برنامه سازی (3)

فصل اول

روال و تابع: مجموعه ای از دستورات که تحت یک نام نوشته شده و در هرجا از برنامه و به تعداد نامحدود برای اجرا فراخوانی می گردد.

1- روال(sub): شیوه تعریف بصورت زیر است:

```
Private یا public sub نام روال( پارامترهای ورودی)
    مجموعه ای از دستورات برنامه
End sub
```

- می توان تعریف آنرا در پنجره تعریف روال هم انجام داد.
  Tools→ procedure ….
- روال مقدار برگشتی ندارد. یعنی نتیجه کار آن بصورت مقداری بر نمی گردد.

فراخوانی: فراخوانی به معنی صدا زدن روال یا تابع می باشد و تا زمانیکه روال یا تابع فراخوانی نشود، اجرا نمی گردد.

الف- روال به صورت زیر فراخوانی می شود:

Call نام روال → بدون پارامتر

Call نام روال(مقادیر پارامترها)→ اگر دارای پارامتر ورودی باشد

مثال: روالی تعریف کنید که اعداد مضرب 5 کوچکتر از 1000 را چاپ کند. سپس داخل رویداد یک دکمه فراخوانی کنید.

```
Private  sub  mazrab5() → تعریف روال برای چاپ مضربهای 5
        Dim I as integer
        For  I = 5 to  1000  step 5
                Print  I;
        Next  I
End  sub
Private Sub Command1_Click()→ فراخوانی در رویداد دکمه
        Call   mazrab5
End Sub
```

مثال 2: روالی به نام sum برای چاپ مجموع اعداد 1 تا 10:

نکته: روال یا تابع تا فراخوانی نگردد، دستورات آن اجرا نمی شوند.

```
Private Sub sum()→ روال محاسبه مجموع
    Dim i As Integer, s As Integer
    s = 0
    For i = 1 To 10
      s = s + i
    Next i
    Print "sum = " & s
End Sub
Private Sub Command1_Click()→ فراخوانی در رویداد دکمه
        Call sum
End Sub
```

پارامتر چیست؟ (نام دیگر : آرگومان)

ورودی به روال یا تابع است که بر اساس آن می تواند در شرایط مختلف کار کند. هر پارامتر بصورت زیر داخل پرانتز تعریف می شود(مانند متغیر بدون dim):

نام پارامتر AS نوع

مثال: روال mazrab5 را طوری تغییر دهید که مضارب دلخواه(m) بر اساس پارامتر ورودی را چاپ کند.

```
Private Sub  Mazrab(m As Integer) → روال چاپ مضربها
  Dim I As Integer
 For I = m To 1000 Step m
 If (I Mod 100 = 0) Then Print →بعد از چاپ 100 عدد به خط بعد برود
 Print I;
 Next I
End Sub
Private Sub Command1_Click()→ فراخوانی در رویداد دکمه
        Dim a  as integer
        A = Val(Text1.Text)
        Call Mazrab(a)  → مقدار  a در پارامتر m قرار گرفته و روال اجرا می گردد.
End Sub
```

مثال: محاسبه ب.م.م و ک.م.م

```
Private Sub   kbmm (m As Integer, n as integer)
→روال محاسبه و چاپ بزرگترین مقسوم علیه مشترک و کوچکترین مضرب مشترک دو عدد
  Dim bmm As Integer, kmm As Integer
  Dim a As Integer, b As Integer
  a = m
  b = n
  Do While (m <> n)
    If (m > n) Then
      m = m - n
    Else
      n = n - m
    End If
  Loop
  bmm = n
  kmm = a * b / bmm
  Print "bmm = " & CStr(bmm)
  Print "kmm = " & CStr(kmm)
End Sub
Private Sub Command1_Click()→ فراخوانی در رویداد دکمه
        Dim a As Integer, b As Integer
          a = Val(Text1.Text)
          b = Val(Text2.Text)
      Call kbmm(a, b)
End Sub
```

2- تابع(function): شیوه تعریف بصورت زیر است:

```
Private یا public  function  نام تابع (پارامترهای ورودی) as  نوع خروجی
   مجموعه ای از دستورات برنامه
   نام تابع =  مقدار برگشتی
End function
```

- تابع مقدار برگشتی دارد. یعنی نتیجه کار آن بصورت مقداری بر می گردد.
- تابع به صورت زیر فراخوانی می شود: نخست باید متغیری از نوع خروجی تابع برای ذخیره نتیجه تعریف کنیم:

```
Dim R as  نوع داده خروجی تابع
R= نام تابع()→ اگر پارامتر نداشته باشد
یا
R= نام تابع(مقادیر پارامتر)→ اگر پارامتر داشته باشد
```

مثال(1): تابعی تعریف کنید که مجموع اعداد کوچکتر از 1000 را محاسبه کرده و به عنوان نتیجه برگرداند . سپس داخل رویداد یک دکمه فراخوانی کنید و با msgbox نتیجه را نشان دهید.

sum= 1+2+3+…+1000

```
Private Function Sum() as Long → تعریف تابع
    Dim a  As Integer, s As Long
    s = 0
    For a= 1 To 1000
      s = s + a
    Next  a
    Sum  =  s   →ذخیره نتیجه در نام تابع
End   function
```

```
Private Sub Command1_Click() → فراخوانی در رویداد دکمه
        Dim  r  as  Long
        r= Sum() →فراخوانی تابع
        msgbox   " sum = "  &  r
End Sub
```

مثال (2): تابعی برای محاسبه فاکتوریل عدد موجود در یک Text

N! = 1*2*3*…*N

```
Private Function Factorial() as Long → تعریف تابع
    Dim a  As Integer, s As Long , n  as integer
   n = Val(Text1.Text)  →عدد ورودی
    s = 1
    For a = 1 To n →  محاسبه فاکتوریل
        s = s * a
    Next  a
    Factorial  =  s   →ذخیره نتیجه در نام تابع
End   function
```

```
Private Sub Command1_Click() → فراخوانی در رویداد دکمه
        Dim  r  as  Long
        r= Factorial ()
        msgbox   "Factorial = "  &  r
End Sub
```

مثال (3): تابع فاکتوریل را طوری تغییر دهید که فاکتوریل عدد مورد نظر در پارامتر را حساب کند:

```
→ تعریف تابع با پارامتر
Private Function Factorial (n as integer) as Long
    Dim a  As Integer, s As Long
    s = 1
    For a = 1 To n →  محاسبه فاکتوریل
        s = s * a
    Next a
    Factorial  =  s   →ذخیره نتیجه در نام تابع
End   function
```

```
Private Sub Command1_Click() → فراخوانی
Dim  r  as  Long, a as integer
a = Val(Text1.Text)  →عدد ورودی
r= Factorial (a)   →  فراخوانی تابع برای محاسبه
فاکتوریل عدد داده شده در پارامتر و برگشت نتیجه
msgbox   a & " ! = "  &  r
End Sub
```

مثال (4): تابعی تعریف کنید که عددی اعشاری(n) را به توان عددی صحیح(m) برساند.

توضیح: n^m یعنی m مرتبه n را در خود ضرب کنیم.

```
Private function  power(n as single, m as integer) as single
   Dim s as single,a  as integer
   S=1
   For a= 1 to m →  مرتبه m
      S= s* n    →ضرب  در خودش
   Next  a
  Power = s    →نتیجه در نام تابع قرار می گیرد
End   function
```

```
Private Sub Command1_Click() → فراخوانی
   Dim a as single, b as integer, t as single
    a = Val(Text1.Text)  →عدد اعشاری مورد نظر
    b = Val(Text2.Text)  →عدد صحیح برای توان
    t = power(a,b)  →فراخوانی تابع
    Text3.Text = cstr(t)   →نمایش نتیجه توان
End Sub
```

تمرین (1): به کمک مثال 2: تابعی به نام sum تعریف کنید که مجموع اعداد 1 تا عدد داده شده در پارامتر (n) را بدست آورد. فراخوانی آنرا در یک رویداد دکمه (ورودی از Text)نشان دهید.

تمرین (2): به کمک روال محاسبه مقسوم علیه ها تابعی به نام prime تعریف کنید که اول بودن یا نبودن عدد داده شده در پارامتر (n) را مشخص نماید. فراخوانی آنرا در یک رویداد دکمه (ورودی از Text)نشان دهید.(راهنمایی: عدد اول عددی است که غیر از یک و خودش مقسوم علیه دیگری ندارد مانند عدد 13و17و53و....)

