فصل پنجم

کاربرد فرم‌های آماده

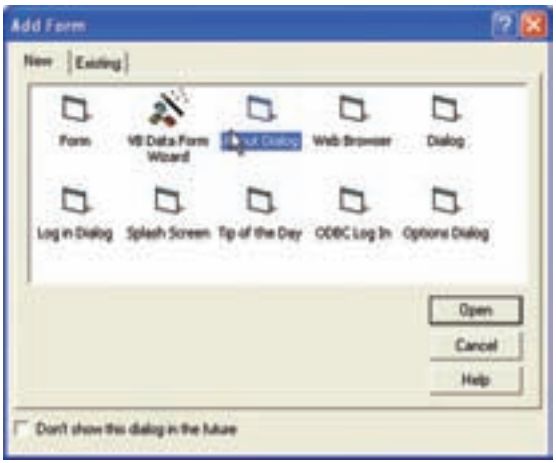
هدف‌های رفتاری: پس از آموزش این فصل، هنرجو می‌تواند:

- **دلیل استفاده از فرم‌های آماده را بیان کند؛**
- **از انواع مختلف فرم‌ها و کادرهای محاوره‌ای آماده مانند File، Color، Printer و غیره در برنامه‌های خود استفاده کند؛**
- **مفهوم رابط چند سندی را شرح دهد؛**
- **پروژه‌هایی ایجاد کند که از رابط چندسندی استفاده می‌کنند.**

۱ــ۵ــ فرم‌های آماده

ویژوال بیسیک دارای تعدادی فرم آماده (از پیش تعریف شده) است که استفاده از آن‌ها سبب صرفه‌جویی در زمان کدنویسی می‌شود. از فرم‌های آمادهٔ ویژوال بیسیک، می‌توان برای دسترسی به داده‌ها، کادر About، صفحه‌های الگو و کادرهای ورود، استفاده کرد.

از منوی Project گزینهٔ Add Form را انتخاب کنید تا کادر محاوره‌ای مربوطه باز شود. در این کادر محاوره‌ای، انواع مختلف فرم‌های آماده وجود دارد. فرم موردنظر را انتخاب کنید (شکل ۱ــ۵).



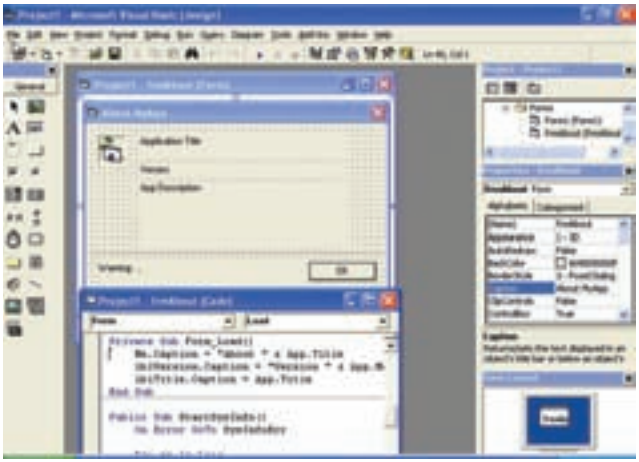
شکل ۱ــ۵ــ می‌توان فرم‌های آماده یا Form Wizard را در کادر محاوره‌ای Add Form انتخاب کرد.

در صورتی که می‌خواهید کادر محاوره‌ای About را به پروژه اضافه کنید، در کادر محاوره‌ای Add Form روی About Dialog کلیک کنید (شکل ۲ــ۵ یک فرم About Dialog را نشان می‌دهد). نه‌تنها ویژوال بیسیک، فرم را به پروژه اضافه می‌کند، بلکه بخشی از کدهایی را که وظایف اصلی فرم را دربر می‌گیرند نیز درج می‌کند. About Dialog کدی را ارایه می‌کند که نام برنامهٔ کاربردی و اطلاعات نسخه را همان‌طور که در کد زیر مشاهده می‌کنید گزارش می‌کند :

```
Private sub Form_load()
Me.caption =“About” & App. Title
LbLVersion. caption = “Version” & App. Major &“.” & App. Minor&“.”
LbLTitle.Caption = App.Title
End Sub
```

در این کد کنترل APP[1] به پروژه جاری و کنترل Me به فرم جاری اشاره می‌کند.

کادر محاوره‌ای About کل کد و فراخوانی API[2] ویندوز را، که برای گزارش اطلاعات سیستم کاربر موردنیاز است، ارایه می‌کند. (در صورتی که می‌خواهید کد اطلاعات سیستم را مرور کنید، فرم About Dialog را به پروژهٔ جدیدی اضافه کنید و کد تحت دکمهٔ System Info را مشاهده کنید.)



شکل ۲ــ۵ــ کادر محاوره‌ای About که به همراه ویژوال بیسیک ارایه می‌شود کدی را که نام برنامه و اطلاعات نسخه و به همین ترتیب اطلاعات سیستم کاربر را گزارش می‌کند، ارایه می‌دهد.

۱ــ در فصل ۸ توضیح آن آمده است.

۲ــ در برنامه‌سازی ۳ با انواع توابع API آشنا خواهید شد.

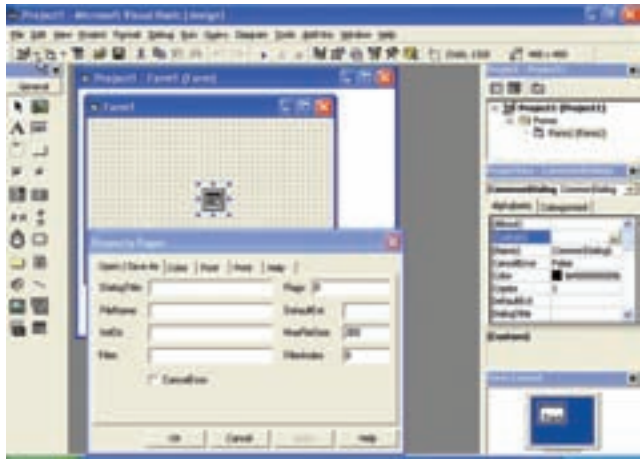
۲ــ۵ــ کنترل Common Dialog

بعضی مواقع ممکن است بخواهید برنامه‌ای بنویسید که کاربران بتوانند نام فایل را تعیین کنند، قلم‌ها و رنگ‌ها را انتخاب کنند و چاپگر را کنترل نمایند. اگرچه می‌توانید کادرهایی محاوره‌ای ایجاد کنید که این وظایف را مدیریت کنند ولی نیاز به انجام این کار نیست. ویژوال بیسیک، کنترل Common Dialog را ارایه می‌کند که به سادگی می‌توان به کمک آن کادرهای محاوره‌ای آماده را برای به‌دست آوردن اطلاعات کاربر، نمایش داد. این کادرهای محاوره‌ای برای کاربران آشنا هستند و همان‌هایی هستند که خود ویندوز به‌کار می‌برد. با استفاده از کنترل Common Dialog، به چهار کادر محاوره‌ای ویندوز دسترسی خواهید داشت :

- **File (Save, Open)**، به کاربر امکان انتخاب فایل برای باز شدن یا انتخاب نام فایل برای ذخیرهٔ اطلاعات را فراهم می‌کند.
- **Font**، امکان انتخاب قلم و تنظیم مشخصه‌های قلم موردنظر را فراهم می‌کند.
- **Color**، امکان انتخاب رنگ یا ایجاد رنگ سفارشی برای استفاده در برنامه را ارایه می‌کند.
- **Print**، امکان انتخاب چاپگر و تنظیم چندین پارامتر چاپگر را فراهم می‌کند.

برای استفاده ازکنترل Common Dialog، ابتدا باید آن را با انتخاب (MicrosoftCommon Dialog Control 6.0) از کادر محاوره‌ایComponents (انتخاب Components از منوی Project) به پروژه اضافه کنید. بعد از اضافه کردن کنترل Common Dialog به جعبه ابزار، روی کنترل کلیک و شبیه کنترل‌های دیگر، روی فرم ترسیم کنید. این کنترل، مانند یک نشانه روی فرم ظاهر می‌شود و در زمان اجرا، قابل رؤیت نیست. ولی هنگامی که کد، Common Dialog را فراخوانی کند، کادر محاوره‌ای خاصی ظاهر می‌شود.

در ادامه، هرکدام از کادرهای محاوره‌ای مربوط به این کنترل را شرح می‌دهیم. برای هر کادر محاوره‌ای، باید مشخصه‌های کنترل را از طریق پنجرهٔ Properties یا کادرمحاوره‌ای Property Pages تنظیم کنید. کادر محاوره‌ای Property Pages دسترسی ساده به مشخصه‌های موردنیاز برای هرنوع کادر محاوره‌ای را ارایه می‌کند (شکل ۳ــ۵). برای دسترسی به این کادر محاوره‌ای، در پنجرهٔ Properties روی سه نقطهٔ مقابل مشخصهٔ Custom مربوط به کنترل Common Dialog کلیک کنید.



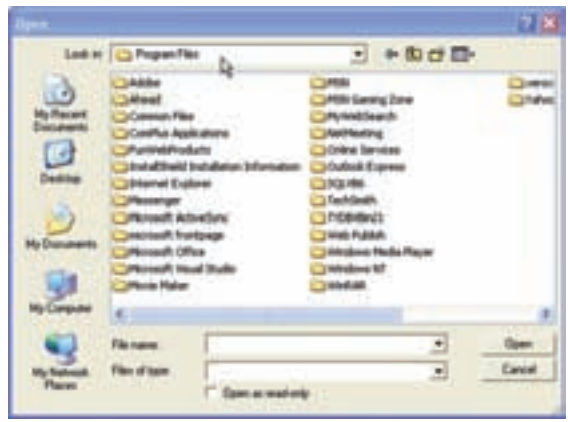
شکل ۳ــ۵ــ کنترل Common Dialog در زمان طراحی قابل رؤیت بوده و می‌تواند با استفاده از Property Pages پیکربندی شود.

