استفاده از شیء تصویر را توضیح دهید.

۱۶ - نحوه استفاده از شیء چاپگر را بیان کرده و خصوصیت‌ها و متدهای آن را توضیح دهید.

۱۷ - کاربرد متدهای PSet، Line، Circle و Point را با ذکر مثال توضیح دهید.

۱۸ - کنترل‌های چند رسانه‌ای را در ویژوال بیسیک نام برده و نحوه استفاده از آن‌ها

را برای پخش فایل‌های موسیقی و نمایش فیلم توضیح دهید.

۱۹ - خصوصیت‌های CurrentY، CurrentX، DrawMode، DrawStyle و DrawWidth را

توضیح دهید.

۲۰ - نحوه ذخیره‌سازی ترسیمات گرافیکی را با استفاده از شیء تصویر توضیح دهید.

آزمون عملی

۱ - پروژه‌ای طراحی کنید که شکل زیر را نمایش دهد.



- ۲ - پروژه‌ای طراحی کنید که کاربر بتواند انواع بیضی، دایره و کمان را با توجه به مقادیر دلخواهش مشاهده کند.
- ۳ - پروژه‌ای طراحی کنید که با دریافت معدل دانش‌آموزان ۵ کلاس ۲۰ نفره میانگین معدل هر کلاس را محاسبه کند، سپس نتایج را به صورت نمودار میله‌ای نمایش دهد.
- ۴ - پروژه‌ای طراحی کنید که به طور پیوسته دایره‌ها با اندازه و رنگ‌های متفاوت به صورت تصادفی ایجاد و نمایش داده شوند.
- ۵ - پروژه‌ای طراحی کنید که مختصات سه رأس یک مثلث را دریافت نموده، سپس آن را ترسیم کند.
- ۶ - پروژه‌ای طراحی کنید که حرکت یک اتومبیل را شبیه‌سازی کند.
- ۷ - پروژه‌ای طراحی کنید که به وسیله آن بتوان هر تصویر دلخواه را با اندازه مورد نظر مشاهده کرد و در صورت تمایل آن را به صورت فایل گرافیکی از نوع BMP ذخیره نمود.

توانایی انجام یک پروژه عملی

هدف‌های رفتاری

پس از مطالعه این واحد کار از فراگیر انتظار می‌رود که:

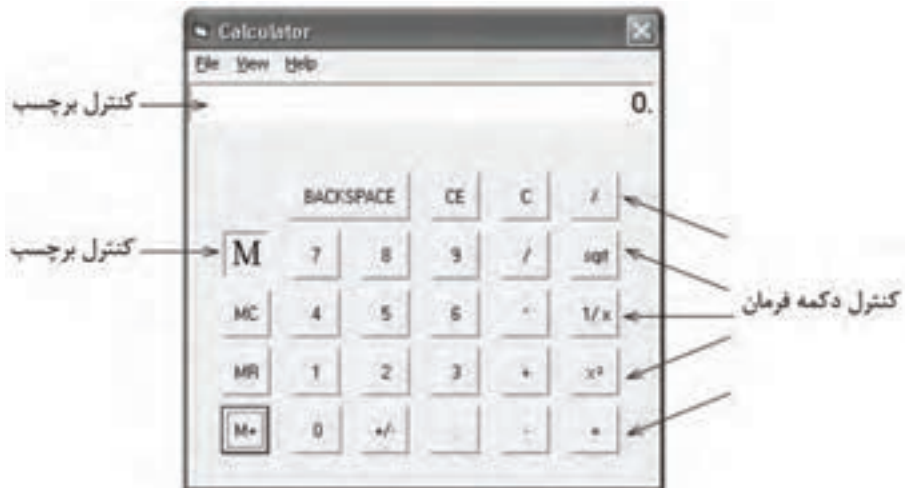
۱ - توانایی انجام یک پروژه برنامه‌نویسی را داشته باشد.

در این واحد کار یک پروژه برنامه‌نویسی واقعی را به صورت عملی انجام خواهید داد تا آنچه را که قبلاً فرا گرفته‌اید به کار بگیرید. یک ماشین حساب مطابق شکل ۱-۱۳ ایجاد کنید که توانایی انجام محاسبات چهار عمل اصلی، محاسبه مجذور و جذر، قرینه اعداد، درصد و معکوس اعداد را مانند یک ماشین حساب الکترونیکی داشته باشد. برای این کار عملیات زیر را به ترتیب انجام دهید:



شکل ۱-۱۳

۱- برنامه ویژوال بیسیک را اجرا کرده، یک پروژه از نوع Standard EXE ایجاد کنید که شامل یک فرم، کنترل دکمه‌های فرمان، دو برچسب مطابق شکل ۲-۱۳ و یک کنترل زمان‌سنج باشد، سپس خصوصیات آن‌ها را مطابق جداول ۱-۱۳ و ۲-۱۳ تنظیم کنید.



شکل ۲-۱۳

دکمه‌های فرمان دارای عرض و ارتفاع ۵۰۰ بوده و عنوان، نام و خصوصیات Top و Left آن‌ها مطابق شکل ۲-۱۳ است و خصوصیت Cancel تمام آن‌ها را مساوی False و خصوصیت Default آن‌ها (بجز کنترل =) را نیز روی False تنظیم کنید.

جدول ۱-۱۳ خصوصیات فرم

| مقدار | خصوصیت |
|----------------|-------------|
| frmmain | Name |
| 1-Fixed Single | BorderStyle |
| Calculator | Caption |

جدول ۲-۱۳ خصوصیات کنترل‌ها

| کنترل خصوصیت | Label | Label | Timer |
|-----------------|----------|------------|----------|
| Name | lblvalue | lblmemory | tmrdelay |
| Alignment | 1 | 2 | - |
| BorderStyle | 1 | 1 | - |
| Enabled | True | True | False |
| Interval | - | - | 150 |
| AutoSize | False | False | - |
| Caption | 0 | بدون عنوان | - |

به بخش تعاریف فرم بروید و متغیرهای مورد نیاز برنامه‌ها را به صورت زیر تعریف کنید:

Dim dblno1 As Double, dblno2 As Double

Dim dblresult As Double, dblmemory As Double

Dim strop As String, strno As String

می‌بینید که متغیرهای مختلفی در بخش تعاریف، تعریف می‌شوند. متغیرهای dblno1 و dblno2 از نوع Double و به ترتیب برای ذخیره‌سازی عددی که قبل از انتخاب عملگر و عددی که بعد از انتخاب عملگر وارد می‌شوند، به کار می‌روند و متغیر dblresult از نوع Double و برای نگهداری نتیجه محاسبات روی متغیرهای dblno1 و dblno2 استفاده می‌شوند.

به علاوه متغیر `dblmemory` به عنوان حافظه ماشین حساب و متغیر رشته‌ای `strop` برای ذخیره کردن عمل ریاضی که کاربر انتخاب کرده، استفاده می‌شوند و به عنوان آخرین متغیر، یک متغیر رشته‌ای دیگر به نام `strno` عدد وارد شده به ماشین حساب را به صورت یک رشته در خود قرار می‌دهد تا در جریان اجرای برنامه با تبدیل آن به مقدار عددی در متغیرهای `dblno۱` و `dblno۲` ذخیره شود.

برای جلوگیری از تکرار دستورات در رویه‌های دیگر لازم است دو رویه را قبل از سایر رویه‌ها ایجاد نمایید. بنابراین اولین رویه را با نام `dellblvalue` و به صورت زیر تعریف کنید:

```
Private Sub dellblvalue()
```

```
    If (strop = "*" Or strop = "/" Or strop = "+" Or strop = "-") And strno = "" Then
```

```
        lblvalue.Caption = ""
```

```
    End If
```

```
End Sub
```

این رویه بررسی می‌کند که آیا قبل از ورود یک رقم، یکی از دکمه‌های چهار عمل اصلی فشرده شده است یا خیر. چون در صورت انتخاب یک عملگر، ورود هر یک از ارقام به عنوان عدد دوم محسوب می‌شود، در نتیجه باید عدد وارد شده که قبل از انتخاب عملگر مورد نظر در کنترل برچسب نمایش داده می‌شود، پاک شود. این عمل با دستور `lblvalue.Caption = ""` انجام خواهد شد. پس از تعریف رویه `dellblvalue` که در رویدادهای `Click` ارقام ۱ الی ۹ و نقطه اعشار اجرا می‌شود، رویه دوم را با نام `inputnumber` و به صورت زیر تعریف کنید. این رویه وظیفه دریافت ارقام و تشکیل عدد مورد نظر کاربر را به عهده دارد.

```
Private Sub inputnumber (strdigit As String)
```

```
    If strdigit <> "." Then
```

```
        If Left(strno, 1) = "0" And Mid(strno, 2, 1) <> "." Then
```

```
            strno = Mid(strno, 2) + strdigit
```

```
        Else
```

```
            strno = strno + strdigit
```

```
        End If
```

```
Else
```

```
    If InStr(1, strno, ".") = 0 Then
```

```
        If Len(lblvalue.Caption) = 0 Then strno = "0." Else _
```

strno=strno + "."

End If

End If

lblvalue.Caption = strno

End Sub

این رویه یک آرگومان به نام `strdigit` از نوع رشته‌ای دارد که در زمان فراخوانی با توجه به این که روی کدام دکمه کلیک شده است، رقم متناظر با آن را به رویه می‌دهد. پس از کلیک روی یکی از ارقام یک تا ۹ یا نقطه اعشار، رویه فراخوانی می‌شود و کاراکتر متناظر با رقم مربوطه در آرگومان `strdigit` ذخیره می‌شود. سپس با استفاده از یک دستور `If` مقدار این آرگومان بررسی می‌شود و در صورتی که مقدار آن مخالف کاراکتر نقطه باشد به این معنی است که دکمه مربوطه یکی از ارقام یک تا ۹ می‌باشد، در نتیجه `If` موجود در بخش `Then` اجرا می‌شود و رشته `strno` را که مجموعه ارقام وارد شده می‌باشد، مورد بررسی قرار می‌دهد.

با استفاده از توابع `Left` و `Mid` رشته `strno` بررسی می‌شود. اگر اولین و دومین کاراکتر رشته `strno` برابر با صفر و نقطه باشند ("0.") به این معنی است که عدد بین صفر و یک است، بنابراین رقم `strdigit` به رشته "0" اضافه می‌شود. اما در صورت نادرست بودن شرط رقم `strdigit` به وسیله دستور `strno=strno + strdigit` به مجموعه ارقام قبلی در رشته `strno` اضافه می‌شود. در واقع این `If` برای کنترل ارقام صفری است که قبل از نقطه اعشار و سایر ارقام غیرصفر وارد می‌شوند. به این صورت از دریافت رقم صفر قبل از سایر ارقام غیرصفر جلوگیری به عمل می‌آید.

اما اگر دکمه کلیک شده، نقطه اعشار باشد `If` موجود در بخش `Else` اولین `If` اجرا می‌شود. این `If` از ورود نقطه اعشار اضافه جلوگیری می‌کند. با استفاده از تابع `InStr` وجود نقطه اعشار در ارقامی که تاکنون وارد شده‌اند یعنی متغیر `strno` بررسی می‌شود. اگر رشته `strno` فاقد نقطه اعشار باشد `If` موجود در آن اجرا می‌شود و با استفاده از تابع `Len` طول رشته بررسی می‌شود و اگر این مقدار برابر صفر باشد به این معنی است عدد با مقدار 0 شروع می‌شود، در غیر این صورت نقطه اعشار با دستور `strno=strno+"0"` به ارقام قبلی اضافه خواهد شد. در پایان این رویه، ارقامی که تاکنون وارد شده‌اند با استفاده از دستور `lblvalue.Caption=strno` در کنترل برچسب نمایش داده می‌شوند. به این ترتیب می‌توان انواع اعداد کوچک‌تر یا بزرگ‌تر از یک را وارد کرد.