تمرین: روالی(sub) تعریف کنید که قد (برحسب متر) و وزن (برحسب کیلوگرم) را به عنوان پارامتر گرفته و با محاسبه نسبت زیر وضعیت تناسب اندام شخص را تعیین نماید:

نسبت = وزن / (قد)^2

حالات نسبت:

- کمتر از 20: عدم تناسب (وزن کم است)
- بین 20 تا 25: تناسب (عادی)
- بیش از 25 : عدم تناسب(اضافه وزن)

نکات مهم در فراخوانی تابع و روال پارامتردار

- رعایت کردن تعداد پارامترها: زیاد و کم نباشد
- رعایت کردن نوع پارامترها: نوع داده ها باید با هم سازگار باشند
- رعایت کردن ترتیب پارامترها: ترتیب پارامترها و داده های ارسالی باید یکی باشد

مثال: در تابع مثال توان (power) دو پارامتر داریم که اولی اعشاری و دومی صحیح است.

```
Private function  power(n as single, m as integer) as single
        مجموعه دستورات تابع
End function
```

فراخوانی های درست:

```
Dim r as single
r= power(6.5 , 3)   یا   r= power(6 , 3)
```

فراخوانی های نادرست:

```
Dim r as single
r= power(6 , 3.5) → ترتیب اول اعشاری و دومی صحیح رعایت نشده است
r= power(6)  →  تعداد(کم است) رعایت نشده است
r= power(6.5 , 3.2)  →  نوع رعایت نشده است(دومین پارامتر اعشاری نیست)
r= power(7,  3 , 4)  →  تعداد(زیاد است) رعایت نشده است
```

نکته: اگر ترتیب پارامترها را ندانیم ولی نام پارامترها را بلد باشیم می توان در فراخوانی به کمک روش آرگومان نامدار به ترتیب دلخواه بنویسیم. برای اینکار مقدار پارامتر را بصورت زیر تعیین می کنیم:

نام پارامتر :=مقدار

```
r= power( m := 3 , n := 5.3)   یا   r= power( n := 5.3 , m := 3)
```

خروج از زیر برنامه ها

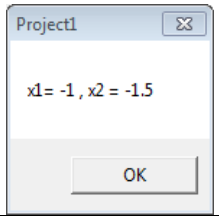
خروج از تابع یا روال: اگر بخواهیم دستورات تابع یا روال از یک نقطه به بعد اجرا نشوند. دستورات زیر را استفاده می کنیم:

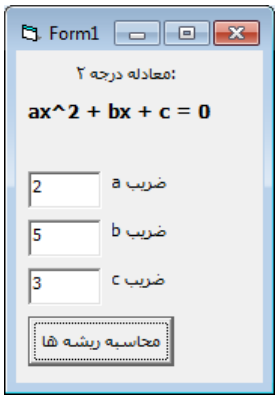
```
Exit  function   → خروج از تابع
Exit  sub  → خروج از روال
```

تمرین: روالی بنویسید که ضرایب معادله درجه 2 را به عنوان پارامتر بگیرد و ریشه های آنرا در صورت وجود چاپ کند.

```
Private sub  rishe(a as integer , b as integer, c  as integer)
   Dim x1  as single , x2 as single
   Dim delta as integer
   Delta =  b ^ 2 –  4 * a * c
   If (delta < 0 ) then     →اگر دلتا منفی باشد ریشه ندارد و باید خارج شود
         Msgbox "ریشه ندارد!"
         Exit sub  →خروج از روال به دلیل عدم وجود ریشه
    End if
   X1= (- b + sqr (delta)) / (2*a)  →ریشه اول
   X2= (- b - sqr (delta)) / (2*a)  →ریشه دوم
   Msgbox "x1= " & cstr (x1)  &  " , x2 = "  &  cstr (x2)  →نمایش ریشه ها
End sub

Private Sub Command1_Click()→ فراخوانی
         Dim a  as integer
         Dim  b  as integer, c  as integer
         a= Val(Text1.Text)
         b= Val(Text2.Text)
         c= Val(Text3.Text)
         call  rishe(a,b,c)
End sub
```

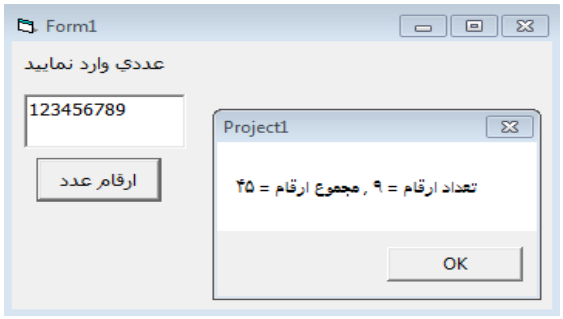




تمرین (4): روالی تعریف کنید که عددی صحیحی(n) را به عنوان پارامتر گرفته و تعداد و مجموع رقم های آنرا نمایش دهد.

```
Private Sub Digits(n As Long)
   Dim sum As Integer, r As Integer, num As Integer
   sum = 0     'متغیر جمع ارقام
   num = 0     'متغیر تعداد ارقام
   Do While (n > 0)     ' تا زمانیکه عدد صفر نشده است
      r = n Mod 10      'بدست آوردن رقم آخر
      sum = sum + r     'جمع ارقام
      num = num + 1     'شمارش ارقام
      n = n \ 10        'بقیه عدد
   Loop
   MsgBox "تعداد ارقام = " & CStr(num) & _
                  " ,مجموع ارقام = " & CStr(sum)
End sub
```

```
Private Sub Command1_Click()
   Dim n As Long
   n = Val(Text1.Text)
   Call Digits(n)
End Sub
```



انواع متغیرها

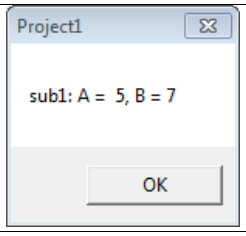
۱. **سراسری**: در قسمت general تعریف می شوند و در کل برنامه بصورت مشترک قابل دسترسی هستند
۲. **محلی**: داخل رویدادها یا زیر برنامه ها تعریف شده و فقط داخل زیر برنامه ای که در آن وجود دارند قابل دسترسی و استفاده هستند. این متغیرها با اتمام کار زیر برنامه خود بخود از بین می روند.

حوزه عمل متغیرها: حوزه عمل(میدان دید) قابلیتی را تعیین می کند که متغیرهای محلی با نام یکسان(که هر یک در زیربرنامه ای جداگانه تعریف شده اند)، دارای مقادیر و دوره حیات مختلف می باشند.

مثال:

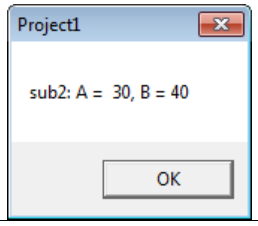
```
Private Sub sub1()
   Dim A  As Integer, B As Integer
   A = 5
   B = 7
   MsgBox "sub1: A =  " & A & ", B = " & B
End sub
```

مقادیر A,B در روال Sub1 بصورت زیر نمایش می یابد




```
Private Sub sub2()
   Dim A  As Integer, B As Integer
   A = 30
   B = 40
   MsgBox "sub2: A =  " & A & ", B = " & B
End sub
```

مقادیر A,B در روال Sub2 بصورت زیر نمایش می یابد



ایجاد برنامه در VB بدون اینکه فرمی ظاهر شود

۱. ایجاد پروژه جدید
۲. حذف form1 با کلیک راست و انتخاب remove روی نام فرم در بخش پانل project
۳. ایجاد یک ماژول:

Project → add module

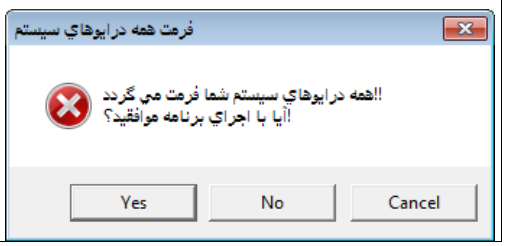
۴. نوشتن دستورات مورد نظر در یک روال با نام sub Main() داخل این ماژول
۵. تنظیمات پروژه و انتخاب این روال به عنوان برنامه startup

Project→ properties → startup object→ sub Main()

۶. اجرا

مثال: برنامه ای بدون فرم بسازید که پیام دلخواهی نمایش دهد.

```
Sub main()
   Dim s As String, r As Integer
    s = "همه درایوهای شما فرمت می شوند." & vbCrLf    & "آیا با اینکار موافقید؟"
   r = MsgBox(s, vbCritical + vbYesNoCancel)
   If r = vbYes Then
      MsgBox "نگران نباش! سرکاری بود.", vbInformation + vbOKOnly
   ElseIf r = vbNo Then
      MsgBox "نمی توانید جلو آنرا بگیرید!! دیگر ", vbExclamation + vbOKOnly
   Else
      MsgBox "به خیال خود چکار می کنید؟", vbCritical + vbOKOnly
   End If
End Sub
```



توابع تشخیص انواع داده ها

- IsDate() بررسی اینکه آرگومان داده شده از نوع تاریخ است یا نه؟
- IsEmpty(): آیا آرگومان مورد نظر مقدار گرفته یا خالی است؟
- IsNull(): آیا آرگومان مقدار NULL (یعنی پوچ) دارد؟
- IsNumeric(): آیا آرگومان داده شده عددی است یا نه ؟

مثال:

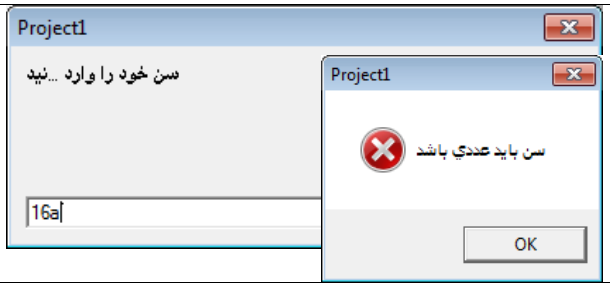
Dim v1 as variant, v2 as variant → نوع داده همه منظوره variant
Dim v3 as variant, v4 as variant
V1=0 →مقدار دهی صفر
V2=NULL →مقدار دهی پوچ
V3="" →مقدار دهی رشته خالی

استفاده از توابع تشخیص نوع داده و جواب حاصل :

IsNumeric(v1) → True چون عددی است
IsNull(v2) → True چون مقدار آن پوچ است
ISEmpty(v4) → true چون به آن هیچ مقداری داده نشده است
IsEmpty(v2) → false چون به آن مقدار پوچ داده شده و خالی نیست
IsEmpty(v3) → false چون به آن مقدار رشته داده شده و خالی نیست(مقدار گرفته)
IsNull(v3) → false چون به آن مقدار رشته خالی داده شده و پوچ نیست
IsNumeric("AB125?") → false چون عددی نیست

مثال: در یک دکمه به کمک Inputbox سن خود را بخوانید و اگر عدد وارد نشد، پیام خطا نمایش دهید

```
Dim s as string
S = inputbox("سن خود را وارد کنید")
If  IsNumeric(s)  then
        Msgbox "با تشکر",vbinformation
Else
        Msgbox "سن باید عددی باشد" ,vbcritical
End if
```



تابع بدست آوردن نوع داده : جدول صفحه 17 کتاب

VarType(نام متغیر)

مثال:

Dim a as integer, b as single, c as byte, s as string

varType(a) →: جواب (vbInteger) یعنی عدد صحیح است: 2 یا
varType(b) → (vbSingle) یعنی عدد اعشاری است: 4 یا
varType(c) → (vbByte) یعنی عدد صحیح بایت است: 17 یا
varType(s) → (vbString) یعنی رشته است: 8 یا

توابع تبدیل انواع داده ها به همدیگر: جدول صفحه 20 کتاب

- CInt(): تبدیل به عدد صحیح(گرد کردن)
- CStr(): تبدیل به رشته
- CDate(): تبدیل به تاریخ

```
CInt(5.7) → 6
CInt(3.4)  → 3
Cint(3.5) → 3
CInt(3.5001) → 4
Cstr(1235) → "1235"
CDate("1389/09/30") → 1389/09/30
```

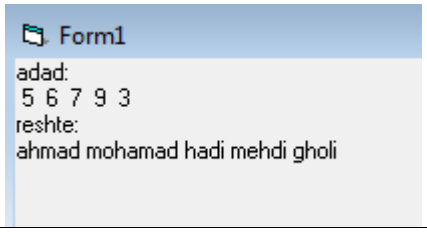
تابع Array

تابعی است که آرایه از نوع variant ایجاد می کند. اندیس شروع صفر است مگر اینکه option base 1 داشته باشیم.

```
Dim v as variant
V= Array(لیست مقادیر از هر نوع)
```

مثال: در یک رویداد دکمه آرایه ای همه منظوره ایجاد کنید و هر بار مقدارهای انواع مختلف آنرا چاپ کنید.

```
dim v as variant, I as integer
V= Array(5,6,7,9,3)  → آرایه ای از اعداد
Print  "adad:"
For I = Lbound(V) to UBound(V)
        Print  v(i);
Next  I
V= Array("ahmad","mohamad","hadi","mehdi","gholi") → آرایه ای از رشته
print
Print  "reshte:"
For I = Lbound(V) to UBound(V)
        Print  v(i);
Next  I
```



فصل دوم- پردازش فایل(پرونده ها)

هدف: ذخیره داده ها و نتایج برنامه در یک رسانه ذخیره سازی مثل دیسک سخت بصورت پرونده (فایل)

انواع فایلها

۱. **ترتیبی**(Sequential): ساده ترین نوع فایل که داده ها به همان ترتیب ذخیره شده، قابل بازیابی هستند. کند بودن کار با آنها از معایب آنهاست.
۲. **تصادفی**(Random): نوشتن و خواندن از هر نقطه از فایل امکان پذیر است. سرعت بالاتری دارند ولی پیچیده تر هستند.
۳. **دودویی**(Binary): نوع خاصی از فایلهای تصادفی

مراحل کار با فایلها و ذخیره و بازیابی اطلاعات

- شماره فایل یا موقعیت فایل روی دیسک(File Handle)

```
Dim fn as integer
```

- FreeFile: بدست آوردن شماره ای آزاد برای فایل از طریق سیستم عامل

```
Fn = FreeFile()
```

- Open : آماده سازی و بازکردن فایل جهت خواندن و نوشتن

```
Open "d:\myfile.txt" for output as #fn
```

- Print(Write): نوشتن در یک فایل

```
Print # شماره فایل , لیست داده ها
Print #fn, "harchi pet khosha!"
```

- Line Input : خواندن خطی از داده ها

```
Dim s as string
Line Input #fn, s
```

- Close : برای بستن فایلهای باز شده

```
Close #fn
```

✓ اگر هیچ شماره ای جلو آن نوشته نشود، تمام فایلهای باز را می بندد.

قالب کلی دستور open

Open	path	for	Input/output/ Append/Random/ Binary	As	#File number	Len = LengthBuffer
نام دستور	مسیر ذخیره فایل شامل درایو و مسیر	کلمه کلیدی	حالت بازکردن فایل	کلمه کلیدی	شماره فایل	طول رکورد (اختیاری)

حالات باز کردن فایل(طبق جدول صفحه 26-27)

نوع فایل	حالت باز کردن فایل	شرح و کاربرد	اگر فایل موجود *باشد*	اگر فایل موجود *نباشد*
ترتیبی	Append	افزودن اطلاعات جدید به انتهای فایل	باز می کند	ایجاد می کند
ترتیبی	Input	برای خواندن از فایل	باز می کند	خطا روی می دهد
ترتیبی	Output	برای نوشتن در فایل	محتوای قبلی فایل پاک می شود	ایجاد می کند
تصادفی	Random	دسترسی تصادفی و ذخیره بصورت رکوردی	باز می کند	ایجاد می کند
دودویی	Binary	برای ذخیره بصورت دودویی(بیت و بایت)	باز می کند	ایجاد می کند

طول رکورد: اطلاعات در فایلهای تصادفی بصورت مجموعه ای از N بخش(شبیه آرایه) ذخیره می شود که هر بخش رکورد نام دارد و هر یک شماره ای دارد. اگر از دستور option Base 1 در اول برنامه استفاده

کنیم، شماره رکوردها از 1 شروع می گردد. پارامتر آخر دستور Open به اندازه یا طول هر رکورد اشاره دارد.

تخصیص شماره به فایل : همزمان چند فایل می توانند باز باشند. برای جلوگیری از اشتباه در انتخاب شماره تکراری، از تابع زیر کمک می گیریم که یک شماره آزاد به فایل می دهد.

```
dim fn as integer
fn = FreeFile(255)
```

عدد داخل پرانتز اختیاری است و محدوده انتخاب را مشخص می کند که از 1 تا 511 می باشد. اگر محدوده تعیین نشود، پیش فرض محدوده 1-255 منظور می شود.

fn = FreeFile() → اختصاص شماره در محدوده 1-255

چند مثال:

- باز کردن یک فایل ترتیبی متنی برای ایجاد و نوشتن

```
Open "d:\myfile.txt" for output as #fn
```

- باز کردن یک فایل ترتیبی متنی برای خواندن

```
Open "d:\myfile.txt" for input as #fn
```

- باز کردن یک فایل ترتیبی متنی برای نوشتن و اضافه کردن به آخر آن

```
Open "d:\myfile.txt" for Append as #fn
```

کار با فایلهای ترتیبی: خواندن و نوشتن آنها *به ترتیب* از اول تا آخر انجام می شود. مثلاً اگر در یک فایل متنی 1000 بایتی، بخواهید یک بایت را اصلاح کنید، باید همه فایل را خوانده و بعد از اصلاح همه را دوباره نوشت.

نوشتن در فایل ترتیبی: نخست باید فایل در حالت output یا Append باز شود و سپس از دستور Print یا write استفاده کنید.

Print #fn, لیست داده ها برای نوشتن

به جای لیست داده ها می توان موارد زیر را نوشت:

- *Spc (تعداد فاصله خالی)* : قراردادن فاصله خالی به تعداد مورد نظر بین خروجی ها

Print #fn, Spc(5) → نوشتن 5 فاصله

- *Tab (شماره ستون)* : قراردادن فاصله خالی به اندازه Tab بین خروجی ها
- عبارت عددی یا رشته ای دلخواه
- *کارکتر خاص* برای محل نوشتن مقدار خروجی بعدی: علامت (;) باعث نوشتن خروجی ها به دنبال هم می شود. اگر این علامت بکار نرود، خروجی بعدی به خط بعد می رود

مثال 1: برنامه ای بنویسید که هر متن وارد شده در یک TextBox را به یک فایل متنی اضافه کند.

حل:

۱. یک TextBox و دو دکمه(Save , Close) به فرم اضافه نمایید.

۲. در قسمت General:

```
Dim fn as integer
```

۳. در رویداد Form_Load دستورات زیر را بنویسید

```
fn = FreeFile()
Open app.path & "\myfile.txt" for Append as #fn   → app.path :مسیر ذخیره پروژه
```

۴. در رویداد دکمه Save دستور زیر را بنویسید

```
Print #fn , spc(5) , Text1.Text ;   → spc(5) بین هر متن و متن بعدی 5 فاصله قرار می دهید
```

۵. در رویداد دکمه Close دستورات زیر را بنویسید

```
Close #fn
End
```

مثال 2: اعداد فرد کوچکتر از 1000 را در فایلی ذخیره کنید.

حل:

۱. دو دکمه(Save , Close) به فرم اضافه نمایید.

۲. در قسمت General:

```
Dim fn as integer
```

۳. در رویداد Form_Load دستورات زیر را بنویسید

```
fn = FreeFile()
Open  app.path & "\Fard.txt"  for  Output  as  #fn
```

۴. در رویداد دکمه Save دستورات زیر را بنویسید

```
Dim I as integer
For I =1 to 1000 step 2
     Print  #fn ,  spc(3)  ,  i ;
Next  I
```

۵. در رویداد دکمه Close دستورات زیر را بنویسید

```
Close  #fn
End
```

تمرین(1): تعداد 10 نمره را با دستور inputbox خوانده و در فایلی متنی با نام Scores.txt در مسیر ذخیره پروژه، ذخیره نمایید. توجه: اگر عدد وارد نشود یا در محدوده 0 تا 20 نباشد، خطا اعلام کند.

مثال 3: برنامه ای بنویسید که هر متن وارد شده در یک TextBox را در یک فایل متنی جدید با تعیین محل دلخواه، ذخیره کند. منویی بسازید بصورت

File → New , Save, Close

حل:

۱. یک TextBox به فرم اضافه کنید و خصوصیت MultiLine آنرا True کنید تا چند خط بتوان در آن نوشت. یک کادر محاوره CommonDialog به نام Dlg نیز اضافه نمایید

۲. در قسمت General:

```
Dim  fn  as integer
```

۳. در رویداد منوی new دستور زیر را بنویسید

```
fn = FreeFile()
Dlg.Filter = "Text Files(*.txt)|*.txt "
Dlg.ShowSave
If  Dlg.FileName <> ""  then
   Open  Dlg.FileName  for  Output  as  #fn
Else
    end
End if
Text1.text=""
```

۴. در رویداد منوی Save دستور زیر را بنویسید

```
Print  #fn , Text1.Text ;
```

۵. در رویداد منوی Close دستور زیر را بنویسید

```
Close  #fn
end
```

تمرین: برنامه ای بنویسید که در یک فایل متنی، نخست اعداد مضرب 5 کوچکتر از 1000 (در دکمه cmd1)و سپس اعداد مضرب 10 کوچکتر از 1000 (در دکمه cmd2)را به دنبال آن ذخیره کند.