۳ــ۵ــ بازیابی و ذخیرهٔ اطلاعات پرونده

کاربرد اصلی کنترل Common Dialog فراهم کردن امکان انتخاب نام پرونده به دو منظور بازکردن و ذخیره‌کردن پرونده است. حالت باز کردن پرونده، به کاربر امکان تعیین نام پرونده برای بازیابی و استفاده در برنامه را می‌دهد. حالت ذخیرهٔ پرونده به کاربر امکان می‌دهد تا نامی را برای ذخیرهٔ اطلاعات تحت یک پرونده، فراهم کند. شکل ۴ــ۵ کادر محاوره‌ای با مؤلفه‌های اصلی را نشان می‌دهد.

کادرهای محاوره‌ای Open و Save شبیه هم هستند. برای باز کردن یک پرونده موجود، از متد ShowOpen کنترل Common Dialog استفاده کنید. (این متد، کادر محاوره‌ای نشان داده شده در شکل ۴ــ۵ را نشان می‌دهد.) از این متد، به صورت زیر استفاده کنید :

```
CommonDlg1.ShowOpen
```



شکل ۴ــ۵ــ Common Dialog به کادرهای محاوره‌ای پرونده‌ای ویندوز دسترسی دارد.

برای بازکردن کادر محاوره‌ای Save As باید از متد ShowSave استفاده کرد.

برای محدود کردن نوع پرونده‌ها مثل پرونده‌های متنی یا سند در کادرهای محاوره‌ای پرونده‌ای، از مشخصهٔ Filter کنترل Common Dialog استفاده کنید. این مشخصه را می‌توان در زمان طراحی از طریق کادر محاوره‌ای Property Pages یا در زمان اجرا با دستور زیر تنظیم کرد :

ControlName.Filter = "description|filtercond"

- ControlName نام تعیین شده برای کنترل Common Dialog است.
- Filter نام مشخصه است.

نکته

دقت کنید که قبل یا بعد از علامت پایپ (|)، فضای خالی قرار ندهید. درصورتی که این کار را انجام دهید، ممکن است لیست فایل مورد نظر را به‌دست نیاورید.

- description شرحی از انواع پرونده‌هایی است که باید نشان داده شوند. مثال‌هایی از این شرح، عبارت‌اند از :"Text Files"، "Word Documents" و "All Files"، خط عمودی (علامت پایپ) باید وجود داشته باشد.
- Filtercond فیلتر واقعی پرونده‌هاست. این فیلتر را به صورت‌های *.txt، *.doc و *.* به‌کار ببرید.

اگر از دستور انتساب فوق استفاده کنید، باید فیلتر را داخل زوج کوتیشن (" ") قرار دهید ولی در صورت استفاده از کادر محاوره‌ایProperty Pages نیازی به این کار نیست.

می‌توان چندین فیلتر را با هم به‌کار برد و بین این فیلترهای مختلف، باید ازعلامت پایپ استفاده کرد. مثال زیر را درنظر بگیرید :

Commondlg1. Filter ="Text Documents | *.txt | All Files (*.*) | *.*"

کادر ترکیبی File of Type در شکل ۴ــ۵ کد فوق را که به کنترل Common Dialog اعمال شده است نشان می‌دهد. بعد از تنظیم فیلتر، از مشخصهٔ Filename کنترل Common Dialog برای بازیابی نام فایلی که کاربر انتخاب کرده است، استفاده کنید :

MyFileName$ = Commond1g1.FileName

مثال ۱ــ۵

می‌خواهیم از کادر محاوره‌ای File برای انتخاب و باز کردن یک پرونده استفاده کنیم. مراحل کار به صورت زیر خواهد بود :

۱ــ پروژهٔ جدیدی را شروع کنید.

۲ــ کنترل Common Dialog را همان‌طور که قبلاًشرح داده شد، به جعبه ابزار اضافه کنید.

۳ــ روی فرم کنترل‌های label، Command Button و Common Dailog اضافه کنید.

۴ــ کنترل برچسب را در بالای کنترل دکمهٔ فرمان قرار دهید. هر دو کنترل را به گوشهٔ چپ بالای فرم، منتقل کنید.

۵ ــ روی دکمهٔ فرمان، دوبار کلیک کنید تا روال رویداد Commandl_Click ایجاد شود.

۶ــ کد زیر را به این روال اضافه کنید :

```
01 CommonDialogl.Filter ="All Files (*.*) | *.*"
02 CommonDialogl.ShowOpen
03 If CommonDialogl. FileName<>"" Then
04    LabeLl.Caption = CommonDialogl.FileName
05 Else
06    LabeL1. Caption = "No file selected"
07 End If
```

عملکرد این کد به صورت زیر است :

خط ۱، فیلتری را برای نمایش تمام پرونده‌ها در یک پوشه، اعمال می‌کند.

خط ۲، کادر محاوره‌ای Open را نمایش می‌دهد. کاربران می‌توانند پرونده‌ای را انتخاب کرده و رویOpen کلیک کنند یا هیچ پرونده‌ای را انتخاب نکرده و روی Cancel کلیک کنند. براساس این انتخاب‌ها، ویژوال بیسیک مقداری را در مشخصهٔ FileName کنترل Common Dialog قرار می‌دهد.

خطوط ۳ تا ۷، مقدار مشخصهٔ FileName را بررسی می‌کنند و در صورت انتخاب پرونده، نام آن در مشخصهٔ Caption برچسب قرار می‌گیرد و در غیر این‌صورت، پیغام No File Selected نوشته می‌شود.

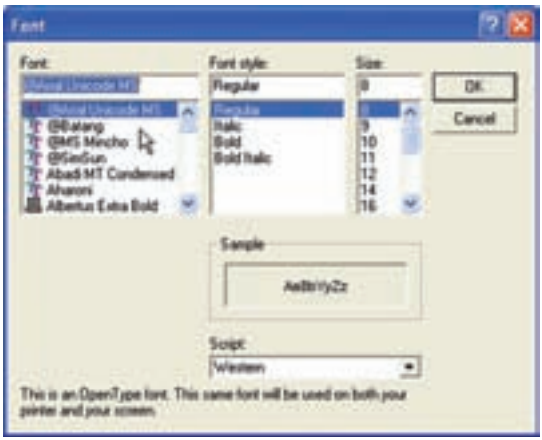
۴ــ۵ــ انتخاب اطلاعات قلم

تنظیم کنترل Common Dialog برای نمایش کادر محاوره‌ای Font شبیه نمایش کادرهای محاوره‌ای پرونده‌ای است.

اولین گام در استفاده از کنترل Common Dialog برای مدیریت قلم‌ها، مقداردهی مشخصهٔ Flags است. این مشخصه به کنترل Common Dialog بیان می‌کند که آیا می‌خواهید قلم‌های صفحهٔ نمایش، قلم‌های چاپگر و یا هر دو را نمایش دهید. مشخصهٔ Flags می‌تواند یکی از سه ثابت زیر را داشته باشد :

جدول ۱ــ۵ــ مقادیر مشخصه Flags در کادر محاوره‌ای Font

مجموعهٔ قلم	ثابت	مقدار
قلم‌های صفحهٔ نمایش	cdlCFScreenFonts	1
قلم‌های چاپگر	cdlCFPrinterFonts	2
هر دو مجموعه	cdCFBoth	3



شکل ۵ــ۵ــ کادر محاوره‌ای Font امکان انتخاب قلم‌ها را به کاربر می‌دهد.

نکته

اگر از کنترل Common Dialog برای انتخاب قلم‌ها استفاده می‌کنید و مقداری را برای مشخصهٔ Flags تنظیم نکرده‌اید، پیغام خطایی را دریافت خواهید کرد مبنی بر این که هیچ قلمی نصب نشده است.

می‌توان مقدار مشخصهٔ Flags را از محیط طراحی با استفاده از کادر محاوره‌ای Property Pages یا از درون برنامه با دستور انتساب تعیین کرد. بعد از تعیین مشخصهٔ Flags، می‌توان کادر محاوره‌ای Font را با متد ShowFont اجرا کرد که دارای شکل کلی مانند متد Show Open است که قبلاً شرح داده شده است.

اطلاعات قلم‌های انتخاب‌شده در مشخصه‌های کنترل قرار داده می‌شود. جدول ۲ــ۵ مشخصه‌های کنترل و صفات قلم را نشان می‌دهد.

جدول ۲ــ۵ــ مشخصه‌های کنترل که صفات قلم را ذخیره می‌کنند

مشخصه	صفت
FontName	نام قلم
FontSize	اندازهٔ قلم برحسب پوینت
FontBold	آیا ضخیم‌بودن قلم انتخاب شده است؟
FontItalic	آیا کج‌بودن قلم انتخاب شده است؟
FontUnderline	آیا قلم زیرخط‌دار شده است؟
FontStrikethru	آیا روی قلم خط کشیده شده است؟

اطلاعات قلم می‌تواند برای تعیین قلم هر نوع کنترلی در برنامه یا حتی شیء Printer مورد استفاده قرار گیرد. کد زیر نشان می‌دهد که اطلاعات قلم چگونه به‌دست آمده و برای تغییر قلم‌ها در کادر متن txtsample استفاده می‌شود.

```
Commondlg1 is the name of a common dialog
Commondlg1.ShowFont
txtSample.FontName = Commondlgl.FontName
txtSample.FontSize = Commondlgl.FontSize
txtSample.FontBold = Commondlgl.FontBold
txtSample.FontItalic = Commondlgl.FontItalic
txtSample.FontUnderline = Commondlg1.FontUnderline
txtSample.FontStrikethru = Commondlgl.FontStrikethru
```

تمرین ۱ــ۵: برنامه‌ای بنویسید که دارای کادر متن و دکمهٔ انتخاب فونت باشد و کاربر بتواند نوع و اندازه و Style فونت کادر متن را تعیین کند.