در این مرحله امکان ورود اعداد را فراهم کرده، رویداد Click دکمه‌های رقمی ۱ تا ۹ و نقطه اعشار را به صورت زیر تنظیم کنید:

```
Private Sub cmd1_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("1")
```

```
End Sub
```

```
Private Sub cmd2_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("2")
```

```
End Sub
```

```
Private Sub cmd3_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("3")
```

```
End Sub
```

```
Private Sub cmd4_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("4")
```

```
End Sub
```

```
Private Sub cmd5_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("5")
```

```
End Sub
```

```
Private Sub cmd6_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("6")
```

```
End Sub
```

```
Private Sub cmd7_Click()
```

```
    Call dellblvalue
```

```
    Call inputnumber ("7")
```

```
End Sub
```



```
Private Sub cmd8_Click()  
    Call dellblvalue  
    Call inputnumber ("8")  
End Sub
```

```
Private Sub cmd9_Click()  
    Call dellblvalue  
    Call inputnumber ("9")  
End Sub
```

```
Private Sub cmddot_Click()  
    Call dellblvalue  
    Call inputnumber (".")  
End Sub
```

۵ - در این مرحله رویداد دکمه فرمان رقم صفر را به صورت زیر تنظیم کنید:

```
Private Sub cmd0_Click()  
If strop="*" Or strop = "+" Or strop="-" Or Stop="/" Then  
    If Left(strno, 1)<>"0" Or Mid(strno, 2, 1) = "." Then  
        strno = strno + "0"  
    Else  
        strno = ""  
        lblvalue.Caption = ""  
    End If  
Else  
    If Left(strno,1)<> "0" Or Mid(strno, 2,1) = "." Then strno=strno+"0"  
End If  
lblvalue.Caption = strno  
End Sub
```

این رویداد از یک دستور If تشکیل شده است که در بخش Then و Else آن نیز مجدداً یک دستور If به کار رفته است. در اولین دستور If فشرده شدن یکی از دکمه‌های مربوط به چهار عمل اصلی بررسی می‌شود. در صورتی که هیچ یک از آن‌ها استفاده نشده باشند نتیجه شرط نادرست بوده و If موجود در بخش Else اجرا می‌شود. در این If نیز با استفاده از تابع Left و Mid بررسی می‌شود که آیا اولین کاراکتر در متغیر strno صفر یا دومین کاراکتر در آن نقطه اعشار است یا خیر؟

۶ - اکنون رویداد Click مربوط به چهار عمل اصلی را به صورت زیر تنظیم کنید:

```
Private Sub cmdmultiply_Click()
```

```
    If strop = "" Then
```

```
        strop = "*"
```

```
        dblno1 = Val(strno)
```

```
        strno = ""
```

```
    Else
```

```
        If strno <> "" Then
```

```
            dblno2 = Val(strno)
```

```
            Call Compute
```

```
            dblno1 = Val(strno)
```

```
            strno = ""
```

```
        End If
```

```
        strop = "*"
```

```
    End If
```

```
End Sub
```

```
Private Sub cmddivision_Click()
```

```
    If strop = "" Then
```

```
        strop = "/"
```

```
        dblno1 = Val(strno)
```

```
        strno = ""
```

```
    Else
```

```
        If strno <> "" Then
```

```
            dblno2 = Val(strno)
```

```
            Call Compute
```

```
            dblno1 = Val(strno)
```

```
            strno = ""
```

```
        End If
```

```
        strop = "/"
```

```
    End If
```

```
End Sub
```

```
Private Sub cmdplus_Click()
```

```
    If strop = "" Then
```

```
strop = "+"  
dblno1 = Val(strno)  
strno = ""  
Else  
If strno <> "" Then  
    dblno2 = Val(strno)  
    Call compute  
    dblno1 =Val(strno)  
    strno = ""  
End If  
strop = "+"  
End If  
End Sub
```

```
Private Sub cmdminus_Click()  
If strop = "" Then  
    strop = "-"  
    dblno1 = Val(strno)  
    strno = ""  
Else  
    If strno <> "" Then  
        dblno2 = Val(strno)  
        Call compute  
        dblno1 =Val(strno)  
        strno = ""  
    End If  
    strop = "-"  
End If  
End Sub
```

دستورات این چهار رویداد مشابه یکدیگر هستند. در این رویدادها ابتدا با استفاده از دستور If مقدار متغیر strop بررسی می‌شود تا مشخص شود تاکنون عملگری انتخاب شده است یا خیر (چون ممکن است عبارتی که به ماشین حساب داده می‌شود مشابه $2*3+4$ یا مانند عبارت $3=2*$ باشد). در صورتی که شرط درست باشد (یعنی عملگری تاکنون انتخاب نشده است) ابتدا کاراکتر متناسب با دکمه‌ای که کلیک شده است در متغیر strop

ذخیره می‌شود، مثلاً اگر روی دکمه * کلیک کنید کاراکتر "*" در strop ذخیره می‌شود، سپس عددی که قبل از کلیک روی چهار عمل اصلی وارد شده است و در متغیر strno نگهداری می‌شود با استفاده از تابع Val به نوع عددی تبدیل و در متغیر dblno1 به عنوان عدد اول ذخیره می‌شود و مجدداً مقدار "" در strno ذخیره می‌شود تا مقدار متغیر رشته‌ای strno برای دریافت عدد دوم از بین برود.

اما اگر قبلاً یکی از چهار عمل اصلی انتخاب شده باشند دیگر مقدار strop برابر رشته "" نیست و دستورات بخش Else اجرا می‌شوند. در این جا یک If دیگر استفاده می‌شود و اگر مقداری در متغیر strno موجود باشد به این معنی است که عدد دوم نیز وارد شده است. بنابراین در صورتی که عباراتی مانند $3+4*2$ یا مشابه آن‌ها وارد شوند به این معنی است که قبل از دریافت عدد ۴ باید محاسبه $3*2$ انجام شود، پس عدد دوم در متغیر dblno2 ذخیره خواهد شد و با فراخوانی رویه compute محاسبات انجام می‌شود. رویه compute را نیز به صورت زیر در ماژول فرم ایجاد کنید.

Private Sub compute()

Select Case strop

Case "*": dblresult = dblno1 * dblno2

Case "/": dblresult = dblno1 / dblno2

Case "+": dblresult = dblno1 + dblno2

Case "-": dblresult = dblno1 - dblno2

Case "power": dblresult = dblno1 ^ dblno2

End Select

lblvalue.Caption = dblresult

strno = dblresult

End Sub

با فراخوانی این رویه و با استفاده از یک دستور Select Case، مقدار متغیر strop بررسی می‌شود تا با توجه به عمل ریاضی انتخاب شده اعمال ضرب، تقسیم، جمع یا تفریق روی اعداد dblno1 و dblno2 انجام شده و نتیجه در متغیر dblresult ذخیره شود. سپس مقدار این متغیر در خصوصیت Caption برچسب ذخیره می‌شود تا نتیجه محاسبات در پنجره ماشین حساب دیده شود و به همین صورت این مقدار برای انجام محاسبات بعدی در متغیر strno ذخیره می‌شود.

به این ترتیب پس از خاتمه اجرای رویه compute نتیجه محاسبات که در متغیر strno

ذخیره شده در متغیر `dblno1` نیز قرار داده می‌شود تا در محاسبات بعدی به عنوان اولین عدد مورد استفاده قرار گیرد و پس از اجرای `strno=""` و حذف نتیجه محاسبات از متغیر `strno` دستورات `If` موجود در `Else` تمام می‌شود و به عنوان آخرین دستور در این بخش، کاراکتر متناسب با عمل ریاضی انتخاب شده در متغیر `strop` برای استفاده در مراحل بعدی ذخیره می‌شود. به این ترتیب یک دوره کامل از ورود اعداد و انجام محاسبات، کدنویسی و آماده شده است.

۷- اکنون باید ماشین حساب را به‌گونه‌ای تنظیم کنید که در صورت استفاده از عباراتی مانند $12.5+6=$ نیز محاسبات به درستی انجام شود، بنابراین رویداد `Click` دکمه فرمان تساوی را به صورت زیر تنظیم کنید:

```
Private Sub cmdequal_Click()  
    If strop="/" Or strop="+" Or strop="-" Or strop="*" Then  
        dblno2 = Val(strno)  
        Call compute  
    End If  
    strop = ""  
End Sub
```

در این رویداد نیز ابتدا مقدار متغیر `strop` بررسی می‌شود تا در صورت کلیک روی یکی از دکمه‌های چهار عمل اصلی محاسبات انجام شود، بنابراین در صورت درست بودن شرط موجود در دستور `If` ابتدا عدد دوم در متغیر `dblno2` ذخیره می‌شود، سپس رویه `compute` فراخوانی می‌شود تا محاسبات مربوطه انجام شود و محتویات متغیر `strop` برای مراحل بعدی خالی شود. ذکر این نکته ضروری است که عدد اول نیز در این حالت در هنگام کلیک روی یکی از چهار عمل اصلی در متغیر `dblno1` ذخیره می‌شود.

۸- در این مرحله رویداد `Click` دکمه `BACKSPACE` را به صورت زیر تنظیم کنید:

```
Private Sub cmdbackspace_Click()  
    If Len (lblvalue.Caption) = 1 Then  
        lblvalue.Caption = "0"  
        strno=""  
    Else  
        lblvalue.Caption = Left(lblvalue.Caption, Len (lblvalue.Caption)-1)  
        strno = lblvalue.Caption  
    End If  
End Sub
```

می بینید که این رویه نیز با یک دستور If آغاز می شود و با بررسی طول رشته ای که در کنترل برچسب نمایش داده می شود عمل حذف کاراکترها را از مقدار موجود، انجام می دهد. بنابراین اگر طول این رشته برابر یک باشد شرط درست بوده و مقدار خصوصیت Caption کنترل برچسب روی صفر تنظیم شده سپس دستور "strno =" برای اعمال تغییرات در محاسبات نیز اجرا می شود. در غیر این صورت با استفاده از توابع Len و Left یک کاراکتر از سمت راست رشته حذف می شود و مقدار خصوصیت Caption در متغیر strno نیز ذخیره می شود تا تغییرات انجام شده در محاسبات نیز اعمال شود.

۹ - اکنون رویداد مربوط به دکمه درصد را به صورت زیر تنظیم کنید:

```
Private Sub cmdpercent_Click()  
    lblvalue.Caption = Val(lblvalue.Caption) * 0.01  
    strno = lblvalue.Caption  
    strop = ""  
End Sub
```

در این رویداد نیز ابتدا با تبدیل خصوصیت Caption کنترل برچسب به عدد و ضرب آن در عدد ۰/۰۱ مقدار را براساس درصد محاسبه کرده و سپس مقدار به دست آمده در متغیر strno برای محاسبات بعدی ذخیره می شود و با اجرای دستور "strno =" از ایجاد تداخل و اشتباه که ممکن است در اثر انتخاب چند عملگر رخ دهد، جلوگیری به عمل می آید.