دستور Write: فرمان مناسب تر دیگری برای نوشتن در فایل است، ویژگیها و تفاوت آن با دستور print عبارتند از:

- جداسازی داده ها با کاما (,)
- قرار دادن مقدار رشته ای در گیومه (" ")
- استفاده از # برای نوشتن تاریخ
- نوشتن مقادیر منطقی (boolean) بصورت #TRUE# , #FALSE#
- نوشتن داده خالی (Null) بصورت #NULL#
- نوشتن کد خطا (Error) بصورت #ErrorCode# که ErrorCode کد خطای ایجاد شده است

مثال: در یک رویداد دکمه دستورات زیر را بنویسید:

```
Dim fn As Integer
Dim var As Variant
fn = FreeFile
Open App.Path & "\Test.txt" For Output As #fn
var = Null
Write #fn, var, 5 < 8, "test text"
Close
```

محتویات فایل بصورت زیر خواهد بود:

```
#NULL# , #TRUE# , "test text"
```

تمرین: مثال 1و2و3 را با دستور write به جای print بازنویسی کنید

دستور Input: فرمانی است برای خواندن داده نوشته شده در فایل و بصورت زیر استفاده می گردد
Input #fn, لیست متغیرها → متناظر با متغیرها داده ها را خوانده و در آنها قرار می دهد

نکته: داده های ذخیره شده در فایل توسط دستور print یا write باید با نوع متغیرهای دستور input تطابق داشته باشند.

مثال: اعداد 1 تا 50 را در فایلی ذخیره کنید و سپس از فایل خوانده و روی فرم چاپ کنید.
حل:

۱. دو دکمه Save و Open
۲. در قسمت General:

```
Dim fn as integer
Dim I as integer
```

۳. در رویداد دکمه Save دستور زیر را بنویسید

```
fn = FreeFile()
Open App.path & "\open.txt" for Output as #fn →باز کردن فایل برای نوشتن
For I =1 to 50
write #fn , i ; →نوشتن اعداد در فایل
Next I
Close #fn
```

۴. در رویداد دکمه open دستور زیر را بنویسید

```
fn = FreeFile()
Open App.path & "\open.txt" for Input as #fn →باز کردن فایل برای خواندن
For I =1 to 50
Input #fn , I →خواندن اعداد از فایل
Print I; →چاپ روی فرم
Next I
Close #fn
```

دستور Line Input: فرمانی است برای خواندن یک خط داده نوشته شده در فایل. انتهای هر خط توسط مجموعه کارکتر کد 13 و 10 یا Chr(10) & Chr(13) که معادل Enter است، تعیین می گردد. این کد در VB توسط ثابت vbCrLf مشخص میگردد. این دستور بصورت زیر استفاده می گردد:

```
Line Input #fn,متغیر رشته ای → یک خط متن یا داده را خوانده و در متغیر رشته ای قرار میدهد
```

دستور EOF : برای بررسی رسیدن به انتهای فایل(End Of File) بکار می رود. شماره فایل مورد نظر به عنوان پارامتر به آن داده می شود. اگر نتیجه این تابع True باشد، به انتهای فایل رسیده ایم. این تابع اغلب برای خواندن کل فایل و همراه حلقه While استفاده میگردد در قالب زیر:

```
Do While Not EOF(fn) →تا زمانیکه به آخر فایل نرسیدی
  Line Input #fn, S →یک خط از فایل بخوان و قرار بده در s
  ..... →دستورات دیگر
Loop
```

مثال: برنامه ای بنویسید که یک فایل متنی دلخواه را باز کند و محتویات آنرا در یک ListBox نشان دهد.

حل:

۱. یک ListBox ، دکمه Open و یک کادر محاوره CommonDialog به نام Dlg اضافه نمایید
۲. در رویداد Open دستورات زیر را بنویسید

```
Dim  fn  as  integer
Dim S as String
fn = FreeFile()
Dlg.Filter = "Text Files(*.txt)|*.txt|Log Files(*.log)|*.log"
Dlg.ShowOpen
If Dlg.FileName <> ""  then
   Open  Dlg.FileName  for  Input  as  #fn
Else
   end
End if
List1.clear
Do While NOT EOF(fn)
        Line Input  #fn,s
        List1.AddItem  s
Loop
Close  #fn
```

کار با پرونده های تصادفی:

نوعی فایل که در هر نقطه از آن(بدون رعایت ترتیب) می توان نوشت یا از آن خواند. اما در فایل ترتیبی مثلاً برای خواندن رکورد ششم، باید 5 رکورد قبلی را خوانده و بعد به رکورد ششم دسترسی داشت. در حالی که در فایل تصادفی، مستقیماً رکورد ششم قابل دسترسی است. قالب داده نوشته شده در فایل تصادفی، توسط کاربر تعیین می شود.

نوع داده تعریف شده توسط کاربر(UDT: User-defined DataType) و گاهی به نام رکورد نیز شناخته میشود. تعریف آن بصورت زیر است

```
Private | public  Type   نام نوع داده
  نام جزء1  AS  نوع داده
  نام جزء2  AS  نوع داده
  ...
End  Type
```

مثال: تعریف رکوردی برای ذخیره مشخصات یک دانش آموز:

```
Private  Type   student      → نام رکورد
     ID  as  long            → شماره دانش آموزی
     Fname  as  string       → نام
     Lname  as  string       → نام خانوادگی
     Avg  as  single         → معدل
     Address  as  string     → آدرس
End  Type
```

***نکته**: این نوع داده تعریف شده مثل نوع داده های استاندارد (مانند Integer , String,...) در برنامه قابل استفاده است.*

برای استفاده از آن باید متغیری از آن تعریف کنیم طبق قالب زیر:

```
Dim   نام متغیر  as  نام نوع داده تعریف شده
Dim   st        as  student     → مثال
```

برای دسترسی به هر یک از اجزای آن باید از قالب زیر کمک گرفت:

نام جزء • نام متغیر از نوع رکورد مورد نظر

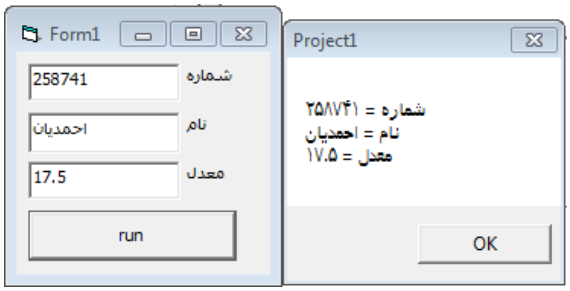
St • Fname → دسترسی به جزء نام دانش آموز

St • Avg → دسترسی به جزء معدل دانش آموز

توجه کنید که student.Fname غلط است؛ چون نوع داده نمی تواند مستقیماً بکار رود و باید با متغیر آن کار کرد.

مثال: مشخصات یک دانش آموز را خوانده و در متغیری از نوع رکورد ذخیره کرده سپس آنرا در یک کادر پیام MsgBox نمایش دهید.

```
Private Type student → general تعریف رکورد دانش آموز در قسمت
        ID as long              →شماره دانش آموز
        name as string * 30     →نام دانش آموز
      Avg as single             → معدل
End Type
Private Sub Command1_Click()→ run
   Dim st As student            →تعریف متغیر ازنوع رکورد دانش آموز
   st.ID = Val(Text1.Text)      →ذخیره شماره
   st.name = Text2.Text         →ذخیره نام
   st.Avg = Val(Text3.Text)     →ذخیره معدل
   MsgBox "شماره = " & st.ID & vbCrLf &  "نام = " & st.name & vbCrLf & "معدل = " & st.Avg
End Sub
```



توجه: vbCrLF باعث می گردد که هر بخش در خط جدید نوشته شود.

تمرین: *رکوردی برای ذخیره مشخصات یک تیم فوتبال تعریف کنید(نام تیم، تعداد بازی، تعداد برد، تعداد مساوی، تعداد باخت،گل زده، گل خورده، امتیاز)؛ سپس مشخصات را خوانده و در یک کادر پیام Msgbox نمایش دهید.*

کار با فایلهای تصادفی

- باز کردن فایل:

```
Dim fn as integer
Fn = FreeFile
Open "نام و مسیر فایل" for Random as #fn len= طول رکورد
```

- نوشتن در فایل تصادفی با دستور Put:

```
Put #fn,[شماره رکورد],متغیر ← شماره رکورد اختیاری است
```

- خواندن از فایل تصادفی با دستور Get:

```
Get #fn,[شماره رکورد],متغیر ← شماره رکورد اختیاری است
```

نکته: اگرشماره رکورد نوشته نشود، عملیات روی رکورد بعد از رکورد فعلی انجام می گیرد.

- رفتن به محل خاص از فایل با دستور Seek:

```
Seek #fn, شماره رکورد
```

توجه: طول رکورد(مجموع اندازه فیلدها) با دستور زیر بدست می آید:

```
Len( نام متغیر از نوع رکورد)
Len(st)  → st طول رکورد متغیر= 4(long)+30(string)+ 4(single)= 38
```

مثال: برنامه ای طراحی کنید که اطلاعات دانش آموزان را در قالب یک رکورد در فایل تصادفی ذخیره کرده و لیست دانش آموزان را بصورت جدولی نمایش دهد.