۵ــ۵ــ انتخاب رنگ‌ها

کادر محاوره‌ای Color امکان انتخاب رنگ‌هایی برای رویه یا زمینهٔ فرم‌ها یا کنترل‌ها فراهم می‌کند (شکل۶ــ۵). کاربران می‌توانند یک رنگ استاندارد را انتخاب کرده یا رنگ سفارشی را ایجاد و انتخاب کنند.

شکل ۶ــ۵ــ کادر محاوره‌ای Color امکان انتخاب رنگی برای استفاده در برنامه را به کاربران می‌دهد.

برای فعال کردن کادر محاوره‌ای Color در کنترل Common Dialog، مشخصهٔ Flags آن را با ثابت cdLccRGBInit مقداردهی و سپس متد ShowColor را فراخوانی کنید.

هنگامی که کاربران رنگی را از کادر محاوره‌ای انتخاب می‌کنند، مقدار رنگ در مشخصهٔ Color کنترل، ذخیره می‌شود.

کد زیر، چگونگی تغییر رنگ زمینهٔ فرم را نشان می‌دهد :

```
CommonDlg1.Flags = cdLccRGBInit
CommonDlg1.ShowColor
```

```
MyForm.BackColor = CommonDlg1.Color
```

تمرین ۲ــ۵: به مثال ۱ـ۵ دکمه‌ای برای انتخاب رنگ متن در TextBox اضافه کنید.

۶ــ۵ــ تنظیم گزینه‌های چاپگر

کادر محاوره‌ای Print به کاربران امکان انتخاب چاپگر و تعیین گزینه‌ها برای فرآیند چاپ را ارایه می‌کند. این گزینه‌ها شامل تعیین کل صفحات، محدوده‌ای از صفحات یا بخش انتخاب شده برای چاپ است. همچنین گزینه‌هایی برای تعیین تعداد کپی‌های چاپی و چاپ در یک پرونده نیز وجود دارد.

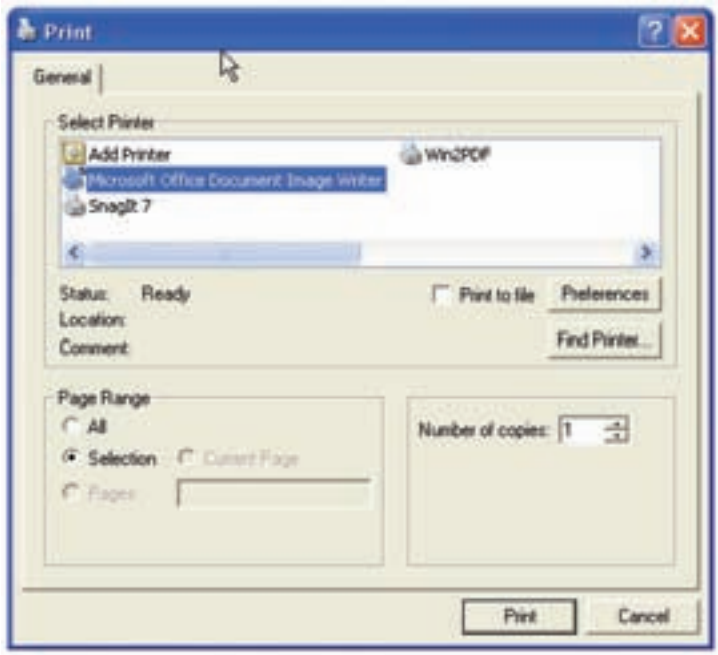
برای اجرای کادر محاوره‌ای Print، باید متد ShowPrinter را اجرا کرد (شکل ۷ــ۵). کاربران می‌توانند در این کادر محاوره‌ای چاپگر را از لیست Name انتخاب کنند که شامل تمام چاپگرهای نصب شده در ویندوز است. زیر لیست Name خط Status قرار دارد که وضعیت فعلی چاپگر انتخاب شده را بیان می‌کند.

کادر محاوره‌ای Print اطلاعات دریافتی از کاربران را در مشخصه‌های کادر محاوره‌ای برمی‌گرداند. مشخصه‌های FromPage و To Page، صفحات شروع و پایان خروجی چاپی را بیان می‌کنند. مشخصهٔ Copies تعداد کپی‌هایی را که کاربران می‌خواهند چاپ کنند نشان می‌دهد.

با استفاده از کادر محاوره‌ای Print تنها می‌توان چاپگر و نحوهٔ چاپ را انتخاب کرد ولی داده‌ها برای چاپ به چاپگر ارسال نمی‌شود برای ارسال داده‌ها به چاپگر باید از شیء Printer[1] استفاده کرد.

برنامه‌های کاربردی MDI دارای ظاهر یکسان و هماهنگ بوده و کار کردن با آن‌ها نسبت به برنامه‌های SDI (تک سندی) ساده‌تر خواهد بود. در برنامه‌های SDI تمام پنجره‌ها مستقل از همدیگرند. اغلب برنامه‌نویسان ترجیح می‌دهند که از برنامه‌های کاربردی MDI استفاده کنند. برنامه‌های NotePad و Pbrush نمونه‌هایی از برنامه‌های تک‌سندی هستند.

۱ــ شیء Printer و متدهای آن در برنامه‌سازی ۳ شرح داده شده است.



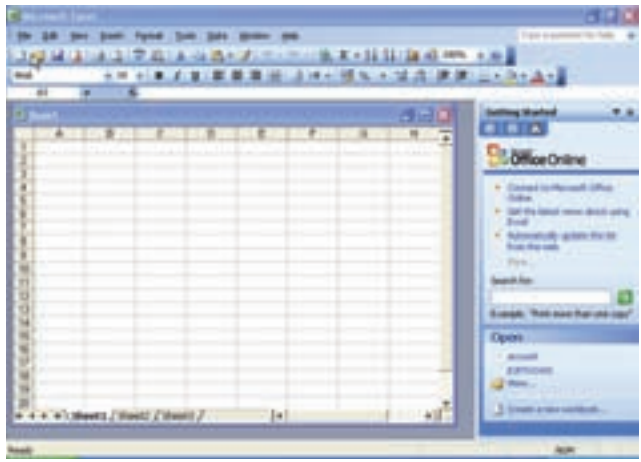
شکل ۷ــ۵ــ کادر محاوره‌ای **print** امکان انتخاب چاپگر و تنظیمات آن را به کاربر می‌دهد.

کد زیر چگونگی استفاده از کادر محاوره‌ای print را نشان می‌دهد. در این کد تنظیمات انجام شده توسط کاربر روی برچسب LblPrint نمایش داده می‌شود.

```
Dim BeginPage, EndPage, NumCopies, I, strs
CommonDlg1. ShowPrinter 'Display Print dialog box
'Get user-selected values
BeginPage = CommonDlg1. FromPage
EndPage = CommonDlg1. Topage
NumCopies = CommonDlg1. Copies
'display user-selected values
strs = "begin page: "& BeginPage
strs = strs & "end page: " & EndPage
strs = strs & "num copies: " & NumCopies
LblPrint. Caption = strs
```

۷ــ۵ــ ایجاد برنامهٔ کاربردی چندسندی ساده

MDI سرنام کلمات Multiple Document Interface (رابط چند سندی) است. یک برنامهٔ MDI (چندسندی)، برنامه‌ای است که می‌تواند هم‌زمان چندین سند باز داشته باشد و روی آن‌ها کار کند. Microsoft Word و Microsoft Excel مثال‌هایی از برنامهٔ کاربردی چندسندی هستند.



شکل ۸ــ۵ــ Microsoft Excel مثالی از یک برنامهٔ کاربردی MDI است.

ویژوال بیسیک یک شیء MDI به نام MDIForm تعریف کرده است. یک برنامهٔ کاربردی در VB می‌تواند فقط شامل یک شیء MDIForm باشد ولی می‌تواند چندین فرم دیگر داشته باشد که بعضی از آن‌ها فرزند شیء MDIForm بوده و بعضی دیگر پنجره‌های مستقل هستند. پنجره‌های فرزند یک MDIForm منوی خاص خودشان را نمی‌توانند داشته باشند و به جای آن، پنجره‌های فرزند به‌وسیلهٔ منوی فرم MDI پدرشان کنترل می‌شوند. اگر منویی را به فرم فرزند MDI اضافه کنید، در زمان اجرا داخل فرم فرزند MDI قابل رؤیت نخواهد بود. منوی فرزند فعال در پنجرهٔ پدر در محلی از منوی پدر، ظاهر می‌شود.

نکته

کنترل‌های محدودی مانند Common Dialog، Picture Box و Timer می‌توانند به‌طور مستقیم روی فرم MDI قرار بگیرند. بنابراین، برای قرار دادن کنترل‌های دیگر روی فرم MDI باید یک کنترل Picture Box روی فرم قرار داد و سپس کنترل‌های موردنظر را داخل آن استفاده کرد.