۱۰ - مانند مرحله ۸ رویداد Click مربوط به دکمه های به توان رسانی (X^2)، جذر (sqrt)، قرینه سازی (+/-) و معکوس سازی ($1/x$) را به این صورت تنظیم کنید. این چهار رویه عملکردی مشابه رویه رویداد Click دکمه درصد دارند.

```
Private Sub cmdpower_Click()  
    lblvalue.Caption = Val(lblvalue.Caption) ^ 2  
    strno = lblvalue.Caption  
    strop = ""  
End Sub
```

```
Private Sub cmdsign_Click()  
    If strno <> "" Then  
        lblvalue.Caption = -Val(lblvalue.Caption)  
        strno = lblvalue.Caption  
    End If  
End Sub
```

```
Private Sub cmdsqrt_Click()  
    lblvalue.Caption = Sqr(Val(lblvalue.Caption))  
    strno = lblvalue.Caption  
    strop= ""
```

End Sub

```
Private Sub cmdinverse_Click()  
    If Val(lblvalue.Caption) <> 0 Then lblvalue.Caption=1/ Val(lblvalue.Caption)  
    strno = lblvalue.Caption  
    strop= ""
```

End Sub

۱۱ - اکنون رویه‌های رویداد مربوط به دکمه‌های $M+$, MR و MC را به صورت زیر تنظیم کنید. دکمه $M+$ برای ذخیره اعداد و نتیجه محاسبات در حافظه ماشین حساب، دکمه MR برای استفاده از محتویات حافظه ماشین حساب در عملیات موردنظر و دکمه MC برای پاک کردن حافظه ماشین حساب به کار می‌روند.

```
Private Sub cmdm_Click()  
    If strop = "/" Or strop = "+" Or strop = "-" Or strop="*" Then  
        dblno2 = Val(strno)  
        Call compute  
    End If  
    dblmemory = dblmemory + Val(lblvalue.Caption)  
    lblmemory.Caption = "M"  
    strop= ""  
End Sub
```

```
Private Sub cmdmc_Click()  
    dblmemory = 0  
    lblmemory.Caption = ""  
End Sub
```

```
Private Sub cmdmr_Click()  
    lblvalue.Caption = str(dblmemory)  
    If dblmemory=0 Then  
        strno= ""  
    Else
```

```
strno=str(dbلمemory)
```

```
End If
```

```
End Sub
```

رویه () cmdm_Click برای ذخیره مقادیر در حافظه ماشین حساب به کار می‌رود و با یک دستور If شروع می‌شود تا در صورت استفاده از اعمال ریاضی قبل از کلیک روی دکمه M+، ابتدا محاسبات با فراخوانی رویه compute انجام شود، سپس نتیجه عملیات به محتویات قبلی حافظه ماشین حساب که در متغیر dbلمemory قرار دارد، اضافه می‌شود و با نمایش کاراکتر M در کنترل برچسب مربوطه، کاربر از ذخیره شدن اطلاعات در حافظه ماشین حساب مطلع می‌شود. البته در صورت عدم انتخاب یک عمل ریاضی دستورات بخش If اجرا نشده و طبیعی است که محاسباتی نیز انجام نمی‌شود، اما عدد موجود در نمایشگر ماشین حساب به حافظه اضافه خواهد شد. آخرین دستور در این رویه یعنی strop="" نیز همان کاربرد قبلی در سایر رویه‌های مشابه را دارد. رویه () cmdmc_Click برای حذف مقدار ذخیره شده در حافظه ماشین حساب به کار می‌رود و با اجرای آن ابتدا مقدار dbلمemory صفر می‌شود سپس حرف M از کنترل برچسب مربوط به حافظه برداشته می‌شود.

رویه () cmdmr_Click نیز برای استفاده از مقدار ذخیره شده در حافظه در زمان انجام محاسبات به کار می‌رود و با تبدیل مقدار متغیر dbلمemory به مقدار رشته‌ای، آن را در کنترل برچسب ماشین حساب نمایش داده و بعد دستور If را برای اعمال تغییرات در محاسبات اجرا می‌کند.

۱۲ - در این مرحله رویداد دکمه‌های C و CE را به صورت زیر تنظیم کنید. همان‌طور که می‌دانید دکمه C تمام اعداد ورودی، عملگرهای انتخاب شده و محتویات حافظه ماشین حساب را پاک می‌کند و دکمه CE عمل ریاضی انتخاب شده پس از ورود اولین عدد را از بین می‌برد، اما عدد اول وارد شده را حذف نمی‌کند.

```
Private Sub cmdc_Click()
```

```
strno = ""
```

```
lblvalue.Caption = "0"
```

```
strop = ""
```

```
dblno1 = 0
```

```
dblno2 = 0
```

```
dblresult = 0
```

```
dbلمemory = 0
```



```
lblmemory.Caption = ""  
End Sub  
Private Sub cmdce_Click()  
If strop="/" Or strop="+" Or strop="-" Or strop="*" Then  
    lblvalue.Caption=dblno1  
    strno=dblno1  
    strop=""  
End If  
End Sub
```

رویداد () cmdce_Click مربوط به دکمه C می‌شود و همان‌طور که می‌بینید کلیه متغیرهای عددی، رشته‌ای، حافظه و نمایشگر ماشین حساب را پاک می‌کند.

رویداد () cmdce_Click نیز نحوه انتخاب عملگرهای ریاضی را بعد از ورود عدد اول بررسی می‌کند و در صورت درست بودن شرط، متغیر dblno1 را در خصوصیت Caption کنترل برچسب ماشین حساب قرار داده و متغیر strop را خالی می‌کند تا عملگر انتخاب شده از بین رفته و عملیات به مرحله بعد از ورود عدد اول بازگردد.

۱۳ - اکنون باید رویدادهای صفحه کلید را نیز به گونه‌ای تنظیم کنید که در صورت استفاده از صفحه کلید نیز ماشین حساب قابل استفاده باشد. برای این کار ابتدا خصوصیت KeyPreview را برای فرم پروژه روی مقدار True و خصوصیت Style کلیه دکمه‌های فرمان را روی مقدار 1-Graphical تنظیم کنید، سپس رویداد KeyPress فرم را به صورت زیر تنظیم نمایید:

```
Private Sub Form_KeyPress(KeyAscii As Integer)  
    Select Case KeyAscii  
        Case 8:  
            Call cmdbackspace_Click  
            cmdbackspace.BackColor = vbRed  
            tmrdelay.Enabled = True  
        Case 48:  
            Call cmd0_Click  
            cmd0.BackColor = vbRed  
            tmrdelay.Enabled = True  
        Case 49:  
            Call cmd1_Click  
            cmd1.BackColor = vbRed
```

tmrdelay.Enabled = True

Case 50:

Call cmd2_Click

cmd2.BackColor = vbRed

tmrdelay.Enabled = True

Case 51:

Call cmd3_Click

Cmd3.BackColor = vbRed

tmrdelay.Enabled = True

Case 52:

Call cmd4_Click

cmd4.BackColor = vbRed

tmrdelay.Enabled = True

Case 53:

Call cmd5_Click

cmd5.BackColor = vbRed

tmrdelay.Enabled = True

Case 54:

Call cmd6_Click

cmd6.BackColor = vbRed

tmrdelay.Enabled = True

Case 55:

Call cmd7_Click

cmd7.BackColor = vbRed

tmrdelay.Enabled = True

Case 56:

Call cmd8_Click

cmd8.BackColor = vbRed

tmrdelay.Enabled = True

Case 57:

Call cmd9_Click

cmd9.BackColor = vbRed

tmrdelay.Enabled = True

Case 43:

Call cmdplus_Click

```
cmdplus.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 45:

```
Call cmdminus_Click
```

```
cmdminus.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 46:

```
Call cmddot_Click
```

```
Cmddot.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 42:

```
Call cmdmultiply_Click
```

```
cmdmultiply.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 47:

```
Call cmddivision_Click
```

```
cmddivision.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 37:

```
Call cmdpercent_Click
```

```
cmdpercent.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 67, 99:

```
Call cmdc_Click
```

```
cmdc.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 69, 101:

```
Call cmdce_Click
```

```
cmdce.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 80, 112:

```
Call cmdpower_Click
```

```
cmdpower.BackColor = vbRed
```

```
tmrdelay.Enabled = True
```

Case 83, 115:

```
Call cmdsqrt_Click
```

`cmdsqr.BackColor = v`

`bRed`

`tmrdelay.Enabled = True`

Case 73, 105:

Call `cmdinverse_Click`

`cmdinverse.BackColor = vbRed`

`tmrdelay.Enabled = True`

Case 68, 100:

Call `cmdmc_Click`

`cmdmc.BackColor = vbRed`

`tmrdelay.Enabled = True`

Case 77, 109:

Call `cmdm_Click`

`cmdm.BackColor = vbRed`

`tmrdelay.Enabled = True`

Case 82, 114:

Call `cmdmr_Click`

`cmdmr.BackColor = vbRed`

`tmrdelay.Enabled = True`

Case 71, 103:

Call `cmdsign_Click`

`cmdsign.BackColor = vbRed`

`tmrdelay.Enabled = True`

Case 61:

Call `cmdequal_Click`

`cmdequal.BackColor = vbRed`

`tmrdelay.Enabled = True`

End Select

End Sub

مشاهده می‌کنید که با استفاده از یک دستور `Select Case` مقدار آرگومان `KeyAscii` رویداد `KeyPress` بررسی می‌شود و با توجه به کلید فشرده شده رویداد `Click` متناظر با کلیدی که قبلاً طراحی شده است، فراخوانی می‌شود. این کار سبب می‌شود تا از تکرار دستورات و شلوغ شدن مازول فرم جلوگیری شود و در هر `Case` رویداد `Click` مربوط به دکمه‌ای که فشرده شده است فراخوانی شود. پس از خاتمه اجرای رویه رویداد کلید

فشرده شده، با استفاده از خصوصیت BackColor کنترل مربوطه، رنگ کنترل به قرمز تغییر داده می‌شود. سپس خصوصیت Enabled کنترل زمان‌سنج روی True تنظیم می‌شود تا زمان‌سنج شروع به کار کند و پس از گذشت زمان کوتاهی رنگ کنترل را به حالت اول بازگرداند و خصوصیت Enabled خود را False می‌کند تا برای دفعه بعد آماده شود. به این صورت با فشردن کلیدهای صفحه کلید روی پنجره ماشین حساب، فشرده شدن کلیدها با تغییر رنگ قابل مشاهده است. سپس رویداد Timer کنترل زمان‌سنج را به صورت زیر تنظیم کنید:

```
Private Sub tmrdelay_Timer()
```

```
    cmddot.BackColor = &H8000000F  
    cmd0.BackColor = &H8000000F  
    cmd1.BackColor = &H8000000F  
    cmd2.BackColor = &H8000000F  
    cmd3.BackColor = &H8000000F  
    cmd4.BackColor = &H8000000F  
    cmd5.BackColor = &H8000000F  
    cmd6.BackColor = &H8000000F  
    cmd7.BackColor = &H8000000F  
    cmd8.BackColor = &H8000000F  
    cmd9.BackColor = &H8000000F  
    cmdplus.BackColor = &H8000000F  
    cmdminus.BackColor = &H8000000F  
    cmdmultiply.BackColor = &H8000000F  
    cmddivision.BackColor = &H8000000F  
    cmdbackspace.BackColor = &H8000000F  
    cmdc.BackColor = &H8000000F  
    cmdce.BackColor = &H8000000F  
    cmdpercent.BackColor = &H8000000F  
    cmdpower.BackColor = &H8000000F  
    cmdsqrt.BackColor = &H8000000F  
    cmdinverse.BackColor = &H8000000F  
    cmdmc.BackColor = &H8000000F  
    cmdm.BackColor = &H8000000F  
    cmdmr.BackColor = &H8000000F
```

```
cmdsign.BackColor = &H8000000F
cmdequal.BackColor = &H8000000F
tmrdelay.Enabled = False
```

End Sub

۱۴- رویداد Load فرم را نیز به صورت زیر تنظیم کنید تا مقادیر متغیرها مقداردهی

اولیه شود:

```
Private Sub Form_Load()
    cmdc_Click()
```

End Sub

۱۵- نوار منو و گزینه‌های موجود در آن را مطابق شکل ۱۳-۳ و ۱۳-۴ ایجاد کنید.

روی کادر علامت Enabled گزینه Black و White کلیک کنید و آن‌ها را از حالت

انتخاب خارج کنید. به علاوه روی کادر علامت Checked گزینه Right Alignment کلیک

کنید و آنرا انتخاب نمایید.