۱ - یک ماژول به پروژه اضافه نمایید(Project→Add Module) و در آن دستورات زیر را بنویسید

```
→ module دستورات داخل
Option Explicit
Public Type student   →تعریف رکورد شامل شماره، نام و معدل
    ID As Long
    name As String * 30
    Avg As Single
End Type
Public fn As Integer      → شماره فایل
Public st As student      → متغیر رکورد
Public FilePath As String →مسیر ذخیره فایل
Public RecNo As Integer   → شماره رکورد
```

۲ - در فرم شماره 1، 3 عدد TextBox و دو دکمه ADD و Read را اضافه کرده و دستورات زیر را در دکمه Add بنویسید

```
Private Sub cmdAdd_Click()
        fn = FreeFile
        FilePath = App.Path & "\test.dat"      → تنظیم مسیر ذخیره فایل به محل ذخیره برنامه
        Open FilePath For Random As #fn Len = Len(st) → باز کردن فایل
        st.ID = Val(Text1.Text)                        →ذخیره مقادیر ورودی در رکورد
        st.name = Text2.Text
        st.Avg = Val(Text3.Text)
        RecNo = FileLen(FilePath) / Len(st)          →بدست آوردن تعداد رکوردهای فایل
        Seek #fn, RecNo + 1                          →رفتن به محل بعد از آخرین رکورد
        Put #fn, , st                                →ذخیره رکورد
End Sub
```


```
Private Sub cmdRead_Click()
    Form2.Show
End Sub
```

۳ - در دکمه Read تنها دستور زیر را بنویسید تا فرم دوم نشان داده شود

۴ - فرم 2 را به پروژه اضافه نموده و یک کنترل MsFlexGrid به نام Grid به فرم اضافه کنید. توجه کنید که نخست باید آنرا به ابزارها اضافه نمایید.

۵ - در رویداد Form_Activate فرم دوم دستورات زیر را بنویسید

```
Private Sub Form_Activate()
        FilePath = App.Path & "\test.dat"  →تعیین مسیر فایل در مسیر ذخیره پروژه
        fn = FreeFile
        Open FilePath For Random As #fn Len = Len(st) →باز کردن فایل
        RecNo = FileLen(FilePath) / Len(st) →بدست آوردن تعداد رکوردهای فایل
        With grid
          .Clear
          .Rows = 1    → تعداد سطرها 1
          .Cols = 4    → تعداد ستونها 4
         .ColWidth(0) = 500  → تنظیم پهنای ستونها
          .ColWidth(1) = 700
          .ColWidth(2) = 1600
          .ColWidth(3) = 700

          .Row = 0   →شروع از سطر اول
          .Col = 0
          .Text = "ردیف"    →عناوین ستونها
          .Col = 1
          .Text = "شماره"
          .Col = 2
          .Text = "نام و نام خانوادگي"
          .Col = 3
          .Text = "معدل"
          For I = 1 To RecNo →گردش حلقه به تعداد کل رکوردهای فایل
              Get #fn, , st      →خواندن رکورد بعدی
              .AddItem I         →اضافه کردن سطر جدید
              .Row = I           →تعیین شماره سطرجدید
              .Col = 1
              .Text = st.ID      →نمایش مشخصات
              .Col = 2
              .Text = st.name
              .Col = 3
              .Text = st.Avg
           Next I
           Close #fn
        End With
End Sub
```

توجه: دستور with برای خلاصه نویسی و جلوگیری از تکرار نام یک کنترل یا شیئ برای دسترسی به اجزای آن استفاده می گردد

```
MySt.ID =100                    With  MySt
MySt.Name = “hadi”      ⇒         .ID = 100
Myst.Avg = 12.5                   .Name =”hadi”
                                  .Avg = 12.5
                                End with
```

کنترلهای نمایش درایو، پوشه و فایل

کادر لیست درایو(DriveListBox): برای انتخاب یک درایو از درایوهای مختلف مثل: هارد، فلاپی، فلاش، CD و ...بصورت پیش فرض، درایو محل ذخیره پروژه انتخاب می گردد. توسط دستور زیر می توان درایو را در برنامه تغییر داد:

نام کنترل.drive =”نام درایو:\” → Drive1.drive =”c:\”

کادر لیست پوشه(DirectoryListBox): برای انتخاب یک پوشه از پوشه های موجود، بصورت پیش فرض، پوشه محل ذخیره پروژه انتخاب می گردد. مشخصه متداول این کنترل Path است که توسط آن مسیر پوشه انتخابی تعیین می گردد. توسط دستور زیر می توان پوشه را در برنامه تغییر داد:

نام کنترل.path =”نام و مسیر پوشه” → Dir1.path =”D:\MyFolder”

کادر لیست فایل(FileListBox): برای انتخاب ونمایش فایلهای موجود، بصورت پیش فرض، لیست فایلهای موجود در محل ذخیره پروژه را نشان می دهد. مشخصه های متداول این کنترل عبارتند از:

- Path: مسیر فایل انتخاب شده
- FileName: نام فایل انتخاب شده
- Pattern: الگو و نوع فایلهای قابل نمایش که مثل فیلتر عمل می کند.
- MultiSelect : امکان انتخاب چندین فایل بصورت همزمان

توجه: برای اینکه مسیر پوشه به مسیر درایو انتخاب شده تغییر یابد باید در رویداد Drive_Change دستورات زیر را نوشت:

```
Private Sub Drive1_Change()
        Dir1.Path = Drive1.Drive
End Sub
```

و برای اینکه لیست فایلهای پوشه انتخاب شده در کادر لیست فایل نمایش داده شود، باید در رویداد Dir_Change دستورات زیر را نوشت:

```
Private Sub Dir1_Change()
   File1.Path = Dir1.Path
End Sub
```

مثال: *برنامه ای طراحی کنید که بتوان درایو و پوشه دلخواه را انتخاب کرد و فقط لیست فایلهای تصویری را مشاهده نمود و با کلیک روی نام هر فایل، تصویر مربوطه در کنترل Image نمایش یابد.*

۱. *کنترلهای درایو، پوشه و فایل و Image و Label را روی فرم قرار دهید.*

۲. در رویداد Drive_Change مربوط به درایو دستورات زیر را بنویسید:

```
Private Sub Drive1_Change()
        Dir1.Path = Drive1.Drive
End Sub
```

۳. در رویداد Dir_Change پوشه دستورات زیر را بنویسید:

```
Private Sub Dir1_Change()
   File1.Path = Dir1.Path
End Sub
```

۴. *در رویداد Form_Load دستور زیر را بنویسید:*

```
File1.Pattern = "*.jpg;*.bmp"      → تعیین نوع فایلهای قابل انتخاب
Image1.Stretch = true  → تا تصویر به اندازه کادر تصویری در آید
```

۵. *در رویداد کلیک کنترل فایل دستورات زیر را بنویسید :*

```
Dim s As String
s = File1.Path & "\" & File1.FileName  → تعیین مسیر و نام فایل انتخاب شده
Image1.Picture = LoadPicture(s)  → نمایش فایل تصویری
Label1.Caption = s  → نمایش نام و مسیر فایل تصویری
```

تمرین: *الگوی فایلهای قابل انتخاب را به متنی وتصویری تغییر دهید. اگر فایل انتخابی تصویری بود در Image و اگر متنی بود محتوای فایل را در یک Listbox به کمک دستورات فایل متنی نمایش دهید.*

راهنمایی: *پسوند فایل انتخاب شده را از دستور زیر بدست آورده و بر اساس آن عمل کنید.*

```
Dim  ext as string
Ext = right(File1.FileName,3 )  → استخراج سه حرف آخر برای پسوند
```

دستورهای کار با فایلها

توسط دستورهای زیر می توان با فایلها مستقیماً کار کرد:

- ChDrive: برای تغییر درایو جاری در قالب زیر

```
ChDrive "نام درایو"  → ChDrive  "d:\"
```

- ChDir: برای تغییر پوشه جاری در قالب زیر

```
ChDir "نام پوشه"  → ChDir  "\MyFolder"
```

- Kill: فایلی(یا فایلهایی) را حذف می کند

```
Kill "نام فایل"  → Kill  "d:\test.dat"
```

- MkDir: پوشه ای را ایجاد می کند

```
MkDir "نام پوشه"  → MkDir  "\MyFolder"
```

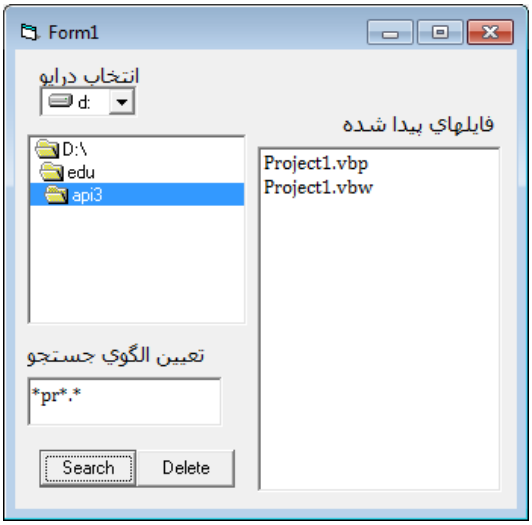
- RmDir: پوشه ای را حذف می کند که حتماً باید خالی باشد(هیچ فایلی در آن نباشد) و گرنه خطا تولید می شود

```
RmDir "نام پوشه"  → RmDir  "\MyFolder"
```

تمرین: در درایو D: پوشه ای با نام MyFolder بسازیدو دو فایل متنی دلخواه در آن قرار داده وسپس آنها را با دستور مربوطه حذف کنید.(قبل از اقدام به حذف، پیام تأیید نمایش دهید)

- Dir: برای جستجوی فایلها استفاده می شود، اگر فایل مورد نظر موجود باشد، نام فایل را برگشت می دهد و اگر موجود نباشد، نتیجه خالی بر می گرداند. در مثال زیر در درایو D:\ فایل Test.dat را جستجو می کند:

```
If (Dir("D:\test.dat") = " test.dat ") then
        Msgbox "Found"
Else
        MsgBox "Not Found"
Endif
```



مثال: برنامه ای طراحی کنید که در درایو های انتخابی فایلهایی که با الگوی خاصی شروع می شوند را پیدا کرده و در یک ListBox نشان دهد و با کلیک روی هر یک در صورت تأیید کاربر، آنرا حذف کند.