پژوهش: بررسی کنید که آیا کنترل‌های دیگری وجود دارند که بتوان به‌طور مستقیم روی فرم MDI قرار داد؟

مثال ۲ــ۵

۱ــ پروژهٔ جدیدی را شروع کنید و نام آن را SimplMDI.VBP قرار دهید.

۲ــ فرم پیش‌فرض پروژه را به frmChild تغییر نام دهید. پرونده فرم را با همین نام ذخیره کنید.

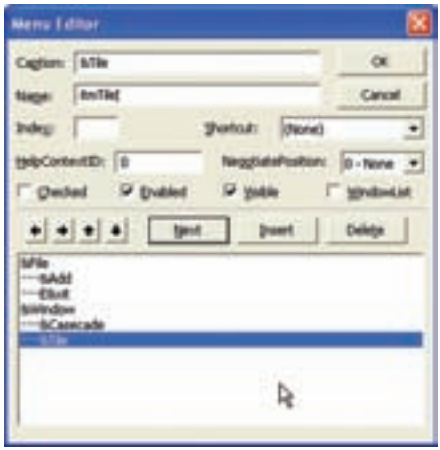
۳ــ ازمنوی Project گزینهٔ Add MDI Form را انتخاب کنید تا یک فرم MDI به پروژه اضافه شود. فرم MDI را به mdiMain تغییر نام دهید و با همین نام ذخیره کنید.

۴ــ مشخصهٔ MDIchild فرم frmChild را با true مقداردهی کنید تا به‌عنوان فرم فرزند mdiMain محسوب شود.

۵ــ کلیدهای Ctrl+E را فشار دهید تا Menu Editor باز شود.

۶ــ مانند شکل ۹ــ۵ منویی را برای فرم mdiMain ایجاد کنید. مقادیر مشخصهٔ Name و Caption منو و گزینه‌های منو را مطابق جدول ۳ــ۵ تعیین کنید. مطمئن شوید که کادر علامت WindowList برای mnuWindow انتخاب شده باشد.

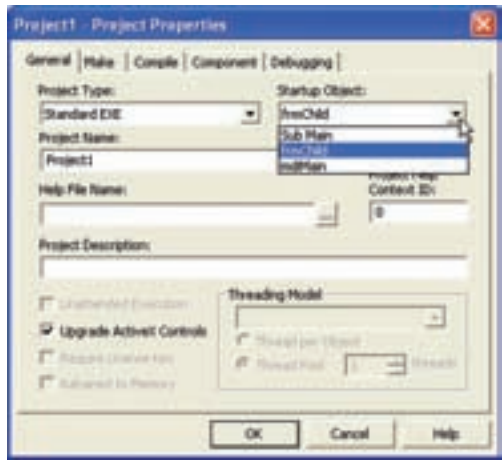
۷ــ از منوی Project گزینهٔ SimplMDI Properties را انتخاب کنید. در صفحهٔ General کادر محاوره‌ای Project Properties، از لیست بازشوی Startup Object گزینهٔ mdiMain را انتخاب کنید (شکل ۱۰ــ۵).



شکل ۹ ــ۵ــ منوهایی را با استفاده از Menu Editor برای یک برنامهٔ کاربردی MDI ایجاد کنید.

جدول ۳ــ۵ــ مشخصات منوی SimplMDI

Cation	Name	Indent
&File	mnuFile	0
&Add	itmAdd	1
&Exit	itmExit	1
&Window	mnuWindow	0
&CasCade	itmCasCade	1
&Tile	itmTile	1



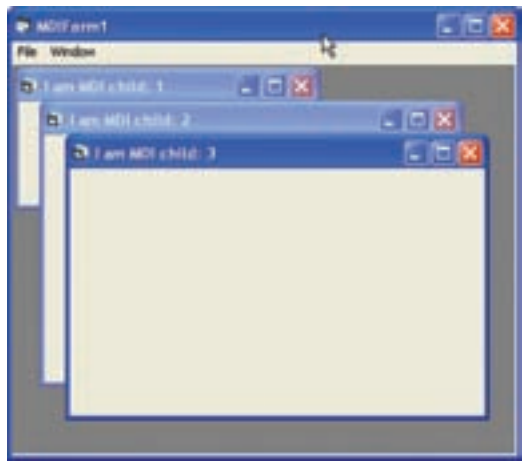
شکل ۱۰ــ۵ــ فرم MDI را به‌عنوان آغازین، تعیین کنید.

۸ ــ کد زیر را برای گزینه‌های منوی برنامهٔ کاربردی بنویسید :

```
Private Sub itmExit–Click()
    End
End Sub
Private Sub itmAdd_Click()
Dim NewForm As New frmChild
Dim FormNum As Integer
```

ایجاد فرم جدید به‌صورت پویا :

```
Load NewForm
FormNum = Forms.Count-1                    : Child تعیین تعداد فرم‌های
NewForm.Caption = "I am MDI child: " + CStr(FormNum)
End Sub
Private Sub itmCasCade_Click()
    mdiMain. Arrange  vbCasCade
End Sub
Private Sub itmTile_Clike()
mdiMain. Arrange vbTile Horizontal
EndSub
```



شکل ۱۱ــ۵ــ پروژهٔ SimplMDI فرم‌های فرزند داخل فرم MDI را نشان می‌دهد.

۹ــ برنامه را اجرا کنید.

روال رویداد itmAdd_Click() به‌طور پویا فرم جدیدی را با استفاده از عملگر New ایجاد می‌کند. Caption فرم جدید دارای یک شماره است (FormNum) که به انتهای رشتهٔ مربوطه اضافه می‌شود تا نشان دهد که چندمین فرم فرزند است. این شماره ازمشخصهٔ Forms.Count به‌دست می‌آید و برای این که تعداد فرم‌های فرزند محاسبه شوند، فرم MDI باید از شمارش، کسر شود. به همین دلیل، شمارهٔ فرم از تفریق تعداد فرم‌ها و عدد ۱ به‌دست می‌آید.

در این برنامهٔ کاربردی، از متد Arrange شیء MDIForm استفاده شده است. این متد، به‌طور خودکار پنجره‌های فرزند را داخل پنجرهٔ MDI قرار می‌دهد. متد Arrange دارای یک آرگومان است که می‌تواند یکی از مقادیر جدول ۴ــ۵ را داشته باشد.

جدول ۴ــ۵ــ تنظیمات آرایش پنجره‌ها

ثابت	مقدار	شرح
vbCasCade	0	تمام فرم‌های فرزند MDI را که به حداقل تبدیل نشده‌اند پشت سرهم قرار می‌دهد.
vbTileHorizontal	1	تمام فرم‌های فرزند MDI را که به حداقل تبدیل نشده‌اند به‌صورت کاشی‌های افقی می‌چیند.
vbTileVertical	2	تمام فرم‌های فرزند MDI را که به حداقل تبدیل نشده‌اند به صورت کاشی‌های عمودی می‌چیند.
vbArrangeIcons	3	آیکن‌های مربوط به MDI حداقل شده را مرتب می‌کند.

۸ــ۵ــ مشخصه‌های Appearance

فرم‌های سه‌بعدی برای کاربران مناسب‌تر است. با تغییر مشخصهٔ Appearance فرم MDI به 1، می‌توان فرم‌های سه‌بعدی ایجاد کرد. اگر مقدار این مشخصه 0 باشد فرم به‌صورت تخت ظاهر می‌شود (0–Flat، 1–3D)

مشخصهٔ AutoShowChildren : اگر مشخصهٔ AutoShowChildren دارای مقدار false باشد و فراموش کنید که در کد از متد Show برای نمایش فرم بعد از بارگذاری آن استفاده کنید، کاربران برای نمایش فرم، دچار مشکل خواهند شد.

به‌طور پیش‌فرض AutoShowChildren دارای مقدار True است و به همین دلیل هنگام استفاده از دستور Load فرم فرزند MDI قابل رؤیت خواهد بود. انجام این کار سبب صرفه‌جویی در زمان برنامه‌نویسی شده و توانایی کد برنامه را افزایش می‌دهد.

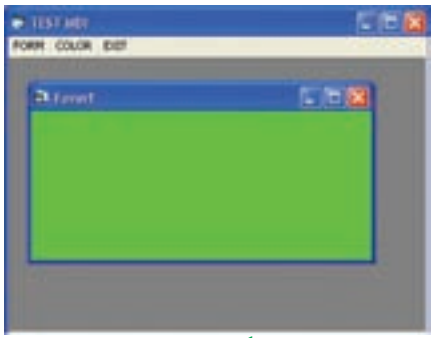
نکته

اگر فرمی دارید که داده‌ها را در درون خودش بارگذاری می‌کند، مشخصهٔ

AutoShowChildren را با False مقداردهی کنید و از متد Show استفاده کنید. نخست داده‌ها را در فرم بارگذاری کنید و مطمئن شوید که همهٔ داده‌ها به درستی بارگذاری شده‌اند. در پایان، فرم را نشان دهید. اگر فرم به تجمع داده‌ها وابستگی دارد، این عمل بالاترین قابلیت را ارایه می‌کند.

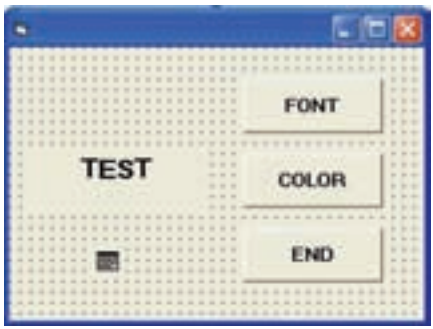
خودآزمایی

۱ــ فرمی مانند شکل روبه‌رو طراحی کنید، طوری که با انتخاب هر فرم از منوی Form ، فرم مربوطه باز شود و با انتخاب هر رنگی از منوی Color فرم جاری تغییر رنگ یابد و منوی Exit نیز سبب خروج از برنامه شود.