شکل ۱۳-۳



شکل ۱۳-۴

۱۶- اکنون به ماژول فرم بروید و رویداد Click گزینه‌های Clear و Exit را به صورت

نشان داده شده در صفحه بعد تنظیم کنید. گزینه Clear همان عملکرد دکمه C را دارد و

انتخاب گزینه Exit سبب نمایش یک کادر پیغام خواهد شد که در صورت کلیک روی

دکمه OK برنامه خاتمه یافته و در صورت کلیک روی دکمه Cancel کاربر به پنجره ماشین

حساب باز می‌گردد.

```
Private Sub mnuclear_Click()
    Call cmdc_Click
```

End Sub

```
Private Sub mnuexit_Click()  
    Dim intanswer As Integer  
    intanswer = MsgBox("Do You Want To Exit ?",vbOKCancel _  
        +vbInformation, "Exit")  
    If intanswer = vbOK Then Unload Me  
End Sub
```

۱۷- در این مرحله رویداد مربوط به گزینه‌های زیرمنوهای ForeGround و Background

را برای تغییر رنگ قلم و زمینه نمایشگر ماشین حساب، به صورت زیر تنظیم کنید:

```
Private Sub mnublack_Click()  
    lblvalue.ForeColor = vbBlack  
    mnublack.Enabled = False  
    mnublue.Enabled = True  
    mnured.Enabled = True  
End Sub
```

```
Private Sub mnublue_Click()  
    lblvalue.ForeColor = vbBlue  
    mnublack.Enabled = True  
    mnublue.Enabled = False  
    mnured.Enabled = True  
End Sub
```

```
Private Sub mnured_Click()  
    lblvalue.ForeColor = vbRed  
    mnublack.Enabled = True  
    mnublue.Enabled = True  
    mnured.Enabled = False  
End Sub
```

```
Private Sub mnuwhite_Click()  
    lblvalue.BackColor = vbWhite  
    mnuwhite.Enabled = False  
    mnugreen.Enabled = True  
    mnumagenta.Enabled = True  
End Sub
```

```
Private Sub mnugreen_Click()
```

```
    lblvalue.BackColor = vbGreen
```

```
    mnuwhite.Enabled = True
```

```
    mnugreen.Enabled = False
```

```
    mnumagenta.Enabled = True
```

```
End Sub
```

```
Private Sub mnumagenta_Click()
```

```
    lblvalue.BackColor = vbMagenta
```

```
    mnuwhite.Enabled = True
```

```
    mnugreen.Enabled = True
```

```
    mnumagenta.Enabled = False
```

```
End Sub
```

۱۸- اکنون رویداد Click مربوط به زیرمنوی Alignment را تنظیم کنید. برای این کار رویه‌های دو گزینه این زیر منو را به صورت زیر تنظیم کنید:

```
Private Sub mnuleft_Click()
```

```
    lblvalue.Alignment = vbLeftJustify
```

```
    mnuleft.Checked = True
```

```
    mnuright.Checked = False
```

```
End Sub
```

```
Private Sub mnuright_Click()
```

```
    lblvalue.Alignment = vbRightJustify
```

```
    mnuleft.Checked = False
```

```
    mnuright.Checked = True
```

```
End Sub
```

در این صورت با انتخاب هر یک از این گزینه‌ها، چک مارک در کنار گزینه‌ای که کلیک شده است، قرار داده می‌شود و چک مارک گزینه دیگر حذف می‌شود. به علاوه با توجه به گزینه کلیک شده، خصوصیت Alignment نمایشگر ماشین حساب نیز تنظیم می‌شود. ۱۹ - در این مرحله لازم است تا برای گزینه About در منوی Help دستورات لازم را تایپ کنید. هدف از این گزینه ارایه شماره نگارش نرم‌افزار به کاربر است، بنابراین یک فرم با مشخصات ارایه شده در جدول ۳-۱۳ و مطابق با شکل ۵-۱۳ به پروژه اضافه کنید.



شکل ۱۳-۵

جدول ۱۳-۳

| کنترل خصوصیت | Image | Label | Command Button | Form |
|-----------------|--------------|---------------------------|-------------------|----------|
| Name | imgcalculate | lblversion | cmdok | frmabout |
| Caption | - | Calculator Version 1-1383 | &OK | About |
| Stretch | True | - | - | - |
| BorderStyle | - | ۱ | - | ۴ |
| Alignment | - | ۲ | - | - |

۲۰ - پس از اضافه کردن فرم جدید آن را با نام About ذخیره کنید و سپس رویداد

Click دکمه OK و Unload فرم About را به صورت زیر تنظیم کنید:
 Private Sub cmdok_Click()

 Unload Me

End Sub

Private Sub Form_Unload(Cancel As Integer)

 frmmain.Enabled = True

End Sub

۲۱ - در این مرحله رویه رویداد Click گزینه About را از منوی Help، به صورت زیر

تنظیم کنید. اولین دستور سبب نمایش فرم About می‌شود و دستور دوم، فرم اصلی را غیرفعال می‌کند و تا زمانی که کاربر روی دکمه OK فرم About کلیک نکند یا آن را نبندد نمی‌تواند از ماشین حساب استفاده کند.

Private Sub mnuabout_Click()

 frmabout.Show

frmmain.Enabled = False

End Sub

۲۲ - پروژه و فرم اصلی را با نام calculator ذخیره کنید، سپس آنرا اجرا کرده و برای حالت‌های مختلف آزمایش کنید.

۲۳ - به اجرای پروژه خاتمه داده و پنجره ویژوال بیسیک را ببندید.

آزمون پایانی (نظری)

۱ - مقدار K پس از خاتمه حلقه زیر چقدر است؟

K=1

Do While (K<3)

K=K+ 1

Loop

الف- ۴ ب- ۳ ج- ۲ د- ۱

۲ - در صورتی که Option Base 1 باشد تعداد اعضای آرایه 5 (A) چقدر خواهد بود؟

الف- ۵ ب- ۶ ج- ۷ د- ۸

۳ - خروجی دستور (3, «AliReza») Right چیست؟

الف- «Ali» ب- «eza» ج- «iReza» د- «AliRe»

۴ - کدام نوع داده می‌تواند هر نوع داده‌ای را نگهداری کند؟

الف- Variant ب- Single ج- Boolean د- Long

۵ - کدام خاصیت کنترل کادر متن، تراز متن را در کنترل تعیین می‌کند؟

الف- Alignment ب- MaxLength

ج- BorderStyle د- Appearance

۶ - خروجی دستور (Weekday (Now)) WeekdayName چیست؟

الف- شماره روز جاری ب- شماره هفته جاری

ج- نام روز جاری د- نام ماه جاری

۷- اگر آرایه grade(10 To 17, 2 To 9) تعریف شود، آنگاه حاصل LBound(grade, 2) چیست؟

الف- ۱۰ ب- ۱۷ ج- ۲ د- ۹

۸ - خروجی دستور (vbBinaryCompare, «This is a book.» , «i» , «I» , ,) Replace چیست؟

الف- «ThIs Is a book» ب- «This Is a book»

ج- «s is a book» د- «s Is a book»

۹ - با فرض این‌که strname = "SuperComputer" باشد، حاصل عبارت

3,5) Left (strname) چیست؟

الف - «pu»

ب - «Sup»

ج - «per»

د - «put»

۱۰ - کدام خاصیت در کنترل کادر لیست، شماره عضو انتخاب شده در آن را تعیین می کند؟

الف - ListIndex

ب - ListCount

ج - Index

د - Item

۱۱ - کدام کاراکتر در متد Print سبب می شود اطلاعات نمایشی بدون فاصله در کنار

هم قرار گیرند؟

الف - ;

ب - ,

ج - !

د - #

۱۲ - حاصل عبارت (0, «ware» , «ware and software», InStr(5, «ware and software»)) چیست؟

الف - ۵

ب - صفر

ج - ۱۸

د - ۱

۱۳ - در صورتی که یک متغیر در بخش تعاریف مازول فرم با دستور Dim معرفی

شود

الف - در زمان اجرای برنامه پیام خطا نمایش داده می شود.

ب - متغیر در تمام رویه های مازول کد قابل استفاده است.

ج - متغیر در تمام رویه های فرم قابل استفاده است.

د - متغیر در تمام مازول های پروژه قابل استفاده است.

۱۴ - حاصل عبارت (MonthName (Month(Date), True)) چیست؟

الف - تاریخ جاری سیستم

ب - نام ماه جاری به صورت کامل

ج - نام ماه جاری به صورت خلاصه

د - شماره ماه جاری

۱۵ - در صورت تعریف یک رویه با کلمه کلیدی Public

الف - رویه مورد نظر در تمام مازول های فرم و مازول های کد قابل استفاده است.

ب - رویه مورد نظر در تمام مازول های فرم قابل استفاده است.

ج - امکان فراخوانی آن به وسیله رویه های رویداد وجود ندارد.

د- امکان فراخوانی آن به وسیله رویه‌های آماده ویژوال بیسیک وجود ندارد.

۱۶ - حاصل عبارت (Asc(Chr(65)) چیست؟

الف - «A»

ب - ۶۵

ج - یک رشته خالی

د - پیام خطا نمایش داده می‌شود.

۱۷ - یک رویه فرعی با کدام یک از عبارت‌های زیر خاتمه می‌یابد؟

الف - Function

ب - Call

ج - End Sub

د - Sub

۱۸ - کدام خاصیت در کنترل MCI امکان نمایش فیلم را در کنترل PictureBox فراهم می‌کند؟

الف - DeviceType

ب - hWndDisplay

ج - FileName

د - Command

۱۹ - کدام کاراکتر برای تعریف یک متغیر از نوع رشته‌ای به کار می‌رود؟

الف - &

ب - #

ج - !

د - \$

۲۰ - حاصل عبارت "B">"A" And 3<5 Or (True) Not چیست؟

الف - True

ب - False

ج - صفر

د - ۱

۲۱ - اگر الگوریتم زیرمجموع و میانگین تعداد مشخصی از اعداد را محاسبه کند،

کدام گزینه در مورد مرحله ۶ درست است؟

۱- شروع

۲- $c \leftarrow 0$ و $Sum \leftarrow 0$ و $I \leftarrow 1$

۳- N را دریافت کن

۴- M را دریافت کن

۵- $Sum \leftarrow Sum + M$

۶-

۷- اگر $I \leq N$ آن‌گاه برو به مرحله ۴

۸- $\frac{Sum}{N}$ را نمایش بده

۹- پایان

الف- $I \leftarrow I + 1$ ب- $I \leftarrow I - 1$ ج- $N \leftarrow N + 1$ د- $N \leftarrow \text{Sum} - 1$

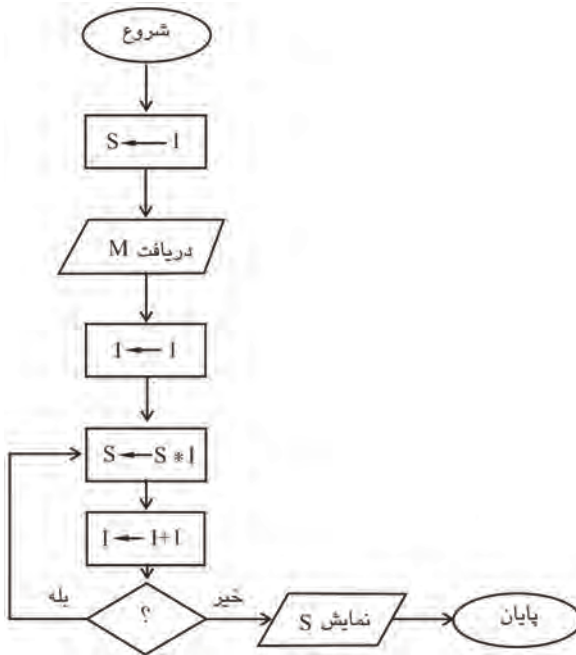
۲۲- در صورتی که بخواهیم حاصل ضرب اعداد ۱ تا M را به دست آوریم کدام گزینه را به جای علامت سؤال قرار می دهیم؟

الف- $I \leq M$

ب- $I \geq M$

ج- $M > I$

د- $M < I$



۲۳- کدام رویداد توانایی تشخیص کلیدهای تابعی را ندارد؟

الف- KeyUp

ب- KeyDown

ج- KeyPress

د- KeyDown و KeyUp

۲۴- الگوریتم زیر چه عملی انجام می دهد؟

۱- شروع

۲- x را دریافت کن

۳- اگر $x < 0$ است آن گاه $x \leftarrow -x$

۴- x را نمایش بده

۵- پایان

الف- محاسبه قدر مطلق x

ب- قرینه اعداد مثبت

ج- جزء صحیح x

د- الگوریتم هیچ عملی انجام نمی دهد.