1) یک کادر DriveListBox, DirListbox و یک ListBox و دکمه جستجو(Search) و حذف(delete) و نیز کادر متن Textبه فرم اضافه نمایید.(ارتباط درایو و پوشه را برقرار کنید)

```
Private Sub Search_Click() →دکمه جستجو
        Dim s As String
        List1.Clear
        ChDrive  Drive1.Drive →تغییر درایو جاری
        chDir  Dir1.Path  →باز کردن پوشه انتخابی
        s = Dir(Text1.Text) →Text1 جستجوی فایل با الگوی متن واردشده در
        Do While  s <> ""
           List1.AddItem  s  →اضافه کردن نام فایل به لیست
           s = Dir                →جستجوی بعدی
        Loop
End sub
Private Sub Delete_Click()  →دکمه حذف
        Dim r As Integer, idx As Integer
        idx = List1.ListIndex → شماره محل نام فایل انتخابی
        If  idx > -1 Then  →اگر فایلی انتخاب شده است
            r = MsgBox("Do You Want to Delete File?", vbYesNo + vbQuestion)
            If  (r = vbYes)  Then
                Kill  List1.List(idx)  →  حذف فایل
                List1.RemoveItem idx  →حذف نام فایل از لیست
            End If
        End If
End sub
```

فصل سوم- مفاهیم شیئ گرایی

مقدمه: برای کنترل برنامه به جای برنامه اصلی از مفاهیمی چون کلاس، مشخصه و شیئ استفاده می گردد.

مفهوم شیئ(object): هرچیز مادی که دارای مشخصه هایی است و در برابر رویدادهای محیط از خود واکنش نشان دهد. جنبه های مهم که به شناخت شیئ کمک می کند عبارتند از:

- مشخصه ها(property): ویژگی های مشخص کننده حالت فعلی شیئ را بیان می کند. مثلاً سرعت ماشین، قد یک نفر و ...
- رفتار یا متد(method): نحوه پاسخ آن شیئ در مقابل رویدادهای محیط. مثلاً شیئ ماشین با فشار دادن پدال گاز حرکت می کند یا با چرخش فرمان، تغییر جهت می دهد.
- روابط(interface): نشان دهنده ارتباط هر شیئ با اشیاء دیگر است. مثلاً ارتباط شیئ ماشین با انسان

مفهوم کلاس(Class): مجموعه همه اشیاء هم نوع را کلاس می گویند. البته همه اشیاء هم نوع مشخصات و رفتار یکسانی ندارند. مثلاً همه ماژیک ها رنگشان مثل هم نیست و مشابه هم نمی نویسند.

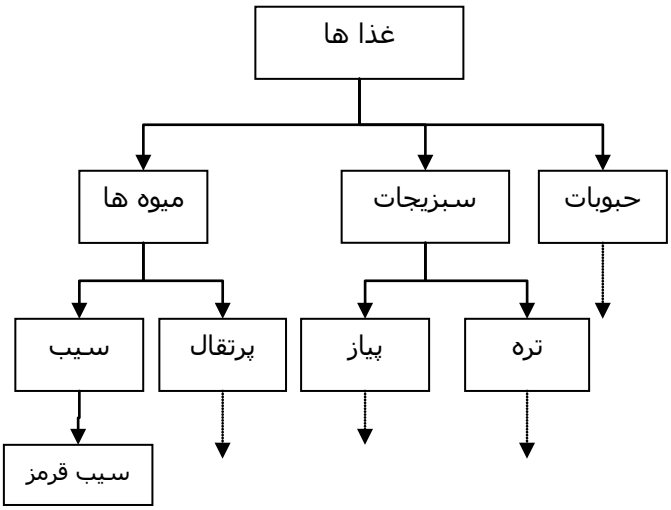
تفاوت شیئ و کلاس : کلاسها، نقشه ايجاد و قالب اشياء را مشخص مي‌كنند (مثل نقشه یک ساختمان) و اشياء نمونه هاي واقعي ساخته شده از كلاسها هستند (مثل ساختمان های ساخته شده طبق یک نقشه).

برنامه نویسی شیئ گرا(Object-Oriented Programing): شامل تعدادی شیئ با مشخصه ها و متدهای متفاوت است به نحوی که روابط خاصی بین آنها برقرار می باشد. متد؛ مجموعه ای از دستورات برنامه می باشد که در صورت بروز رویدادی خاص اجرا می شوند. این مجموعه از دستورات رفتار هر شیئ را در برابر رویدادها مشخص می نماید.

مفاهیم و مشترکات زبانهای برنامه نویسی شیئ گرا

الف) کپسوله سازی(Encapsulation): *هر برنامه از دو عنصر اصلی دستورات(کد برنامه) و داده ها تشکیل می شود. كپسوله سازي يا نهان سازی، يک مكانيزم برنامه‌نويسي است كه كد و داده را با هم در يک جا قرار داده و هر دو را از استفاده نادرست و تداخل خارجي ايمن نگه مي‌دارد.* نهان سازي فرايندي را مشخص مي‌نمايد كه طي آن، برنامه بصورت بسته‌اي(Pack) درآمده و اشياء بدون اطلاع از چگونگی عملکرد یکدیگر با هم ارتباط برقرار می کنند. مثلاً ماشین دارای مشخصات چرخ، فرمان، بدنه، موتور و .... است و متدهای حرکت، روشن و خاموش کردن و جلو یا عقب بردن و ... را دارد. برخی مشخصات فنی و اتفاقات داخل موتور هنگام رانندگی از دید راننده پنهان است. یعنی نیازی نیست راننده از همه جزئیات آگاه باشد و این یعنی کپسوله سازی.

ب) چند ریختی(Polymorphism): *چند ريختي كميتي است كه به يک رابط امكان مي‌دهد تا براي يک كلاس عمومي‌از عمليات مورد استفاده قرار گيرد.* به طور كلي مفهوم چند ريختي اغلب توسط عبارت " يک رابط. چندين روش " بيان مي‌شود. مثال ساده‌اي براي چندريختي، فرمان در وسايل نقليه‌است. فرمان در همه آنها جهت چرخش و تغيير جهت حركت است، اما شكل ظاهري، اندازه و طرح آن در وسايل مختلف(موتور سيكلت، كشتي، ماشين و ...)، متفاوت است. مثال دیگر: میوه ها همگی دارای ویژگی های مشترکی(آبدار و ...) هستند ولی هر کدام طعم، شکل، نحوه رسیدن و ... خاص خود را دارند.



ج) وراثت(Inheritance): *وراثت، عملي است كه يک شي مي‌تواند مشخصه هاي شي ديگري را بدست آورد.* به همين دليل از مفهوم دسته بندي سلسله مراتبي پشتيباني مي‌كند. به عنوان مثال سيب قرمز بخشي از دسته بندي سيب كه آن هم بخشي از كلاس ميوه هاست كه آن هم در كلاس بزرگتري به نام غذا قرار دارد. كلاس غذا داراي مشخصات اصلي (خوراكي. پروتئين و غيره) است كه به طور منطقي به زير كلاس هاي غذا اعمال مي‌شود. علاوي بر اين مشخصات كلاس ميوه داراي مشخصه هاي (آبدار، شيرين و غيره) است كه آن را از ساير غذا ها متمايز مي‌كند. كلاس سيب نيز مشخصه هاي خاصي را براي يک سيب تعريف مي‌كند كه عبارتند از : رسيدن روي درخت و غيره. يک سيب قرمز تمام خصوصيات كلاسهاي بالاتر را به ارث مي‌برد و فقط مشخصه هايي كه منحصر به فرد هستند را تعريف خواهيم كرد.

وراثت با خانواده‌اي از كلاسها و ارتباط بين آنها سروكار دارد. اين امكان به ما اجازه مي‌دهد كه مشتركات را در يک كلاس سطح بالا قرار دهيم و در زير كلاسهاي رده پاين تر از آنها فاكتور بگيريم. البته هر كلاس علاوه بر خصوصيات مشترک ارثي از سطوح بالاتر، مي‌تواند خصوصيات خاص خود را نيز داشته باشد. مثلاً تعداد پاها در همه خزندگان با هزارپا يكي نيست!

پیاده سازی کلاسها : از دید پیاده سازی هر کلاس شامل مجموعه ای از داده ها و متدهاست که داده ها همان متغیرهای کلاس و متدها؛ روالها یا توابع هستند. تعریف آن در برنامه مشابه UDT یا داده تعریف شده توسط کاربر است ولی علاوه بر داده ها، شامل روالها و توابع نیز هست.
برای نوشتن کلاسها در برنامه باید یک ماژول کلاس به برنامه اضافه نماییم.

Project → add Class Module

ماژولهای کلاس بصورت فایلهایی با پسوند .cls ذخیره می شوند و باید نامی برای کلاس خود در قسمت properties وارد نمایید.

تفاوت ماژول کلاس با ماژولهای معمولی(استاندارد) در برنامه:

- میدان دید داده های تعریف شده در ماژول معمولی تمام برنامه است یعنی تا آخر برنامه قابل دسترسی و استفاده هستند. اما داده های ماژول کلاس فقط برای هر شیئ ایجاد شده از نوع آن وجود دارند و با اتمام کار شیئ دیگر وجود ندارند.
- متغیرهای تعریف شده بصورت public در ماژول معمولی، در هر جایی ار برنامه قابل روئیت و استفاده هستند. اما، داده های Public در ماژول کلاس، فقط توسط اشیاء ایجاد شده از نوع کلاس قابل دسترسی هستند.