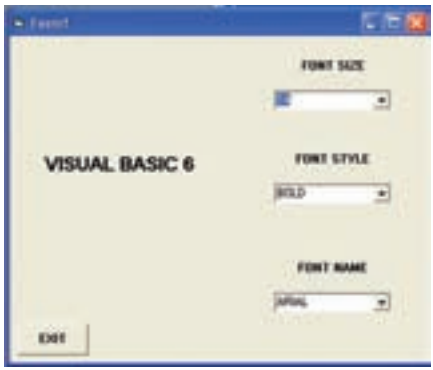
شکل ۱۲ــ۵

۲ــ فرمی مانند شکل روبه‌رو طراحی کنید به نحوی که با کلیک روی دکمهٔ Font بتوان خصوصیات ظاهری قلم برچسب، و با انتخاب Color بتوان رنگ برچسب را از طریق کادرهای محاوره‌ای مربوطه تغییر داد.



شکل ۱۳ــ۵

۳ــ فرمی مانند شکل روبه‌رو طراحی کنید به نحوی که با انتخاب اندازه و حالت و نام قلم، نوشتهٔ روی برچسب تغییر کند.



شکل ۱۴ــ۵

فصل ششم

زمان و تاریخ

هدف‌های رفتاری: پس از آموزش این فصل، هنرجو می‌تواند:

- **در برنامه‌های خود از عنصر زمان استفاده کند؛**
- **برنامه‌هایی بنویسد که تاریخ و ساعت را به‌طور کامل مدیریت کنند؛**
- **انواع قالب‌ها را توضیح دهد و به‌کار گیرد.**

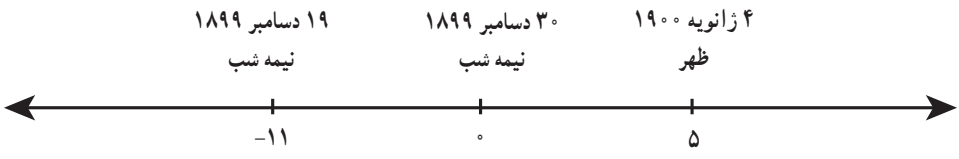
۱ــ۶ــ آشنایی با Serial Time

ویژوال بیسیک از نوع دادهٔ Date استفاده می‌کند که روز را به‌عنوان واحد اصلی زمان به‌کار می‌برد. بنابراین، یک ساعت، $\frac{۱}{۲۴}$ و یک ثانیه $\frac{۱}{۸۶۴۰۰}$ روز است. یک هفته را به‌صورت ۷ نمایش می‌دهیم، زیرا یک هفته شامل هفت روز است. نوع دادهٔ Date تاریخ را مطابق با قالب زمان کامپیوتر، نمایش می‌دهد؛ یعنی براساس قالب ۱۲ ساعتی یا ۲۴ ساعتی.

در تقویم میلادی اولین روز، اول ژانویهٔ سال است ولی ویژوال بیسیک، اولین روز را ۳۱ دسامبر[۱] سال ۱۸۹۹ فرض می‌کند. بنابراین، دومین روز، اول ژانویه سال ۱۹۰۰ است و ۱۲ ژوئن سال ۱۹۶۸، روز ۲۵۰۰۱ است.

تاریخ‌های قبل از ۳۱ دسامبر ۱۸۹۹، با اعداد منفی نمایش داده می‌شوند. به‌عنوان مثال، ۴ جولای سال ۱۷۷۶، روز ۴۵۱۰۳- است یعنی ۴۵۱۰۳ روز قبل از ۳۰ دسامبر ۱۸۹۹.

به محور زیر توجه کنید :



۱ــ ماه قبل از ژانویه

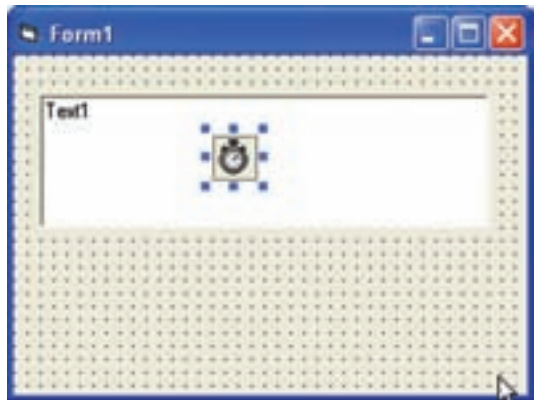
می‌توان مقادیر تاریخ را با قرار دادن بین دو علامت عددی (#)، در متغیرهای Date ذخیره کرد. به‌عنوان مثال :

```
Dim MyDate As Date
MyDate = #May 15, 1998#
```

نوع دادهٔ Date می‌تواند به هر نوع دادهٔ دیگری تبدیل شود. به عنوان مثال، P.M May 15,1998 2 به صورت یک عدد اعشاری 35390.58333 از نوع Double خواهد بود. هر چیزی که در سمت چپ نقطهٔ اعشار است، نمایانگر روز بوده و هر چیزی که در سمت راست نقطهٔ اعشار باشد، نمایانگر زمان یا بخشی از روز است : ساعت، دقیقه، ثانیه و میلی‌ثانیه. توجه داشته باشید که مقادیر اعشاری کوچک‌تر از ۱ می‌توانند فقط برای تعیین زمان به‌کار روند. به‌عنوان مثال، 0.12345 زمان 2:57:46 A.m را نشان می‌دهد[۱].

۲ــ۶ــ آشنایی با کنترل Timer

در ویژوال بیسیک، کنترل Timer امکان پی‌گیری زمان را فراهم می‌کند. فرض کنید که Timer ساعتی است که یک رویداد قابل برنامه نویسی را در یک فاصلهٔ زمانی که تعیین کرده‌اید اجرا می‌کند (شکل ۱ــ۶). رویداد Timer فراخوانی شده و روال رویدادی که برای TimerName_Timer() برنامه‌نویسی کرده‌اید اجرا می‌شود. Timer Name مقدار مشخصهٔ Name کنترل Timer است.



شکل ۱ــ۶ــ کنترل Timer در زمان اجرا قابل رؤیت نخواهد بود.

۱ــ هر روز ۲۴ ساعت و هر ساعت ۶۰ دقیقه و هر دقیقه ۶۰ ثانیه است.

۰/۱۲۳۴۵ × ۲۴ = ۲/۹۶۲۸

۰/۹۶۲۸ × ۶۰ = ۵۷/۷۶۸ ⇒ ۲: ۵۷: ۴۶ Am

۰/۷۶۸ × ۶۰ = ۴۶/۰۸

می‌توان فاصلهٔ زمانی را که طی آن رویداد Timer اجرا می‌شود با تعیین مقداری برای مشخصهٔ Interval کنترل Timer تنظیم کرد. واحد اندازه‌گیری مشخصهٔ Interval میلی‌ثانیه است، بنابراین اگر می‌خواهید رویداد Timer در نیم ثانیه اجرا شود، کد زیر را بنویسید :

```
Timerl.Interval = 500
```

نکته

حداکثر مقدار مشخصهٔ Interval، می‌تواند ۶۵,۵۳۵ باشد. این بدین معنی است که حداکثر فاصلهٔ زمانی که می‌توانید تنظیم کنید، ۶۵,۵ ثانیه است. برای استفاده از فاصله‌های زمانی طولانی‌تر مثل ۱۰ دقیقه، باید ۱۰ فاصلهٔ زمانی ۶۰۰۰۰میلی‌ثانیه را قرار دهید.

فاصلهٔ زمانی می‌تواند خیلی کوچک مثل هزارم ثانیه، و یا خیلی بزرگ باشد. کوتاه‌ترین فاصلهٔ زمانی، ۵۵ میلی‌ثانیه است، زیرا ساعت سیستم در هر ثانیه فقط ۱۸ بار پالس تولید می‌کند.

به دلیل این‌که کنترل Timer رویداد Timer را اجرا می‌کند، دارای مشخصه‌های زیادی نیست (جدول ۱ــ۶).

جدول ۱ــ۶ــ مشخصه‌های کنترل Timer

مشخصه	شرح
Name	نام کنترل. اگر یک کنترل Timer داشته باشید، پیش‌فرض Timerl است و الی آخر.
Enabled	کنترل Timer را فعال یا غیر فعال می‌کند. پیش‌فرض، True یا On است.
Index	محل کنترل Timer را هنگامی که به‌عنوان عنصری در آرایه‌ای از کنترل‌های Timerباشد تعیین می کند.
Interval	فاصلهٔ زمانی را که رویداد Timer اجرا خواهد شد تعیین می‌کند (برحسب میلی‌ثانیه).
Left	موقعیت گوشهٔ سمت چپ کنترل Timer را تعیین می‌کند.
Tag	شبیه یک متغیر درونی در کنترل است که داده‌های موردنیاز را ذخیره کرده و تا پایان برنامه به صورت پایدار نگه‌داری می‌کند.
Top	موقعیت گوشهٔ بالای کنترل Timer را تعیین می‌کند.

نکته

مشخصه‌های Left و Top کاربردی ندارند، زیرا کنترل Timer در زمان اجرا روی فرم نمایش داده نمی‌شوند.

مثال ۱ــ۶

فرمی طراحی کنید که دارای یک برچسب باشد و با اجرای برنامه این برچسب از سمت چپ به سمت راست فرم حرکت کند و با رسیدن به انتهای سمت راست ازحرکت بایستد.