۲۵ - کدامیک از موارد زیر از ویژگی‌های الگوریتم محسوب نمی‌شود؟
الف- رعایت ترتیب
ب- خاتمه پذیر بودن

ج- ارایه جزئیات کافی
د- ارایه فرمول‌های پیچیده

۲۶ - کدام نوع داده برای ذخیره‌سازی سن یک فرد مناسب‌تر است؟

الف - Integer ب - Long ج - Word د - Byte

۲۷ - کدام خاصیت، اولویت انجام رویدادهای صفحه کلید را بین فرم و کنترل‌های

موجود در آن تعیین می‌کند؟

الف- خاصیت Key در فرم
ب- خاصیت KeyPreview در فرم

ج- خاصیت Key در کنترل‌ها
د- خاصیت KeyPreview در کنترل‌ها

۲۸ - حاصل عبارت $2 + 3 * 2 + 18 \div 3$ چیست؟

الف- ۱۳
ب- ۱۴

ج- ۱۵
د- ۱۶

۲۹ - عنوان یک فرم به وسیله کدام خاصیت تعیین می‌شود؟

الف - Enabled
ب - Text

ج - Name
د - Caption

۳۰ - به وسیله کدام متد می‌توان رنگ یک نقطه از تصویر را به دست آورد؟

الف - PSet
ب - Point

ج - Tab
د - Spc

آزمون پایانی (عملی)

پروژه‌ای طراحی کنید که یک پازل را شبیه‌سازی کند.

پاسخنامه

پیش آزمون

(ب-۱) (ب-۲) (ب-۳)

آزمون نظری واحد کار ۱

(ج-۱) (ب-۲) (الف-۳)
(ج-۲) (الف-۶) (الف-۷)
(الف-۵) (الف-۱۰) (الف-۱۱)
(ب-۴) (د-۸) (ب-۱۱)

آزمون نظری واحد کار ۲

(ج-۱) (ب-۲) (د-۳)
(ب-۵) (الف-۶) (الف-۷)
(الف-۹) (ب-۱۰) (b-۱۱)
(ب-۴) (ج-۸)

آزمون نظری واحد کار ۳

(ب-۱) (ب-۲) (ب-۳)
(ب-۵) (ج-۶) (د-۷)
(ج-۹) (ب-۱۰) (د-۱۱)
(ب-۱۳) (الف-۱۴) (ب-۱۵)
(b-۱۷) (الف-۱۲) (d-۱۶)

آزمون نظری واحد کار ۴

(ج-۱) (ب-۲) (ب-۳)
(ج-۵) (ج-۶) (الف-۷)
(c-۹) (ب-۴) (ب-۸)

آزمون نظری واحد کار ۵

(ج-۱) (ب-۲) (ب-۳)
(ب-۵) (الف-۶) (ج-۷)
(د-۹) (ج-۱۰) (b-۱۱)
(الف-۴) (ب-۸)

آزمون نظری واحد کار ۶

(ب-۱) (ب-۲) (ج-۳)
(الف-۵) (د-۶) (ب-۷)
(ج-۴) (ج-۸)

- (ج-۱۲) (ب-۱۱) (ج-۱۰) (ب-۹)
 (د-۱۴) (الف-۱۳)

آزمون نظری واحد کار ۷

- (ب-۴) (د-۳) (ب-۲) (ب-۱)
 (ب-۸) (ب-۷) (ب-۶) (الف-۵)
 (د-۱۲) (الف-۱۱) (ج-۱۰) (د-۹)
 (ج-۱۳)

آزمون نظری واحد کار ۸

- (ب-۴) (الف-۳) (ب-۲) (ج-۱)
 (ج-۸) (ج-۷) (ب-۶) (ج-۵)
 (ا-۹)

آزمون نظری واحد کار ۹

- (الف-۴) (ب-۳) (ج-۲) (ب-۱)
 (الف-۸) (ج-۷) (ب-۶) (ب-۵)
 (ب-۱۱) (الف-۱۰) (الف-۹)

آزمون نظری واحد کار ۱۰

- (ب-۴) (ب-۳) (الف-۲) (ب-۱)
 (ب-۸) (د-۷) (ج-۶) (د-۵)
 (د-۹)

آزمون نظری واحد کار ۱۱

- (ب-۴) (ج-۳) (د-۲) (ب-۱)
 (ب-۸) (الف-۷) (ج-۶) (الف-۵)
 (ب-۱۱) (الف-۱۰) (د-۹)

آزمون نظری واحد کار ۱۲

- (ب-۴) (ج-۳) (ب-۲) (الف-۱)
 (ب-۸) (ج-۷) (د-۶) (ج-۵)
 (ج-۱۲) (ب-۱۱) (د-۱۰) (الف-۹)
 (ج-۱۳)

آزمون پایانی (نظری)

| | | | |
|----------|----------|----------|----------|
| (ب-۱) | (۲-الف) | (۳-ب) | (۴-الف) |
| (۵-الف) | (۶-ج) | (۷-ج) | (۸-الف) |
| (۹-د) | (۱۰-الف) | (۱۱-الف) | (۱۲-ج) |
| (۱۳-ج) | (۱۴-ج) | (۱۵-الف) | (۱۶-ب) |
| (۱۷-ج) | (۱۸-ب) | (۱۹-د) | (۲۰-ب) |
| (۲۱-الف) | (۲۲-الف) | (۲۳-ج) | (۲۴-الف) |
| (۲۵-د) | (۲۶-د) | (۲۷-ب) | (۲۸-ج) |
| (۲۹-د) | (۳۰-ب) | | |

ضمیمه ۱

نصب نرم‌افزار Visual Basic 6.0

برای نصب زبان برنامه‌نویسی ویژوال بیسیک ابتدا باید دیسک‌های CD نرم‌افزار Microsoft Visual Studio 6 Enterprise Edition نگارش نهایی (Enterprise Edition) را تهیه کنید. در این جا چگونگی نصب این نرم‌افزار از روی دیسک CD، توضیح داده می‌شود. مدل‌های دیگر نیز تقریباً دارای مراحل نصب مشابهی هستند.


CD نصب نرم‌افزار را داخل درایو قرار دهید. اگر CD شما از نوع - Aut Run باشد بعد از قرار گرفتن CD در درایو مربوطه، برنامه نصب ویندوز به طور خودکار اجرا می‌شود و اولین کادر محاوره مربوط به نصب نرم‌افزار ظاهر می‌شود، اما در صورتی که برنامه نصب به طور خودکار اجرا نشد برنامه My Computer یا Window Explorer را اجرا کرده و در ریشه درایو CD خود، فایل Setup.exe را پیدا کنید و سپس روی آن دابل کلیک کنید. برنامه نصب اجرا شده و اولین کادر محاوره نصب نمایش داده می‌شود (شکل ۱).



شکل ۱

اکنون به ترتیب این مراحل را دنبال کنید:

- ۱ - ابتدا روی دکمه Next در اولین کادر محاوره کلیک کنید.
- ۲ - پس از مشاهده شکل ۲ روی دکمه انتخاب I accept the agreement و سپس روی دکمه Next کلیک نمایید.

در صورت تمایل می‌توانید در هر کادرمحاوره با کلیک روی دکمه Back به کادر
نکته  محاوره قبل بازگردید.



شکل ۲

۳ - در شکل ۳ در دو کادر متنی که در بخش: Please enter your product's ID number: کادر محاوره قرار گرفته‌اند، شماره (ID) نرم‌افزار را تایپ کنید. سپس در کادر متن‌های Your name: و Your company's name: اسامی دلخواه خود را بنویسید و در پایان مجدداً روی دکمه Next کلیک کنید.



شکل ۳

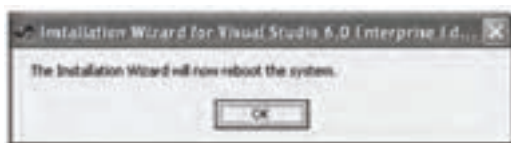


نکته در صورت نادرست بودن شماره ID امکان نصب برنامه وجود ندارد. اگر شماره ID ارایه شده در این جا قابل استفاده نباشد، می‌توانید از فایل‌های موجود در CD، شماره ID را پیدا کنید یا آن را از فروشنده درخواست کنید.

- ۴ - در کادرمحاوره شکل ۴ روی دکمه Next کلیک کنید.
- ۵ - در کادرمحاوره شکل ۵ روی دکمه OK کلیک کنید تا کامپیوتر مجدداً راه‌اندازی شود



شکل ۴



شکل ۵

۶ - پس از راه‌اندازی مجدد کامپیوتر کمی صبر کنید، کادرمحاوره‌ای که ظاهر می‌شود (شکل ۶) که روش‌های مختلفی را برای نصب نرم‌افزار پیشنهاد می‌کند، روی دکمه انتخاب Custom و سپس روی دکمه Next کلیک کنید.

۷ - در کادرمحاوره شکل ۷ می‌توانید مسیر نصب نرم‌افزار را انتخاب کنید. برای این کار مسیر دلخواه خود را در کادر متن تایپ کنید یا با استفاده از دکمه Browse این کار را انجام دهید.



شکل ۶



شکل ۷

در این کادر محاوره حداقل فضای مورد نیاز برای نصب نرم افزار و فضای آزاد درایوی که نرم افزار روی آن نصب می شود، قابل مشاهده است. توجه داشته باشید که فضای آزاد روی درایو مورد نظر کمتر از حداقل فضای پیشنهادی نباشد. در پایان این مرحله روی دکمه Next کلیک کنید تا کادر محاوره بعد (شکل ۸) ظاهر شود.



شکل ۸

۸ - در این مرحله، مدتی صبر کنید تا کادر محاوره بعدی نمایش داده شود.



شکل ۹

۹ - در این کادر محاوره (شکل ۹) روی دکمه Continue کلیک کنید تا کادر محاوره مطابق شکل ۱۰ مشاهده شود. در این کادر محاوره روی دکمه OK کلیک کنید تا روند نصب نرم‌افزار ادامه یابد.



شکل ۱۰

نکته اگر مایل هستید عملیات نصب را خاتمه دهید، می‌توانید در کادر محاوره شکل ۹ روی دکمه Exit Setup کلیک کنید و در کادر محاوره‌ای که ظاهر می‌شود (شکل ۱۱) روی دکمه Exit Setup کلیک نمایید و اگر می‌خواهید مجدداً به کادر محاوره قبل (شکل ۹) بازگردید روی دکمه Resume Setup کلیک کنید.



شکل ۱۱

پس از کلیک روی دکمه OK، پنجره نصب نرم‌افزار به صورت شکل ۱۲ در می‌آید. مدتی صبر کنید تا وارد مرحله بعد شوید.



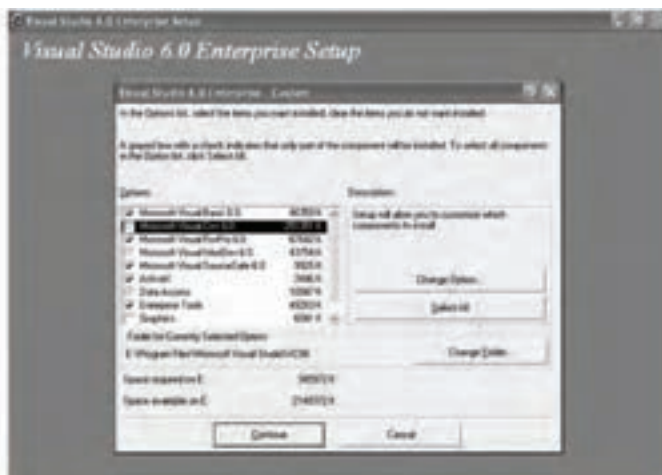
شکل ۱۲

۱۰ - اگر نسخه‌های دیگری از بسته نرم‌افزاری Visual Studio یا همین نگارش را قبلاً نصب کرده باشید، کادر محاوره‌ای مشابه شکل ۱۳ نمایش داده می‌شود. برای اطمینان بیشتر از ایجاد تغییرات در بخش‌های دیگر سیستم خود، در صورت مشاهده این کادر محاوره روی دکمه No کلیک کنید.