پیاده سازی اجزای کلاس:

الف) داده ها: داخل ماژول کلاس، داده ها مثل متغیرهای معمولی بصورت Public یا Private تعریف می شوند:

Private / public نام داده as نوع داده
مثال→ Private MyData as integer

داده های Private فقط توسط متدهای کلاس قابل دسترسی هستند ولی داده های Public علاوه بر متدهای کلاس، در سایر زیربرنامه های موجود در برنامه قابل دسترسی هستند.
چون داده های Private فقط داخل ماژول قابل استفاده هستند، برای دسترسی به آنها در برنامه توسط اشیاء باید مشخصه(Property) تعریف کرد. برای این کار از منوی زیر استفاده نمایید:

Tools → Add Procedure→public به صورت Property وارد کردن نام و انتخاب گزینه

با انجام این کار دو روال مشخصه بصورت زیر ایجاد خواهد شد(فرض کنید نام Test را وارد کرده باشید):

Public Property Get Test() As Variant

End Property

Public Property Let Test(ByVal vNewValue As Variant)

End Property

- روال Get برای خواندن مقدار داده مورد نظر بکار می ورد. مثلاً برای برگرداندن مقدار داده MyData، دستور زیر را در آن بنویسید:

نام داده مورد نظر = نام مشخصه
Test = MyData

هرگاه شیئی بخواهد مقدار مشخصه را بخواند این روال خودبخود فراخوانی می گردد.

- روال Let برای نوشتن و مقداردهی به داده مورد نظر بکار می ورد مثلاً برای تغییر مقدار داده MyData، دستور زیر را در آن بنویسید:

مقدار پارامتر مشخصه = نام داده مورد نظر
MyData = *vNewValue*

هرگاه شیئی بخواهد مقدار مشخصه را تغییر دهد این روال خودبخود فراخوانی می گردد.

***نکته(1)**: باید نوع Variant را در دو روال Get و Let به نوع داده مورد نظر بصورت دستی تغییر داد:*

```
Public Property Get Test() As Integer
        Test = MyData
End Property

Public Property Let Test(ByVal vNewValue As Integer)
        MyData  = vNewValue
End Property
```

***نکته (2)**:به ازای هر داده تعریف شده در ماژول کلاس، باید جفت مشخصه Let و Get برای هر کدام تعریف نمود.*

***نکته (3)**: هرگاه داده شیئ توسط مشخصه خوانده شود، روال Get و هرگاه داده شیئ توسط مشخصه آن با دستور انتساب تغییر یابد، خودبخود روال Let آن فراخوانی می گردد*

***نکته (4)**:علاوه بر روالهای مشخصه، می توان توابع یا روالهای معمولی دیگری نیز تعریف کرد که برای پردازشهای داخل ماژول، دسترسی به داده ها و یا دستکاری آنها استفاده می شوند.*

Public Sub ShowData() → روالی معمولی برای نمایش مقدار داده در یک کادر پیام
```
  MsgBox "Data = " & CStr(Mydata)
End Sub
```

استفاده از کلاس و ایجاد اشیاء: در برنامه اصلی می توان از نوع کلاس تعریف شده در ماژول (فرض کنید نام کلاس را MyClass گذاشته اید)اشیائی را تعریف کرد. برای اینکار دو روش وجود دارد:

۱. استفاده از New بصورت زیر:

Public نام شیئ as New نام کلاس
Public MyObject as New MyClass

۲. استفاده از Set بصورت زیر:

Public نام شیئ as نام کلاس
Set نام شیئ = New نام کلاس
Public MyObject as MyClass
Set MyObject = New MyClass

بعد از ایجاد اشیاء، توسط قالب زیر می توان به اجزای Public آن دسترسی داشت:

نام جزء.نام شیئ
MyObject.Test

پس گرفتن حافظه شیئ: بعد از پایان کار شیئ می توان حافظه آنرا پس گرفت. برای این کار باید دستور زیر را بکار برد:

Set نام شیئ = Nothing

Set *MyObject* = Nothing → مثال

ایجاد رویدادها: برخی اعمال شیئ از طریق رویدادها انجام می گیرند. مشخصه ها و متدها، رابط های درونی هستند و رویدادها، رابطهای بیرونی می باشند. ماژولهای کلاس می توانند دو رویداد درونی شیئ را بصورت زیر تعریف کنند:

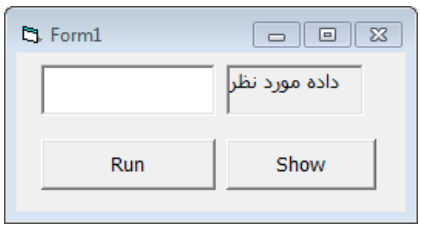
- **Initialize**: هنگامی که ایجاد نمونه ای از کلاس (تعریف شیئ) و قبل از مقدار دهی به شیئ، روی می دهد و اغلب برای مقدار دهی اولیه به داده های کلاس استفاده می گردد.

```
Private Sub Class_Initialize()
        MyData = 0
End Sub
```

- **Terminate:** هنگامی که کار شیئ تمام شده و حافظه آن پس گرفته می شود، این رویداد فعال می گردد. این رویداد اغلب برای ذخیره اطلاعات، unloadکردن فرمها و یا اجرای دستورات بعد از کار شیئ(پاک سازی) استفاده می گردد.

```
Private Sub Class_Terminate()
        MsgBox "GoodBye!!"
End Sub
```

مثال 1 :



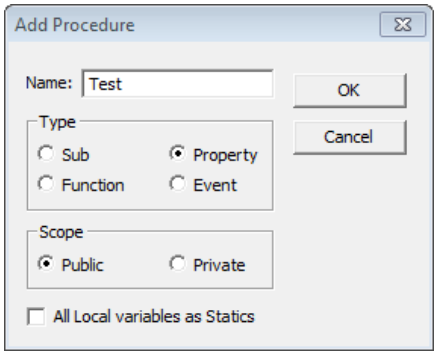
۱. یک دوکمه Run, Show و یک TextBox روی فرم اصلی قرار دهید.
۲. یک ماژول کلاس به برنامه اضافه نموده و نام آنرا MyClass بگذارید و در آن دستورات زیر را بنویسید:

Project → Add ClassModule

Private MyData As Integer → Module داده کلاس داخل

Tools → Add Procedure → Property , Name: Test, Public

سپس نوع Variant را به Integer یعنی نوع داده کلاس تغییر دهید.




```
Public Property Get Test() As Integer  →روال مشخصه برای دسترسی به داده
    Test = MyData
End Property

Public Property Let Test(ByVal vNewValue As Integer)  →  روال مشخصه برای ذخیره مقدار در آن
    MyData = vNewValue
End Property
```

دو روال زیر را بصورت دستی به کلاس اضافه کنید که هنگام ایجاد و پایان کار کلاس خودبخود فراخوانی می شوند(نامگذاری آنها را دقیقاً رعایت کنید):

```
Private Sub Class_Initialize()  →رویداد مقدار گذاری اولیه داده های شیئ
        MyData = 0
End Sub

Private Sub Class_Terminate()  →رویداد اتمام کار شیئ
        MsgBox "GoodBye!!"
End Sub
```

۳. در بخش General شیئی از کلاس تعریف کنید: `Public MyObject As MyClass`
۴. در رویداد Form_Load دستور رویرو را بنویسید: `Set MyObject = New MyClass`
۵. در رویداد دکمه Run دستورات زیر بنویسید:

```
MyObject.Test = Val(Text1.Text) →Test ذخیره مقدار در داده های کلاس توسط مشخصه
MsgBox "Data Saved !"
```

دقت کنید، چون مشخصه Test توسط انتساب مقداری به آن داده شده است، خودبخود روالLet آن فراخوانی می گردد.

۶. در رویداد دکمه Show دستور زیر را بنویسید:

MsgBox "Data = " & CStr(MyObject.Test) ←*خواندن مقدار داده کلاس توسط مشخصه Test*

دقت کنید، چون مشخصه Test فقط خوانده شده است ، خودبخود روال Get آن فراخوانی می گردد.

مثال 2: پیاده سازی اعمال برداشت، واریز و مشاهده موجودی یک حساب بانکی به کمک مفاهیم شیئ گرایی

۱. در ماژول کلاس به نام CBank داده ها، مشخصه ها و روال و توابع زیر را تعریف کنید

```
Private Balance As Long  →میزان موجودی
Private Deposit As Long  →مبلغ واریزی
Private Withdraw As Long  →مبلغ برداشت

Public Function ShowBalance() As Long  →تابع نمایش موجودی
        ShowBalance = Balance
End Function

Public Property Get Varizi() As Long  →مشخصه خواندن آخرین واریز
        Varizi = Deposit
End Property

Public Property Let Varizi(ByVal vNewValue As Long)  →مشخصه نوشتن آخرین واریز
        Deposit = vNewValue
        Balance = Balance + vNewValue  → اصلاح موجودی
End Property

Public Property Get Bardasht() As Long  →مشخصه خواندن آخرین برداشت
        Bardasht = Withdraw
End Property

Public Property Let Bardasht(ByVal vNewValue As Long)  →مشخصه نوشتن آخرین برداشت
        Withdraw = vNewValue
        Balance = Balance – vNewValue → اصلاح موجودی
End Property

Private Sub Class_Initialize()  →رویداد برای مقدارگذاری اولیه
        Balance = 0
        Withdraw = 0
        Deposit = 0
End Sub

Private Sub Class_Terminate()  →رویداد اتمام کار شیئ
        MsgBox "GoodBye My Friend, See you next time!!", vbInformation
End Sub
```

۲. فرم برنامه را به شکل زیر آماده کنید

نام کنترل	تعداد	کنترل
OptVarizi, OptBardasht	2	Option Button
LblDeposit, LblWithdraw LblBalance	3	Label
مهم نیست	4	Label
TxtMoney	1	TextBox
CmdExit, cmdUpdate	2	Command button