برای این برنامه نیاز به یک Label و یک Timer داریم.

```
Private Sub Form_Load()
    Timer1.Interval = 300
    Label1.Caption = "Move"
End Sub
Private Sub Timer1_Timer()
    If Label1. Left<Form1.Width–Label1. Width Then
        Label1. Left = Label1. Left +100
    Else
        Timer1. Enabled = False   متوقف‌کردن حرکت برچسب در انتهای سمت راست
    End If
End Sub
```

تمرین: در مثال ۱ـ۶، دو دکمهٔ فرمان به نام Start و End قرار دهید که با کلیک روی Start حرکت شروع و با کلیک روی End خاتمه یابد.

۳ــ۶ــ کاربرد توابع Date، Time و Now

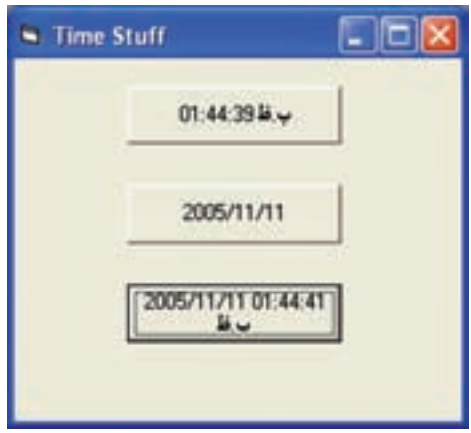
اگرچه می‌توان برای رویداد Timer برنامه‌ای نوشت، ولی نمی‌توان زمان اجرا را با این کنترل تنظیم کرد. برای این که Timer زمان را گزارش کند باید ساعت سیستم را بررسی کند.

از تابع Time برای به‌دست آوردن زمان سیستم و از تابع Date برای به‌دست آوردن تاریخ سیستم استفاده کنید. در صورتی که می‌خواهید هم‌زمان و هم تاریخ را از ساعت سیستم به‌دست آورید، از تابع Now استفاده کنید.

مثال ۲ــ۶

برای مشاهدهٔ چگونگی عملکرد این توابع، پروژه‌ای را ایجاد کرده و سه دکمهٔ فرمان به فرم اضافه کنید. این دکمه‌ها را به ترتیب cmdTime، cmdDate و cmdNow نام‌گذاری کنید و کد زیر را برای روال رویداد هر دکمه اضافه کنید. شکل ۲ــ۴ داده‌هایی را که هر تابع برمی‌گرداند نشان می‌دهد.

```
Private Sub cmdTime_Click()
        cmdTime.Caption = CStr(Time)
End Sub
Private Sub cmdDate_Click()
        cmdDate.Caption = CStr(Date)
End Sub
Private Sub cmdNow_Click()
        cmdNow.Caption = CStr(Now)
End Sub
```



شکل ۲ــ۶ــ توابع Time، Date و Now یک نوع دادهٔ Variant(Date) را برمی‌گردانند که با تابع CStr() به رشته تبدیل می‌شوند.

مثال ۳ــ۶

استفاده از Timer برای ایجاد یک برنامهٔ ساعت : این برنامه زمان اجرا را روی فرم و تاریخ جاری را در نوار عنوان فرم نشان می‌دهد. هنگامی که فرم به حداقل اندازه رسیده باشد، زمان اجرا در عنوان فرم در نوار ابزار ظاهر می‌شود. مراحل انجام کار به صورت زیر است :

۱ــ پروژهٔ Standard EXE جدیدی را شروع کنید.

۲ــ یک کنترل label و یک کنترل Timer روی فرم قرار دهید (کنترل Timer را می‌توانید در هر جایی قرار دهید، زیرا قابل رؤیت نخواهد بود.)

۳ــ نام فرم را frmClock و نام برچسب را lblTimeDisplay قرار دهید. نام کنترل Timer را تغییر ندهید (همان نام پیش‌فرض Timerl را حفظ کنید).

۴ــ مشخصهٔ BorderStyle فرم را به ۱–Fixed single تغییر دهید. مشخصهٔ MinButton فرم را True قرار دهید.

۵ــ کد زیر را برای روال رویداد ()Form_Load بنویسید.

```
Private Sub Form_Load()
    lblTimeDisplay.Top = frmClock.ScaleTop
    lblTimeDisplay.Left = frmClock.ScaleLeft
    lblTimeDisplay.Width = frmClock.ScaleWidth
    lblTime Display.Height = frmClock. ScaleHeight
End Sub
```

خطوط ۲ تا ۵ برای بزرگ کردن برچسب به اندازهٔ فرم است.

برای تعیین محل و ابعاد کنترل‌ها از مشخصه‌های Top، left، width و Height استفاده می‌کنیم. می‌توان علاوه‌بر چهار مشخصه فوق از مشخصات ScaleTop، Scaleleft، ScaleWidth و ScaleHeight استفاده کرد. Scalewidth و ScaleHeight به ترتیب پهنا و ارتفاع کنترل را تقسیم‌بندی می‌کنند. برای مثال Scalewidth = 100 پهنای کنترل را به ۱۰۰ قسمت مساوی تقسیم می‌کند و واحد اندازه‌گیری افقی را مشخص می‌نماید به همین ترتیب ScaleHeight=50 ارتفاع کنترل را به ۵۰ قسمت مساوی تقسیم کرده و واحد اندازه‌گیری عمودی را مشخص می‌کند.

از چهار مشخصهٔ ScaleTop، Scaleleft، ScaleWidth و ScaleHeight در مواردی استفاده

می‌کنیم که بخواهیم شیء همیشه در موقعیت ثابتی از فرم قرار داشته باشد. در کد (Form–load) بالا می‌خواهیم اندازه برچسب همیشه به اندازهٔ فرم باشد یعنی با تغییر اندازهٔ فرم، اندازه برچسب هم تغییر کند.

برای مشخص کردن واحد اندازه‌گیری ابعاد کنترل از خصوصیت ScaleMode استفاده می‌کنیم که مقادیر آن در جدول ۲ــ۶ آمده است.

جدول ۲ــ۶

ثابت‌ها	مقادیر	شرح
vbUser	۰	هرگاه حداقل یکی از مشخصات ScaleTop، ScaleWidth، ScaleHeight و ScaleLeft توسط برنامه‌نویس مقداردهی شود.
vbTwips	۱	Twip (پیش‌فرض) هر اینچ ۱۴۴۰ توئیپ و هر سانتیمتر ۵۶۷ توئیپ است.
vbPoints	۲	Point هر اینچ ۷۲ پوینت است.
vbPixels	۳	Pixel
vbCharacters	۴	Character (هر واحد افقی ۱۲۰ twips و هر واحد عمودی ۲۴۰ twips است.)
vbInches	۵	اینچ
vbMillimeters	۶	میلی‌متر
vbCentimeters	۷	سانتی‌متر

مقادیر ScaleTop و Scaleleft مساوی صفر است جز حالتی که ScaleMode مساوی صفر (vbUser) باشد.

۶ــ مشخصهٔ Interval کنترل Timer را با ۵۰۰ (نیم ثانیه) مقداردهی کنید. مقدار مشخصهٔ Enabled کنترل Timer را به True تغییر دهید.

۷ــ کد مربوط به روال Timerl_Timer() را به صورت زیر بنویسید.

```
Private Sub Timerl_Timer()
    If frmClock.WindowState = vbNormal Then
        lblTimeDisplay. Caption = CStr(Time)
        frmClock.Caption = Format(Date," Long Date")
```

```
    Else
        frmClock.Caption = CStr(Time)
    End If
End Sub
```

۸ ــ برنامه را ذخیره کرده و آن را اجرا کنید (شکل ۳ــ۶).

شکل ۳ــ۶ــ تنظیم مشخصهٔ MaxButton با مقدار False دکمهٔ به حداکثررسانی را غیرفعال می‌کند.

برنامهٔ ساعت، از چندین مورد استفاده می‌کند که ممکن است قبلاً مشاهده نکرده باشید. تابع Format() قالب نمایش تاریخ را به صورت روز، ماه و سال کامل تبدیل می‌کند. (در ادامه راجع به این تابع توضیح خواهیم داد.) همچنین تنظیم مشخصهٔ MinButton با True، دکمهٔ به‌حداقل رسانی را روی نوار عنوان نمایش می‌دهد و امکان به حداقل‌رسانی فرم را فراهم می‌کند.

فرم‌ها دارای سه مشخصهٔ MinButton، MaxButton و ControlBox هستند که روی سه دکمهٔ سمت راست نوار عنوان تأثیر می‌گذارند. MinButton و MaxButton برای فعال و غیرفعال کردن دکمه‌های Minimize و Maximize مورد استفاده قرار می‌گیرند. این مشخصه‌ها فقط در زمان طراحی و هنگامی که مشخصهٔ BorderStyle فرم دارای مقدار 1–Fixed Single یا 2–Sizeable باشد، قابل دسترسی هستند.

مشخصهٔ ControlBox منوی کنترل سمت چپ نوار عنوان را فعال یا غیرفعال می‌کند. فقط در صورتی می‌توان این مشخصه را فعال کرد که مشخصهٔ BorderStyle فرم دارای یکی ازمقادیر 1–Fixed Single ، 2–Sizable یا 3–Fixed Dialog باشد.

نکته

هنگام استفاده از مشخصهٔ Window State دقت کنید. ممکن است به‌طور اتفاقی روی فرمی که Border Style آن Fixed Single است، مشخصهٔ Window State را با Maximized مقداردهی کنید. نتیجهٔ این کار، فرمی است که کل صفحه را در برمی‌گیرد و نمی‌توان آن را تغییر اندازه داد. کاربر نیز روی این فرم کنترلی ندارد.