شکل ۱۳

۱۱- پس از انجام مرحله ۱۰ پنجره نصب برنامه مجدداً به صورت شکل ۱۲ مشاهده می‌شود، مدتی صبر کنید تا کادر محاوره بعدی مطابق شکل ۱۴ نمایش داده شود.



شکل ۱۴

۱۲- در این مرحله (شکل ۱۴) شما باید زبان‌های برنامه‌نویسی، مانند زبان‌های برنامه‌نویسی VB6، VC++ 6 و VFoxPro6 اجزای گرافیکی مانند انواع فایل‌های گرافیکی که در پروژه‌های برنامه‌نویسی می‌توان از آن‌ها استفاده نمود و سایر ابزارهای مورد نیاز خود را برای نصب انتخاب کنید. چون در این کتاب، زبان برنامه‌نویسی ویژوال بیسیک ۶ مورد نظر است؛ بنابراین با کلیک روی کادر علامت مربوط به موارد زیر آن‌ها را انتخاب کنید یا سایر موارد را در لیست این کادر محاوره از حالت انتخاب خارج کنید. کادرهای علامتی که باید انتخاب شوند عبارتند از:

Microsoft Visual Basic 6.0

ActiveX

Data Access

Graphics


Tools

پس از انجام تنظیمات گفته شده، کادر محاوره مربوطه به صورت شکل ۱۵ در می آید



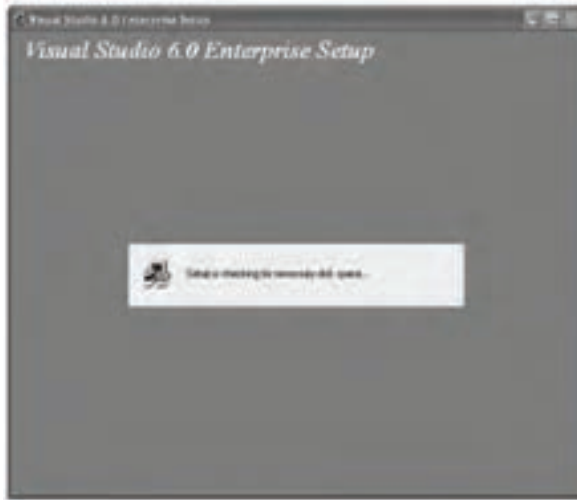
شکل ۱۵

در خاتمه این مرحله روی دکمه Continue کلیک کنید.

 **نکته** • هر یک از گروه‌های موجود در لیست، به چند جزء کوچک‌تر تقسیم می‌شوند که اگر بخواهید در یک گروه فقط بعضی از اجزای آن را نصب کنید، می‌توانید از دکمه Change Option در کادر محاوره شکل ۱۴ استفاده کنید. البته پیشنهاد می‌شود از دستکاری این بخش‌ها بدون آگاهی کافی خودداری نمایید.

• اگر بخواهید تمام گروه‌ها را در لیست ۱۴ انتخاب کنید، روی دکمه Select All کلیک کنید.
• در صورتی که بخواهید مسیر نصب هر یک از گروه‌ها را روی دیسک سخت تغییر دهید می‌توانید از دکمه Change Folder... در کادر محاوره شکل ۱۴ استفاده کنید.

۱۳ - پس از انجام مرحله ۱۲، ابتدا برنامه نصب فضای لازم را برای نصب نرم‌افزار محاسبه می‌کند (شکل ۱۶) و در صورت وجود فضای آزاد به اندازه مناسب، مرحله انجام عملیات کپی فایل‌های نرم‌افزار مطابق شکل ۱۷ آغاز می‌شود.



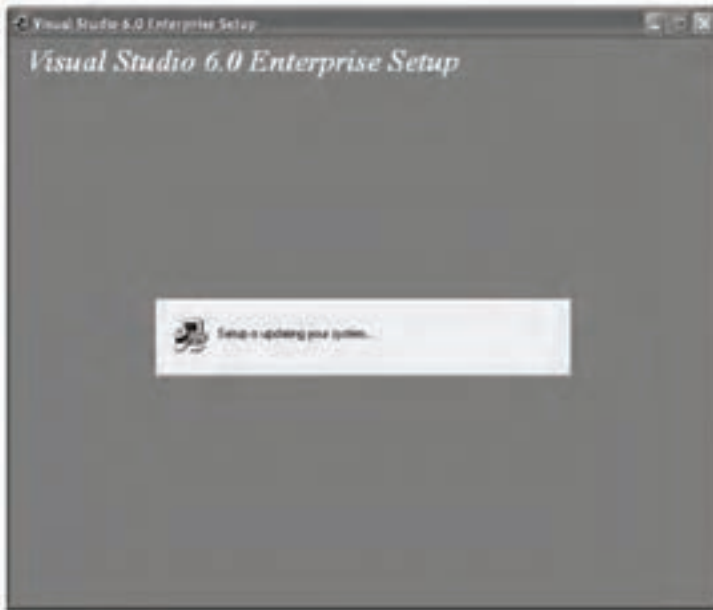
شکل ۱۶

۱۴ - پس از خاتمه مرحله ۱۳ کادر محاوره‌ای مطابق شکل ۱۸ نمایش داده می‌شود که بیانگر به روز کردن اطلاعات مورد نیاز سیستم است.

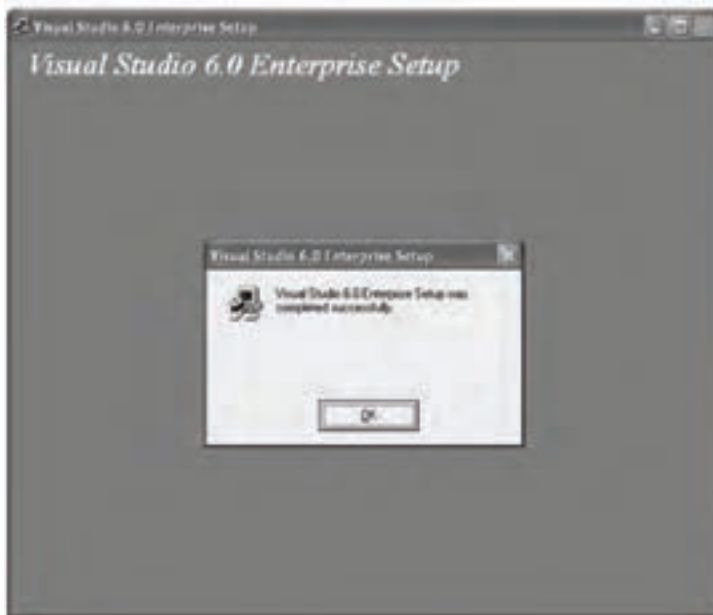
۱۵ - در مرحله بعد کادر محاوره‌ای مطابق شکل ۱۹ نمایش داده می‌شود، مبنی بر این که عمل نصب با موفقیت انجام شده است. پس از کلیک روی دکمه OK کادر محاوره دیگری مطابق شکل ۲۰ نمایش داده می‌شود، در این کادر محاوره روی دکمه Restart Windows کلیک کنید تا سیستم مجدداً راه‌اندازی شود.



شکل ۱۷



شکل ۱۸



شکل ۱۹



شکل ۲۰

۱۶ - پس از راه‌اندازی مجدد سیستم کادر محاوره دیگری (شکل ۲۱) ظاهر می‌شود. اگر دیسک‌های CD مربوط به راهنمای MSDN مایکروسافت را در اختیار داشته باشید، می‌توانید به وسیله این بخش راهنمای مزبور را نصب کنید تا امکان استفاده از منوی Help ویژوال بیسیک فراهم شود. برای این کار کافی است دیسک‌های CD مربوطه را داخل درایو قرار داده و روی دکمه Next کلیک کنید.



شکل ۲۱

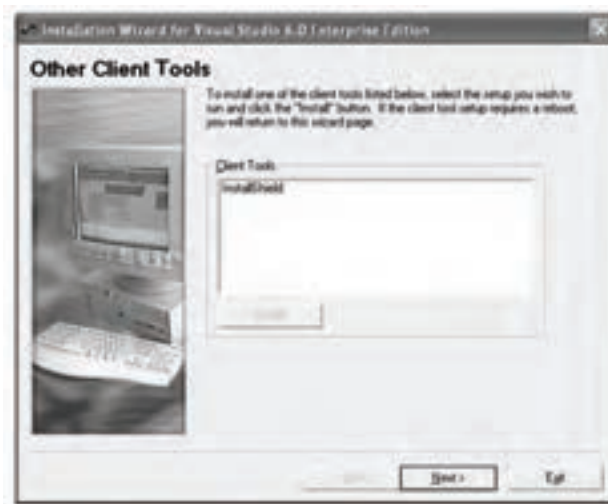
اما اگر دیسک‌های CD مربوطه را در اختیار نداشته باشید باید روی کادر علامت MSDN Install موجود در این کادر محاوره کلیک کنید تا از حالت انتخاب خارج شود؛ سپس روی دکمه Next کلیک کنید.

۱۷ - پس از انجام مرحله ۱۶ (در صورت عدم نصب MSDN) یک کادر پیغام مطابق شکل ۲۲ مبنی بر عدم امکان استفاده از منوی Help در برنامه ویژوال بیسیک نشان داده خواهد شد. روی دکمه Yes کلیک کنید.



شکل ۲۲

۱۸ - در این مرحله کادر محاوره‌ای مطابق شکل ۲۳ برای نصب بعضی از سرویس‌ها نمایش داده می‌شود که به آن‌ها نیازی ندارید. مجدداً روی دکمه Next کلیک کنید.



شکل ۲۳

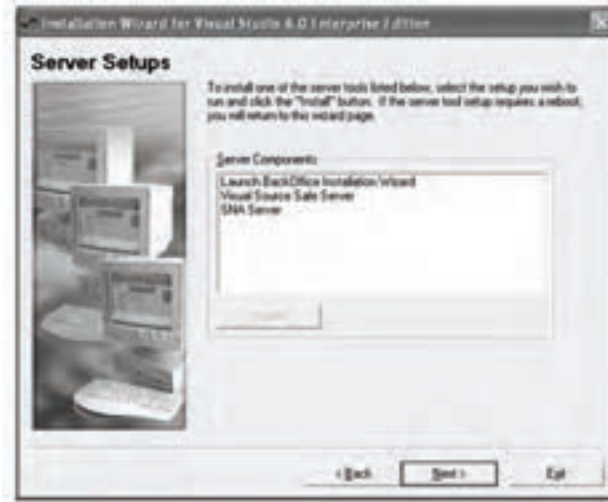
۱۹ - در این مرحله نیز در کادر محاوره شکل ۲۴ روی دکمه Next کلیک کنید، زیرا نیازی به نصب سرویس‌های موجود در لیست این کادر محاوره نخواهید داشت.

۲۰ - اگر مایل هستید در مرحله بعد اطلاعات خود را در سایت اینترنتی مربوطه ثبت کنید (در صورت عدم رعایت حقوق Copy Right امکان استفاده از این بخش میسر نیست).

در پایان روی دکمه Finish کلیک کنید تا نصب نرم‌افزار خاتمه یابد سپس سیستم را restart نمایید.

مطابق شکل ۲۵ در کادر محاوره مربوطه، روی کادر علامت Register Now کلیک کنید تا از حالت انتخاب خارج شود، سپس روی دکمه Finish کلیک کنید.

۲۱- در این مرحله عملیات نصب نرم‌افزار خاتمه یافته است و شما می‌توانید زبان برنامه‌نویسی ویژوال بیسیک را از منوی Start ویندوز اجرا کنید.