همچنین برنامهٔ ساعت، مقدار مشخصهٔ Window State فرم را برای تعیین به حداقل رسانی فرم،بررسی می‌کند. مشخصهٔ Window State دارای سه مقدار است :

0– Normal(vbNormal)

1– Minimized (vbMinimized)

2– Maximized (vbMaximized)

می‌توان مقدار این مشخصه را خواند و تعیین کرد که آیا پنجره در اندازهٔ عادی، تمام صفحه یا در نوار وظیفه است. همچنین می‌توان مشخصهٔ Window State را تعیین کرد تا پنجره را به صورت تمام صفحه تبدیل کند، آن را به حالت عادی برگرداند یا در نوار وظیفه با حداقل اندازه نشان دهد.

۴ــ۶ــ کاربرد تابع Format()

تابع Format() یکی از توابع قوی ویژوال بیسیک است که امکان کنترل روش نمایش رشته‌ها را فراهم می‌کند. این تابع معمولاً برای نمایش مقادیر تاریخ/زمان و اعداد استفاده می‌شود ولی می‌تواند رشته‌ها را نیز به صورت موردنظر تبدیل کند. شکل کلی این تابع، به صورت زیر است :

MyString = Format (Expression[,Format_String [,First Day Of Week [,First Week Of Year]]])

- My String نام متغیر رشته‌ای است که خروجی تابع در آن قرار می‌گیرد.
- Format نام تابع است.
- Expression عبارتی است که می‌خواهیم این تابع آن را به قالب موردنظر تبدیل کند.

Format_String الگوی رشته‌ای است که به تابع بیان می‌کند که می‌خواهد رشتهٔ خروجی به چه صورتی ظاهر شود.

- FirstDayOfWeek یک ثابت اختیاری است که اولین روز هفته را تعیین می‌کند. پیش‌فرض، Sunday است ولی اگر می‌خواهید روز شنبه باشد، باید آن را بنویسید (Saturday).
- First Week Of Year یک عبارت ثابت اختیاری است که اولین هفتهٔ سال را تعیین می‌کند. به‌طور پیش فرض، اول ژانویه، اولین هفتهٔ سال است.

کلید کار کردن با تابع Format()، آشنایی با پارامتر Format_String است. جدول ۲ــ۶ چگونگی استفاده از تنظیمات این تابع را برای تغییر ظاهر رشته‌های تاریخ و زمان ارایه می‌کند.

نکته

باید پارامتر Format_String را بین زوج کوتیشن " " قرار دهید، زیرا تابع این پارامتر را به صورت رشته به‌کار می‌برد. به‌عنوان مثال، دستور My String$ =Format(.50،percent) خطایی را تولید خواهد کرد ولی این دستور به‌صورت My String$ =Format(.50،"percent") صحیح خواهد بود.

جدول ۲ــ۶ــ کاربرد تابع Format () برای زمان و تاریخ

Format – String	مثال	نتیجه
“Long Date”	Format(36000, “Long Date”)	Friday, July24,1998
“Medium Date”	Format(36000, “Medium Date”)	24 – Jul–98
“Short Date”	Format(36000, “Short Date”)	7/24/98
“Long Time”	Format(0.874, “Long Time”)	8: 58: 34 p.m.
“Medium Time”	Format(0.874, “Medium Time”)	8: 58 p.m.
“Short Time”	Format(0.874, “Short Time”)	20: 58

جدول ۳ــ۶ چگونگی استفاده از تابع Format()برای کار کردن با مقادیر عددی برای نمایش به صورت رشتهٔ دلخواه را نشان می‌دهد.

جدول ۳ــ۶ــ کاربرد تابع Format () برای اعداد

Format String	Example	Result
"General Number"	Format(36000, "General Number")	36000
"Currency"	Format(36000, "Currency")	$36,000.00
"Fixed"	Format(36000, "Fixed")	36000.00
"Standard"	Format(36000, "Standard")	36,000.00
"Percent"	Format(36000, "Percent")	3600000.00%
"Scientific"	Format(36000, "Scientific")	3.60E+04
"Yes/No"	Format(36000, "Yes/No")	Yes
"True/False"	Format(36000, "True/False")	True
"On/Off"	Format(36000, "On/Off ")	On

همچنین می‌توان قالب‌های رشته‌ای دیگری را نیز به کار برد. به‌عنوان مثال، اگر می‌خواهید مطمئن شوید که ورودی کاربر همیشه دو رقم اعشار داشته باشد، از قالب ("###,##0.00",235.6)Format استفاده کنید که مقدار 235.60 را برمی‌گرداند.

۵ــ۶ــ محاسبهٔ اختلاف تاریخ‌ها

هنگامی که نیاز دارید، میزان زمان بین دو تاریخ را بدانید، از تابع DateDiff() استفاده کنید. تابع DateDiff() دارای شکل کلی زیر است :

```
My Long = Date Diff (Interval, Start_date, End_Date [,First Day of Week
[,First Week Of Year]])
```

- MyLong نام متغیری است که خروجی تابع در آن قرار می‌گیرد و از نوع Long است.
- DateDiff نام تابع است.
- Interval رشته‌ای است که فاصلهٔ زمانی براساس آن، تفاوت تاریخ را اندازه‌گیری خواهد کرد (جدول ۴ــ۶).

جدول ۴ــ۶ــ مقادیر مختلف پارامتر InterVal مربوط به تاریخ DateDiff()

Value	Interval	Usage	Return Value
"yyyy"	Year	DateDiff("yyyy","7/4/76","7/4/86")	10
"q"	Quarter	DateDiff("q","7/4/76","7/4/86")	40
"m"	Month	DateDiff("m","7/4/76","7/4/86")	120
"y"	Day of year	DateDiff("y","7/4/76","7/4/86")	3652
"d"	Day	DateDiff("d","7/4/76","7/4/86")	3652
"w"	Weekday	DateDiff("w","7/4/76","7/4/86")	521
"ww"	Week	DateDiff("ww","7/4/76","7/4/86")	521
"h"	Hour	DateDiff("h","7/4/76","7/4/86")	87648
"n"	Minute	DateDiff("n","7/4/76","7/4/86")	5258880
"s"	Second	DateDiff("s","7/4/76","7/4/86")	315532800

- Start_Date تاریخ شروع است (از نوع Date).
- End_Date تاریخ پایان است (از نوع Date).
- FirstDayOf Week، یک ثابت اختیاری است که اولین روز هفته است.
- FirstWeekOfYear، عبارت ثابت اختیاری است که اولین هفتهٔ سال را تعیین می‌کند.

تابع ()DateDiff اولین تاریخ (Start_Date) را منهای دومین تاریخ (End_Date) می‌کند. بعد از تفریق، تابع عددی از نوع Long را برمی‌گرداند که اختلاف بین این تاریخ‌هاست. واحد اندازه‌گیری این اختلاف، براساس مقدار رشتهٔ پارامتر Interval تعیین می‌شود (جدول ۴ــ۶).

نکته

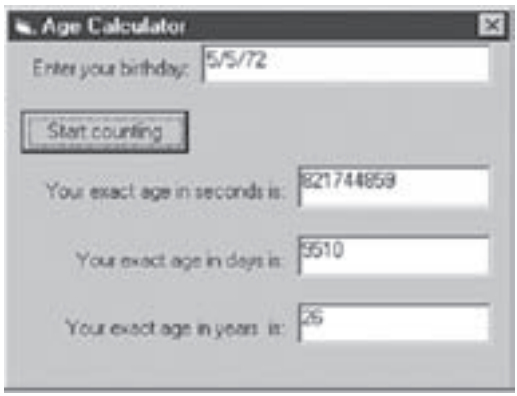
()IsDate یک تابع VB است که یک رشته یا مقدار تاریخ را بررسی می‌کند. اگر رشته یا تاریخ شبیه یک تاریخ معتبر باشد، تابع ()IsDate مقدار True را برمی‌گرداند و در غیراین‌صورت مقدار False را برمی‌گرداند. به عنوان مثال، IsDate("September 16 1998") و IsDate(#9/16/98#) مقدار True را برمی‌گرداند ولی IsDate("Birthday") مقدار False را برمی‌گرداند.

تابع CDate ورودی خود را به نوع Date تبدیل می‌کند لذا هرگاه رشته‌ای از نظر تابع IsDate شبیه یا یک تاریخ معتبر است می‌توان با استفاده از تابع CDate آن را به قالب تاریخ (نه شبیه به تاریخ) تبدیل کرد.

مثال ۴ــ۶

برنامه‌ای بنویسید که تاریخ تولد کاربر را دریافت و در صورت معتبر بودن، اعلام کند که چند سال، چند روز و چند ثانیه از آن تاریخ گذشته است.

پروژه‌ای مطابق شکل ۴ــ۶ ایجاد کنید (نام پروژه را DateDiff.vbp قرار دهید).



شکل ۴ــ۶ــ استفاده از تابع ()Format خواندن مقادیر روز و ثانیه‌ها را ساده‌تر می‌کند.

این پروژه چگونگی استفاده از تابع DateDiff() را برای گزارش سن شخص برحسب ثانیه، روز و سال، شرح می‌دهد. کاربر، تاریخ تولد خودش را در کادر متن وارد می‌کند و سپس روی دکمهٔ Start Counting کلیک می‌کند. روال رویداد Click مربوط به دکمه، ورودی را بررسی کرده و درصورتی که شبیه یک رشتهٔ تاریخی باشد، با استفاده از تابع IsDate() مقدار True را برمی‌گرداند. اگر رشته به عنوان تاریخ معتبر باشد، متن تاریخ تولد به تاریخ تبدیل شده و در متغیری به نام Bdate ذخیره می‌شود، سپس Timer فعال می‌شود اما در صورتی که رشته به عنوان تاریخ معتبر نباشد، پیغام خطایی ظاهر می‌شود.

بعد از این که کاربر کادر پیام را بست، مکان‌نما به کادر متن برگشته و متن نادرست مشخص شده و از روال خارج می‌شود.

در روال رویداد Timer1_Timer() که هر نیم‌ثانیه (Interval = 500) اجرا می‌شود، کنترل Timer با استفاده از تابع Now، تاریخ و ساعت سیستم را به‌دست می‌آورد. روال رویداد از تاریخ DateDiff() سه بار استفاده می‌کند (یک بار برای به‌دست آوردن اختلاف زمان برحسب ثانیه و سپس روز و در پایان سال). مقادیر برگشتی در متغیرهای LngYear و LngSec قرار داده می‌شود. کد زیر، رویدادهای مربوط به این برنامه را ارائه می‌کند :

```
01    Private Sub cmdStart_Click()
02
05    If IsDate(txtBDate. Text)Then                    بررسی اعتبار ورودی
09    Bdate = CDate(txtBDate.Text)                     تبدیل به فرمت تاریخ
10    Else
12    MsgBox "You must enter a proper date!",vbCritical, "Data error"
14    txtBDate.SetFocus                                انتقال فوکوس به کادر متن
16    txtBDate.SelStart = 0
18    txtBDate.SelLength = Len(txtBDate.Text)  Highlight کردن متن داخل textbox
20    Exit Sub
21    End If
24    Timer1.Enabled = True                            فعال کردن تایمر
25    End Sub
```

```
26
27      Private Sub Timerl_Timer()
28      Dim LngSec As Long
29      Dim LngDay As Long
30      Dim LngYear As Long
31
33       LngSec = DateDiff("s", BDate, Now)          چند ثانیه از تاریخ تولد گذشته
34       LngDay = DateDiff("d", BDate, Now)          چند روز از تاریخ تولد گذشته
36      LngYear = DateDiff("yyyy", BDate, Now)       چند سال از تاریخ تولد گذشته

39
42      lblAgeSecs.Caption = CStr(LngSec)
43      lblAgeDays.Caption = CStr(LngDay)
44      lblAgeYears.Caption = CStr(LngYear)
45      End Sub
```

۶ــ۶ــ کاربرد متغیرهای ایستا به همراه Timer

فرض کنید می‌خواهید برنامه‌ای بنویسید که عملی را در هر نیم‌ثانیه تا ۱۰ بار انجام دهد. برای انجام این کار، نیاز دارید متغیری ایجاد کنید که تعداد دفعات اجرای رویداد Timer را نگه دارد. اگر یک متغیر شمارنده داخل رویداد Timer ایجاد کنید، هربار که روال قطع شود، متغیر از حوزهٔ عمل خود خارج شده و با صفر مقداردهی می‌شود.

```
        Dim i As Integer
01      Private Sub Timer1_Timer()
02          Form1.Caption = "i is: " & Cstr(i)
03          If I > 10 Then Timer1.Enabled = False
04          i = i + 1
05      End Sub
```

اگرچه این کد، کار خواهد کرد ولی بهینه نیست، زیرا ایجاد متغیر عمومی آن را مستقل از روال می‌کند. اگر بخواهید Timer را از کد حذف کنید، این متغیر عمومی، بدون استفاده خواهد ماند.

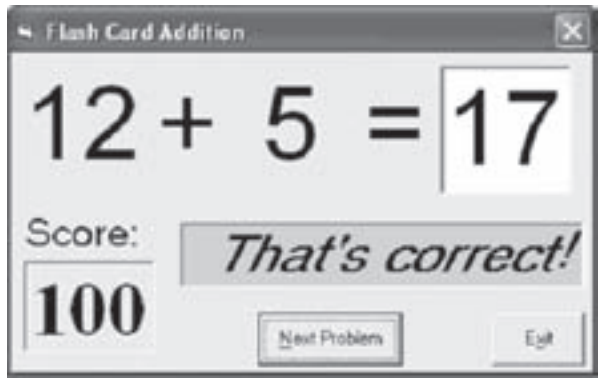
یک راهکار بهتر، تعریف متغیر شمارنده با کلید واژهٔ Static است. برای انجام این کار در کد فوق، بین خطوط ۱ و ۲ کد زیر را بنویسید :

```
Static i As Integer
```

هنگامی که یک متغیر Static ایجاد می‌کنید، مقدار آن حتی بعد از خروج از روالی که در آن تعریف شده است، حفظ می‌شود. مزیت انجام این کار، این است که متغیر درون کنترلی که به آن وابسته است، کپسوله‌سازی می‌شود. مقدار متغیر بدون درنظرگرفتن وضعیت روالی که در آن ایجاد شده است، پایدار باقی می‌ماند.

مثال ۵ــ۶

فرمی به‌صورت زیر (شکل۵ــ۶) ایجاد کنید. می‌خواهیم با کلیک روی دکمهٔ Next Problem به‌طور خودکار ۲ عدد تصادفی (بین صفر تا ۲۰) تولید شود و حاصل‌جمع آن‌ها را کاربر تایپ کند. در صورتی که پاسخ مثبت بود، به امتیاز Score ، ۵ امتیاز اضافه شود و پیغام صحیح بودن بدهد. اگر پاسخ نادرست بود، ۵ امتیاز کسر کند و با پیغام خطا پاسخ درست را اعلان کند (توجه کنید که امتیاز از ۱۰۰ بیشتر نباشد). با کلیک روی Next این عملیات تکرار شود.



شکل ۵ــ۶

کد این برنامه به‌صورت زیر جمع خواهد بود :

```
Option Explicit
Dim Sum As Interger
```

```
Dim NumProb As Integer, NumRight As Integer

Private Sub cmdExit_Click()
        End
End Sub
Private Sub cmdNext_Click()
    Dim Number1 As Integer
    Dim Number2 As Integer
    txtAnswer.Text =" "
    lblMessage. Caption = " "
    Number 1 = Int(Rnd *21)                    تولید عدد تصادفی اول بین ۰ و ۲۱
    Number2 = Int(Rnd *21)                     تولید عدد تصادفی دوم بین ۰ و ۲۱
    LbINum1. Caption = Number1                 نمایش عدد اول
    LbINum2.Caption = Number2                  نمایش عدد دوم
    Sum = Number1 + Number2                    محاسبهٔ حاصل جمع دو عدد
    cmdNext.Enabled = False
    txtAnswer. SetFocus        انتقال فوکوس به کادر متن txtAnswer برای دریافت جواب
End Sub
Private Sub Form_Activate()
    Call cmdNext_Click
End Sub

Private Sub Form_Load()
    Randomize Timer
    NumProb = 0
    NumRight = 0
End Sub

Private Sub txtAnswer_KeyPress(KeyAscii As Integer)
```

```
Dim Ans As Integer
If KeyAscii = 13 Then                                   در صورت فشرده‌شدن کلید Enter
    If Val(txtAnswer.Text) = Sum Then   بررسی محتوای txtAnswer با حاصل جمع واقعی
    lblMessage.Caption = "That's correct!"
    If Val (lblScore.Caption) < 100 Then lblScore.Caption = lblScore.
    Caption +5                                  به روزرسانی امتیاز با اضافه‌کردن به میزان ۵
Else
    lblMessage.Caption = "Answer is " +Sum                  نمایش پاسخ صحیح
    IblScore.Caption = lblScore.Caption – 5   به روزرسانی امتیاز با کم‌کردن به میزان ۵
End If
    cmdNext. Enabled = True
    cmdNext.SetFocus
End If
End Sub
```

تمرین: شمارش معکوس

پروژه‌ای مطابق شکل ۶ـ۶ ایجاد کنید و یک کنترل Timer و یک برچسب روی فرم قرار دهید تا شمارش معکوس را شبیه‌سازی کند.

شکل ۶ـ۶

خودآزمایی

۱ــ در هریک از دستورات زیر مقدار ذخیره‌شده در متغیر strS را مشخص کنید.

الف) strS = Format(d, "hampm")

ب) strS = Format(12345.67, "######.### ")

۲ــ فرمی تهیه کنید به این شرح که، با انتخاب تاریخ، در قسمت پایین تقویم، تاریخ نمایش داده شود. با کلیک روی دکمهٔ مقایسه، تاریخ با تاریخ روز مقایسه شود و پیغام دهد که آیا تاریخِ گذشته یا آینده است؟

۳ــ به مثال ۱۰ــ۶ دکمهٔ CmdTimer اضافه کنید که با کلیک آن تصاویر به طور خودکار نمایش داده شوند (هر ۱۰ ثانیه یک تصویر) و با کلیک دوبارهٔ این دکمهٔ نمایش خودکار متوقف شود.

۴ــ برنامه‌ای بنویسید که از ۱ تا ۱۰ را بشمارد و فاصلهٔ بین هر عدد با عدد بعدی ۳ دقیقه باشد (هر سه دقیقه به شماره قبلی یکی اضافه کند) برنامه را یکبار با استفاده از متغیر عمومی و بار دیگر با استفاده از متغیر ایستا بنویسید.

۵ــ برنامه‌ای بنویسید که تاریخ تولد دو دانش‌آموز را دریافت کرده و اختلاف سنی آن‌ها را برحسب ثانیه، روز و سال نمایش دهد.