شکل ۲۴



شکل ۲۵

ضمیمه ۲

کد کلیدها برای رویدادهای صفحه کلید

کدهای مربوط به کلیدهای حرفی در رویدادهای صفحه کلید

| ثابت رشته‌ای | کد کلید | توضیح |
|--------------|---------|----------------|
| vbKeyA | 65 | کلید A |
| vbKeyB | 66 | کلید B |
| vbKeyC | 67 | کلید C |
| vbKeyD | 68 | کلید D |
| vbKeyE | 69 | کلید E |
| vbKeyF | 70 | کلید F |
| vbKeyG | 71 | کلید G |
| vbKeyH | 72 | کلید H |
| vbKeyI | 73 | کلید I |
| vbKeyJ | 74 | کلید J |
| vbKeyK | 75 | کلید K |
| vbKeyL | 76 | کلید L |
| vbKeyM | 77 | کلید M |
| vbKeyN | 78 | کلید N |
| vbKeyO | 79 | کلید O |
| vbKeyP | 80 | کلید P |
| vbKeyQ | 81 | کلید Q |
| vbKeyR | 82 | کلید R |
| vbKeyS | 83 | کلید S |
| vbKeyT | 84 | کلید T |
| vbKeyU | 85 | کلید U |
| vbKeyV | 86 | کلید V |
| vbKeyW | 87 | کلید W |
| vbKeyX | 88 | کلید X |
| vbKeyY | 89 | کلید Y |
| vbKeyZ | 90 | کلید Z |
| vbKeySpace | 32 | کلید SPACEBAR |
| vbKeyBack | 8 | کلید BACKSPACE |
| vbKeyTab | 9 | کلید TAB |
| vbKeyReturn | 13 | کلید ENTER |
| vbKeyEscape | 27 | کلید ESC |

کدهای مربوط به کلیدهای رقمی در رویدادهای صفحه کلید

| ثابت رشته‌ای | کد کلید | توضیح |
|--------------|---------|--------|
| vbKey0 | 48 | کلید 0 |
| vbKey1 | 49 | کلید 1 |
| vbKey2 | 50 | کلید 2 |
| vbKey3 | 51 | کلید 3 |
| vbKey4 | 52 | کلید 4 |
| vbKey5 | 53 | کلید 5 |
| vbKey6 | 54 | کلید 6 |
| vbKey7 | 55 | کلید 7 |
| vbKey8 | 56 | کلید 8 |
| vbKey9 | 57 | کلید 9 |

کدهای مربوط به کلیدهای رقمی بخش عددی صفحه کلید در

رویدادهای KeyUp و KeyDown

| ثابت رشته‌ای | کد کلید | توضیح |
|---------------|---------|--------|
| vbKeyNumpad0 | 96 | کلید 0 |
| vbKeyNumpad1 | 97 | کلید 1 |
| vbKeyNumpad2 | 98 | کلید 2 |
| vbKeyNumpad3 | 99 | کلید 3 |
| vbKeyNumpad4 | 100 | کلید 4 |
| vbKeyNumpad5 | 101 | کلید 5 |
| vbKeyNumpad6 | 102 | کلید 6 |
| vbKeyNumpad7 | 103 | کلید 7 |
| vbKeyNumpad8 | 104 | کلید 8 |
| vbKeyNumpad9 | 105 | کلید 9 |
| vbKeyMultiply | 106 | کلید * |
| vbKeyAdd | 107 | کلید + |
| vbKeySubtract | 109 | کلید - |
| vbKeyDecimal | 110 | کلید 0 |
| vbKeyDivide | 111 | کلید / |

کدهای مربوط به کلیدهای تابعی در رویدادهای KeyDown و KeyUp

| ثابت رشته‌ای | کد کلید | توضیح |
|--------------|---------|----------|
| vbKeyF1 | 112 | کلید F1 |
| vbKeyF2 | 113 | کلید F2 |
| vbKeyF3 | 114 | کلید F3 |
| vbKeyF4 | 115 | کلید F4 |
| vbKeyF5 | 116 | کلید F5 |
| vbKeyF6 | 117 | کلید F6 |
| vbKeyF7 | 118 | کلید F7 |
| vbKeyF8 | 119 | کلید F8 |
| vbKeyF9 | 120 | کلید F9 |
| vbKeyF10 | 121 | کلید F10 |
| vbKeyF11 | 122 | کلید F11 |
| vbKeyF12 | 123 | کلید F12 |
| vbKeyF13 | 124 | کلید F13 |
| vbKeyF14 | 125 | کلید F14 |
| vbKeyF15 | 126 | کلید F15 |
| vbKeyF16 | 127 | کلید F16 |

کدهای مربوط به سایر کلیدها در رویدادهای KeyUp و KeyDown

| ثابت رشته‌ای | کد کلید | توضیح |
|---------------|---------|-----------------|
| vbKeyShift | 16 | SHIFT key |
| vbKeyControl | 17 | CTRL key |
| vbKeyMenu | 18 | ALT key |
| vbKeyPause | 19 | PAUSE key |
| vbKeyCapital | 20 | CAPS LOCK key |
| vbKeyPageUp | 33 | PAGE UP key |
| vbKeyPageDown | 34 | PAGE DOWN key |
| vbKeyEnd | 35 | END key |
| vbKeyHome | 36 | HOME key |
| vbKeyLeft | 37 | LEFT ARROW key |
| vbKeyUp | 38 | UP ARROW key |
| vbKeyRight | 39 | RIGHT ARROW key |
| vbKeyDown | 40 | DOWN ARROW key |
| vbKeyInsert | 45 | INS key |
| vbKeyDelete | 46 | DEL key |
| vbKeyNumlock | 144 | NUM LOCK key |

کدهای کاراکتر حرفی کوچک برای رویداد KeyPress

| توضیح | کد کلید | ثابت رشته‌ای |
|--------|---------|--------------|
| کلید a | 97 | VbKeya |
| کلید b | 98 | VbKeyb |
| کلید c | 99 | VbKeyc |
| کلید d | 100 | vbKeyd |
| کلید e | 101 | vbKeye |
| کلید f | 102 | vbKeyf |
| کلید g | 103 | vbKeyg |
| کلید h | 104 | vbKeyh |
| کلید i | 105 | vbKeyi |
| کلید j | 106 | vbKeyj |
| کلید k | 107 | vbKeyk |
| کلید l | 108 | vbKeyl |
| کلید m | 109 | vbKeym |
| کلید n | 110 | vbKeyn |
| کلید o | 111 | vbKeyo |
| کلید p | 112 | vbKeyp |
| کلید q | 113 | vbKeyq |
| کلید r | 114 | vbKeyr |
| کلید s | 115 | vbKeys |
| کلید t | 116 | vbKeyt |
| کلید u | 117 | vbKeyu |
| کلید v | 118 | vbKeyv |
| کلید w | 119 | vbKeyw |
| کلید x | 120 | vbKeyx |
| کلید y | 121 | vbKeyy |
| کلید z | 122 | vbKeyz |

کدهای سایر کاراکترها برای رویداد KeyPress

| کد کلید | توضیح |
|---------|-----------|
| 60 | کلید > |
| 62 | کلید < |
| 44 | کلید , |
| 63 | کلید ? |
| 47 | کلید / |
| 92 | کلید \ |
| 124 | کلید |
| 58 | کلید : |
| 59 | کلید ; |
| 34 | کلید » |
| 39 | کلید > |
| 123 | کلید } |
| 125 | کلید { |
| 91 | کلید] |
| 93 | کلید [|
| 61 | کلید = |
| 95 | Underline |
| 40 |) |
| 41 | (|
| 38 | & |
| 94 | ^ |
| 37 | % |
| 36 | \$ |
| 35 | # |
| 64 | @ |
| 33 | ! |
| 126 | ~ |
| 96 | ` |

مقادیر مربوط به آرگومان Shift در رویدادهای KeyUp و KeyDown

| ثابت رشته‌ای | ثابت عددی | توضیح |
|--------------------------------------|-----------|--|
| vbShiftMask | 1 | در صورتی که کلید Shift فشرده شود. |
| vbCtrlMask | 2 | در صورتی که کلید Ctrl فشرده شود. |
| vbAltMask | 4 | در صورتی که کلید Alt فشرده شود. |
| vbShiftMask+vbCtrlMask | 3 | در صورتی که کلیدهای Shift+Ctrl فشرده شوند. |
| vbShiftMask+vbAltMask | 5 | در صورتی که کلیدهای Shift+Alt فشرده شوند. |
| vbCtrlMask+vbAltMask | 6 | در صورتی که کلیدهای Ctrl+Alt فشرده شوند. |
| +vbShiftMask+vbCtrlMask vbAltMask | 7 | در صورتی که کلیدهای Shift+Ctrl+Alt فشرده شوند. |

ضمیمه ۳

کد و پیام‌های خطا در ویژوال بیسیک

| Code | Message |
|------|---|
| 3 | Return without GoSub |
| 5 | Invalid procedure call |
| 6 | Overflow |
| 7 | Out of memory |
| 9 | Subscript out of range |
| 10 | This array is fixed or temporarily locked |
| 11 | Division by zero |
| 13 | Type mismatch |
| 14 | Out of string space |
| 16 | Expression too complex |
| 17 | Can't perform requested operation |
| 18 | User interrupt occurred |
| 20 | Resume without error |
| 28 | Out of stack space |
| 35 | Sub, Function, or Property not defined |
| 47 | Too many DLL application clients |
| 48 | Error in loading DLL |
| 49 | Bad DLL calling convention |
| 51 | Internal error |
| 52 | Bad file name or number |
| 53 | File not found |
| 54 | Bad file mode |
| 55 | File already open |
| 57 | Device I/O error |
| 58 | File already exists |
| 59 | Bad record length |
| 61 | Disk full |
| 62 | Input past end of file |
| 63 | Bad record number |
| 67 | Too many files |

| Code | Message |
|------|--|
| 68 | Device unavailable |
| 70 | Permission denied |
| 71 | Disk not ready |
| 74 | Can't rename with different drive |
| 75 | Path/File access error |
| 76 | Path not found |
| 91 | Object variable or With block variable not set |
| 92 | For loop not initialized |
| 93 | Invalid pattern string |
| 94 | Invalid use of Null |
| 97 | Can't call Friend procedure on an object that is not an instance of the defining class |
| 98 | A property or method call cannot include a reference to a private object, either as an argument or as a return value |
| 298 | System DLL could not be loaded |
| 320 | Can't use character device names in specified file names |
| 321 | Invalid file format |
| 322 | Can't create necessary temporary file |
| 325 | Invalid format in resource file |
| 327 | Data value named not found |
| 328 | Illegal parameter; can't write arrays |
| 335 | Could not access system registry |
| 336 | Component not correctly registered |
| 337 | Component not found |
| 338 | Component did not run correctly |
| 360 | Object already loaded |
| 361 | Can't load or unload this object |
| 363 | Control specified not found |
| 364 | Object was unloaded |
| 365 | Unable to unload within this context |
| 368 | The specified file is out of date. This program requires a later version |
| 371 | The specified object can't be used as an owner form for Show |
| 380 | Invalid property value |
| 381 | Invalid property-array index |
| 382 | Property Set can't be executed at run time |
| 383 | Property Set can't be used with a read-only property |
| 385 | Need property-array index |
| 387 | Property Set not permitted |
| 393 | Property Get can't be executed at run time |
| 394 | Property Get can't be executed on write-only property |
| 400 | Form already displayed; can't show modally |
| 402 | Code must close topmost modal form first |
| 419 | Permission to use object denied |
| 422 | Property not found |
| 423 | Property or method not found |
| 424 | Object required |
| 425 | Invalid object use |
| 429 | Component can't create object or return reference to this object |

| Code | Message |
|-------|--|
| 430 | Class doesn't support Automation |
| 432 | File name or class name not found during Automation operation |
| 438 | Object doesn't support this property or method |
| 440 | Automation error |
| 442 | Connection to type library or object library for remote process has been lost |
| 443 | Automation object doesn't have a default value |
| 445 | Object doesn't support this action |
| 446 | Object doesn't support named arguments |
| 447 | Object doesn't support current locale setting |
| 448 | Named argument not found |
| 449 | Argument not optional or invalid property assignment |
| 450 | Wrong number of arguments or invalid property assignment |
| 451 | Object not a collection |
| 452 | Invalid ordinal |
| 453 | Specified not found |
| 454 | Code resource not found |
| 455 | Code resource lock error |
| 457 | This key is already associated with an element of this collection |
| 458 | Variable uses a type not supported in Visual Basic |
| 459 | This component doesn't support the set of events |
| 460 | Invalid Clipboard format |
| 461 | Method or data member not found |
| 462 | The remote server machine does not exist or is unavailable |
| 463 | Class not registered on local machine |
| 480 | Can't create AutoRedraw image |
| 481 | Invalid picture |
| 482 | Printer error |
| 483 | Printer driver does not support specified property |
| 484 | Problem getting printer information from the system. Make sure the printer is set up correctly |
| 485 | Invalid picture type |
| 486 | Can't print form image to this type of printer |
| 520 | Can't empty Clipboard |
| 521 | Can't open Clipboard |
| 735 | Can't save file to TEMP directory |
| 744 | Search text not found |
| 746 | Replacements too long |
| 31001 | Out of memory |
| 31004 | No object |
| 31018 | Class is not set |
| 31027 | Unable to activate object |
| 31032 | Unable to create embedded object |
| 31036 | Error saving to file |
| 31037 | Error loading from file |

Invalid procedure call or argument (Error 5)

Some part of the call can't be completed. This error has the following causes and solutions:

An argument probably exceeds the range of permitted values. For example, the Sin function can only accept values within a certain range.

Positive arguments less than 2,147,483,648 are accepted, while 2,147,483,648 generates this error.

Check the ranges permitted for arguments.

This error can also occur if an attempt is made to call a procedure that isn't valid on the current platform.

For example, some procedures may only be valid for Microsoft Windows, or for the Macintosh, and so on.

Check platform-specific information about the procedure.

For additional information, select the item in question and press F1.

Overflow (Error6)

An overflow results when you try to make an assignment that exceeds the limitations of the target of the assignment.

This error has the following causes and solutions:

The result of an assignment, calculation, or data type conversion is too large to be represented within the range of values allowed for that type of variable.

Assign the value to a variable of a type that can hold a larger range of values.

An assignment to a property exceeds the maximum value the property can accept.

Make sure your assignment fits the range for the property to which it is made.

You attempt to use a number in a calculation, and that number is coerced into an integer, but the result is larger than an integer. For example:

Dim x As Long

x = 2000 * 365 ' Error: Overflow

To work around this situation, type the number, like this:

Dim x As Long

x = CLng(2000) * 365

For additional information, select the item in question and press F1

Out of memory (Error 7)

More memory was required than is available, or a 64K segment boundary was encountered.

This error has the following causes and solutions:

You have too many applications, documents, or source files open.

Close any unnecessary applications, documents, or source files that are open.

You have a module or procedure that's too large.

Break large modules or procedures into smaller ones. This doesn't save memory, but it can prevent hitting 64K segment boundaries.

You are running Microsoft Windows in standard mode.

Restart Microsoft Windows in enhanced mode.

You are running Microsoft Windows in enhanced mode, but have run out of virtual memory.

Increase virtual memory by freeing some disk space, or at least ensure that some space is available.

You have terminate-and-stay-resident programs running.

Eliminate terminate-and-stay-resident programs.

You have many device drivers loaded.

Eliminate unnecessary device drivers.

You have run out of space for Public variables.

Reduce the number of Public variables.

Subscript out of range (Error 9)

Elements of arrays and members of collections can only be accessed within their defined ranges. This error has the following causes and solutions:

You referenced a nonexistent array element.

The subscript may be larger or smaller than the range of possible subscripts, or the array may not have dimensions assigned at this point in the application.

Check the declaration of the array to verify its upper and lower bounds.

Use the UBound and LBound functions to condition array accesses if you're working with arrays that are redimensioned.

If the index is specified as a variable, check the spelling of the variable name.

You declared an array but didn't specify the number of elements. For example, the following code causes this error:

```
Dim MyArray() As Integer
```

```
MyArray(8) = 234 ' Causes Error 9.
```

Visual Basic doesn't implicitly dimension unspecified array ranges as 0 - 10. Instead, you must use Dim or ReDim to specify explicitly the number of elements in an array.

You referenced a nonexistent collection member.

Try using the For Each...Next construct instead of specifying index elements.

You used a shorthand form of subscript that implicitly specified an invalid element.

For example, when you use the ! operator with a collection, the ! implicitly specifies a key.

For example, object!keyname.

value is equivalent to object.item(keyname).value. In this case, an error is generated if keyname represents an invalid key in the collection.

To fix the error, use a valid key name or index for the collection.

Division by zero (Error 11)

Division by zero isn't possible. This error has the following cause and solution:

The value of an expression being used as a divisor is zero.

Check the spelling of variables in the expression. A misspelled variable name can implicitly create a numeric variable that is initialized to zero.

Check previous operations on variables in the expression, especially those passed into the procedure as arguments from other procedures.

Type mismatch (Error 13)

Visual Basic is able to convert and coerce many values to accomplish data type assignments that weren't possible in earlier versions. However, this error can still occur and has the following causes and solutions:

The variable or property isn't of the correct type. For example,

a variable that requires an integer value can't accept a string value unless the whole string can be recognized as an integer.

Try to make assignments only between compatible data types. For example, an Integer can always be assigned to a Long,

a Single can always be assigned to a Double, and any type (except a user-defined type) can be assigned to a Variant.

An object was passed to a procedure that is expecting a single property or value.

Pass the appropriate single property or call a method appropriate to the object.

A module or project name was used where an expression was expected, for example:

```
Debug.Print MyModule
```

Specify an expression that can be displayed.

You attempted to mix traditional Basic error handling with Variant values having the Error

subtype (10, vbError), for example:

```
Error CVErr(n)
```

To regenerate an error, you must map it to an intrinsic Visual Basic or a user-defined error, and then generate that error.

A CVErr value can't be converted to Date. For example:

```
MyVar = CDate(CVErr(9))
```

Use a Select Case statement or some similar construct to map the return of CVErr to such a value.

At run time, this error typically indicates that a Variant used in an expression has an incorrect subtype,

or a Variant containing an array appears in a Print # statement.

To print arrays, create a loop that displays each element individually.

Out of string space (Error 14)

Visual Basic permits you to use very large strings. However, the requirements of other programs and

the way you manipulate your strings may cause this error. This error has the following causes and solutions:

Expressions requiring that temporary strings be created for evaluation may cause this error. For example,

the following code causes an Out of string space error on some operating systems:

```
MyString = "Hello"
```

```
For Count = 1 To 100
```

```
    MyString = MyString & MyString
```

```
Next Count
```

Assign the string to a variable of another name.

Your system may have run out of memory, which prevented a string from being allocated.

Remove any unnecessary applications from memory to create more space.

Sub, Function, or Property not defined (Error 35)

A Sub, Function, or Property procedure must be defined to be called. This error has the following causes and solutions:

You misspelled the name of your procedure.

Check the spelling and correct it.

You tried to call a procedure from another project without explicitly adding a reference to that project in the References dialog box.

To add a reference

Display the References dialog box.

Find the name of the project containing the procedure you want to call. If the project name doesn't appear in the References dialog box, click the Browse button to search for it.

Click the check box to the left of the project name.

Click OK.

The specified procedure isn't visible to the calling procedure.

Procedures declared Private in one module can't be called from procedures outside the module.

If Option Private Module is in effect,

procedures in the module aren't available to other projects. Search to locate the procedure.

You declared a Windows dynamic-link library (DLL) routine, but the routine isn't in the specified library or code resource.

Check the ordinal (if you used one) or the name of the routine. Make sure your version of the

DLL is the correct one.

The routine may only exist in later versions of the DLL.

If the directory containing the wrong version precedes the directory containing the correct one in your path, the wrong DLL is accessed.

You gave the right DLL name, but it isn't the version that contains the specified function.

File not found (Error 53)

The file was not found where specified. This error has the following causes and solutions:

A statement, for example, Kill, Name, or Open, refers to a file that doesn't exist.

Check the spelling of the file name and the path specification.

An attempt has been made to call a procedure in a dynamic-link library (DLL) or Macintosh code resource,

but the library specified in the Lib clause of the Declare statement can't be found.

Check the spelling of the file name and the path specification.

In the development environment, this error occurs if you attempt to open a project or load a text file that doesn't exist.

Check the spelling of the project name or file name and the path specification.

File already exists (Error 58)

This error has the following causes and solutions:

This error occurs at run time when the new file name, for example, one specified in a Name statement, is identical to a file name that already exists.

Specify a new file name in the Name statement or delete the old file before specifying it in a Name statement.

You used the Save As command to save a currently loaded project, but the project name already exists.

Use a different project name if you don't want to replace the other project.

Disk full (Error 61)

This error has the following causes and solutions:

There isn't enough room on the disk for the completion of a Print #, Write #, or Close operation.

Move some files to another disk or delete some files.

There isn't enough room on the disk to create required files.

Move some files to another disk or delete some files.

Disk not ready (Error 71)

:This error has the following causes and solutions

There is no disk in the specified drive.

.Put a disk in the drive and retry the operation

The drive door of the specified drive is open.

Close the drive door and retry the operation.

For loop not initialized (Error 92)

For loop counters must be initialized. This error has the following cause and solution:

You jumped into the middle of a For...Next loop.

Remove the jump into the loop. Placing labels inside a For...Next loop isn't recommended.

Invalid pattern string (Error 93)

The pattern string specified in the Like operation of a search is invalid. This error has the

following cause and solution:

A common example of an invalid character list expression is [a-b , where the right bracket is missing.

Review the valid characters for list expressions.

Invalid use of Null (Error 94)

Null is a Variant subtype used to indicate that a data item contains no valid data. This error has the following cause and solution:

You are trying to obtain the value of a Variant variable or an expression that is Null. For example:

```
MyVar = Null
```

```
For Count = 1 To MyVar
```

```
...
```

```
Next Count
```

Make sure the variable contains a valid value.

The specified object can't be used as an owner form for Show() (Error 371)

You must use an appropriate object with the Show method.

Invalid property value (Error 380)

Most properties only accept values of a certain type, within a certain range. This error has the following cause and solution:

An inappropriate value has been assigned to a property.

See the property's Help topic to determine what types and range of values are appropriate for the property.

Property Set can't be executed at run time (Error 382)

It may not be possible to obtain a reference to a property at run time. This error has the following cause and solution:

You tried to get a reference to a property that's either read-only or write-only at run time. Since you can use a reference for both reading and writing, the property must provide run-time support for both operations for a reference to be obtained at run time.

Property Set can't be used with a read-only property (Error 383)

It may not be possible to obtain a reference to a property at run time. This error has the following cause and solution:

You tried to get a reference to a property that's read-only at run time. Since you can use a reference for both reading and writing, the property must provide run-time support for both operations for a reference to be obtained at run time. You can only use a Property Get with this property.

Property not found (Error 422)

Not all objects support the same set of properties. This error has the following cause and solution:

This object doesn't support the specified property. Check the spelling of the property name. Also, you may be trying to access something like a «text» property when the object actually supports a «caption» or some similarly named property. Check the object's documentation.

Property or method not found (Error 423)

The spelling of an object name must exactly match the definition in its object library. This error

You may have misspelled the name of the object. To see what properties and methods are defined for an object, display the Object Browser. Select the appropriate object library to view a list of available properties and methods.

Named argument not found (Error 448)

A named argument may not be used in a procedure invocation unless it appears in the procedure definition. This error has the following cause and solution:

You specified a named argument, but the procedure was not defined to accept an argument by that name.

Check the spelling of the argument name.

Code resource not found (Error 454)

This error can only occur on the Macintosh. This error has the following cause and solution:

A call was made to a procedure in a code resource, but the code resource could not be found.

Check to be sure the resource is available and properly referenced.

For additional information, select the item in question and press F1.

Out of memory (Error 31001)

Your system could not allocate or access enough memory or disk space for the specified operation.

فهرست منابع

۱- راهنمای MSDN

2- Microsoft Visual Basic 6 Programmer's Guide (Microsoft press)