۳. در بخش General شیئ بانک را تعریف کنید:

```
Private MyBank As Cbank
```

۴. در Form_Load شیئ را ایجاد کنید:

```
Set MyBank = New Cbank
```

۵. در دکمه خروج دستور زیر را بنویسید:

```
Set MyBank = Nothing
  End
```

۶. در دکمه انجام عملیات دستورات زیر را بنویسید:

```
Dim MyMoney As Long
  MyMoney = Val(TxtMoney.Text)
  If MyMoney <> 0 Then
        Call UpdateMoney(MyMoney)
  Else
        MsgBox "لطفاً مبلغ را وارد نمایید!!", vbExclamation
  End If
```

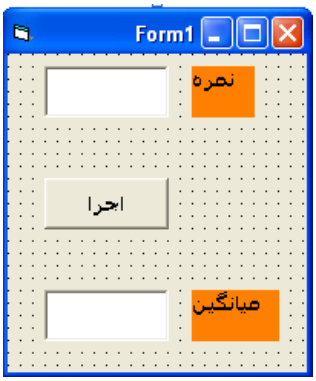
۷. در بخش General روالی برای به روزرسانی مبلغ به صورت زیر تعریف کنید:

```
Private Sub UpdateMoney(Money As Long)
  Dim msg As String
  If OptVarizi.Value Then
        MyBank.Varizi = Money
  ElseIf OptBardasht.Value Then
        If (Money <= MyBank.ShowBalance) Then
              MyBank.Bardasht = Money
        Else
              msg = "موجودی کافی نیست!!" & vbCrLF" & " مبلغ موجودی = " & CStr(MyBank.ShowBalance)
              MsgBox msg, vbCritical
        End If
  End If
  LblDeposit.Caption = CStr(MyBank.Varizi)
  LblWithdraw.Caption = CStr(MyBank.Bardasht)
  LblBalance.Caption = CStr(MyBank.ShowBalance)
End Sub
```

تمرین:

در یک ماژول کلاس، دو داده مجموع نمره(Sum) و تعداد نمره (Count) و مشخصه Majmue برای دسترسی به مجموع و مشخصه Tedad برای دسترسی به تعداد تعریف کنید و آنها را تنظیم نمایید سپس فرم زیر را طراحی و دستورات مربوطه را بنویسید

نمونه اجرا به صورت زیر است:



نمره جدید	Sum	Count	میانگین جدید Sum/Count
شروع	0	0	0
15	15=0+15	1=0+1	15/1=15
18	33=18+15	2=1+1	33/2=16.5
16	49=16+33	3=2+1	49/3=16.33

فصل چهارم- آشنایی با ADO(Active Data Objects)

همه برنامه ها به نحوی نیاز به دسترسی به داده ها دارند و یکی از روشها، یک مدل شیئی است که روشهای مختلف دسترسی به داده ها را ارائه می کند. سه نوع رابط دسترسی وجود دارند:

- ADO(Active Data Objects)
- RDO(Remote Data Objects)
- DAO(Data Access Objects)

ADO جدید ترین و قوی ترین رابط برای OLE DB است که رابطی سطح پایین برای دسترسی به تمام داده ها را فراهم می کند. ویژگیها و قابلیتهای این رابط عبارتند از:

- دسترسی به بانکهای رابطه ای و غیر رابطه ای
- پست الکترونیکی
- سیستمهای فایل
- ابزارهای مدیریت پروژه
- صفحه گسترده

مقدمه: مفاهیم پایه بانک های اطلاعاتی

موجودیت: هر چیزی که می خواهیم در مورد آن اطلاعاتی ذخیره کنیم مثل دانش آموز، معلم و ...
فیلد: یک تکه از اطلاعات در مورد یک موجودیت که آنرا توصیف می کند(صفت) مانند *نام، نام خانوادگی* و...
رکورد: مجموعه ای از فیلدها و مقادیر صفات موجودیت برای یک نمونه خاص از آن ، مثلاً اطلاعات کامل در مورد یک دانش آموز خاص مانند *علی علی زاده*
جدول: مجموعه ای از رکوردهای ذخیره شده در مورد نمونه های مختلف مثلاً *اطلاعات همه دانش آموزان سوم کامپیوتر*

استفاده از ADO برای اتصال برنامه به بانک اطلاعاتی

تعریف کنترلهای مقید به داده ها(Data-Bound): کنترلهایی هستند که دارای مشخصه DataSource بوده و قابلیت نمایش مقادیر فیلدهای جداول بانک اطلاعاتی را دارند. انواع مختلف کنترلهای قابل استفاده عبارتند از:

۱ - کنترلهایی که در هر لحظه فقط اطلاعات یک فیلد مربوط به یک رکورد را نشان می دهند:

- TextBox
- CheckBox
- ComboBox
- Label
- ListBox
- PictureBox
- Image

۲ - کنترلهایی که در هر لحظه اطلاعات یک فیلد خاص در مورد همه رکوردها را نشان می دهند:

- DataList
- DataCombo

۳ - کنترلهایی که در هر لحظه اطلاعات همه فیلدها در مورد همه رکوردها را نشان می دهند:

- DataGrid

نحوه ایجاد یک برنامه ساده برای اتصال به بانک اطلاعاتی از طریق Data Form Wizard

۱ - از منوی Add-ins گزینه Add-in manager را زده و از پنجره ظاهر شده VB 6 Data Form wizard را انتخاب و سپس کادر Loaded/unloaded را تیک و Ok را بزنید
۲ - حال دوباره از منوی Add-ins گزینه اضافه شده Data Form Wizard را انتخاب نمایید.
۳ - در اولین پنجره ظاهر شده، Next را زده و در مرحله بعد گزینه Access و سپس در ادامه روی دکمه Browse کلیک کرده و نام بانک اطلاعاتی مورد نظر را انتخاب کنید
۴ - در پنجره بعد در قسمت Form layout یکی از موارد را انتخاب کنید(گزینه اول تک رکوردی و گزینه دوم بشکل جدولی است و بقیه فعلاً مورد بحث ما نیستند) . در قسمت راست Binding Type گزینه اول را در نظر بگیرید. در کادر بالا نیز می توانید نامی برای فرم وارد کنید

۵ - در پنجره بعد در قسمت Record Source نام جدول مورد نظر و در قسمت Available Fields فیلدهای خود را انتخاب کنید.
۶ - در پنجره بعد می توانید دکمه های کار با داده ها را فعال یا غیرفعال نمایید. با زدن دکمه Finish کار تمام است و یک فرم به پروژه اضافه میگردد.
۷ - برای مشاهده نتیجه باید فرم جدید را به عنوان Startup از منوی Project→ properties تنظیم کنید.

نکته مهم: VB6 فقط با بانک اطلاعاتی ایجاد شده در Access 97 سازگاری دارد و باید بانک اطلاعاتی طراحی شده خود را به آن تبدیل کنیم. Access 2007 فقط قابلیت تبدیل به 2000 را دارد! پس باید آنرا به کمک Office XP تبدیل نماییم. البته دو بانک نمونه برای برنامه نویسی در مسیر نصب VB98 وجود دارند:
C:\Program Files\Microsoft Visual Studio\VB98

- *BIBLIO: اطلاعات کتاب و نویسنده و ناشر و ...*
- *NWIND: شرکت نمونه، مشتری، سفارش و ...*

توجه: *می توان در محیط VB نیز از طریق منوی Add-ins→Visual data manager یک بانک اطلاعاتی Access طراحی نمود.*

مراحل ایجاد یک برنامه برای اتصال به بانک اطلاعاتی

۱ - پوشه ای برای ذخیره برنامه در مسیری دلخواه ایجاد کنید و فایل بانک اطلاعاتی خود را در پوشه DB قرار دهید
۲ - پروژه جدید در VB و ذخیره آن در مسیر فوق
۳ - استفاده از کنترل ADO Data برای اتصال به بانک و طبق مسیر زیر به ابزار ها و فرم اضافه می کنیم.
Project→Components→ MicroSoft ADO Data 6.0
۴ - تنظیمات لازم: روی ADO کلیک راست کرده و Property را انتخاب می کنیم. در پنجره مربوطه و در زبانه General به سه روش می توان به بانک وصل شد. حال به معرفی روش دوم می پردازیم.
نحوه ایجاد ODBC Data source به صورت زیر است:
Control panel→ Administrative Tools→ Data source(ODBC)→ System DSN → ADD
→ Driver do Microsoft Access(*.mdb) →finish
در پنجره بعد با زدن دکمه Select نام بانک اطلاعاتی خود را از مسیر مورد نظر انتخاب کرده و بعد از وارد کردن نامی برای آن در کادر Data Source name روی OK کلیک می کنیم.
۵ - بعد از ایجاد ODBC در لیست کشویی ظاهر شده برای ADO نام آنرا انتخاب می کنیم.
۶ - در زبانه RecordSource در قسمت Command Type گزینه 2-adCmdTable و در قسمت Table نام جدول مورد نظر خود را انتخاب می کنیم.
۷ - کنترلهای نمایش داده های خود را به فرم اضافه می کنیم. مثلاً Label برای نمایش عنوان فیلد و Text برای نمایش محتوای فیلدها. دو خصوصیت TextBox زیر را تنظیم می کنیم:
DataSource : نام کنترل ADO مثل Adodc1
DataField : نام فیلد مربوطه از جدول انتخابی مرحله 6

دکمه های روی کنترل ADO
▶: رفتن به رکورد بعدی
◀: رفتن به رکورد قبلی
|▶: رفتن به آخرین رکورد
◀|: رفتن به اولین رکورد

روش دیگر تنظیمات ADO برای اتصال به بانک اطلاعاتی به کمک رشته اتصال

1. کلیک راست ADO data → Property → General → Use Connection String→ Build

در پنجره Data Link Provider:

2. Provider → Microsoft Jet 4.0 OLE DB Provider
3. Connection → ... کلیک روی دکمه → انتخاب بانک اطلاعاتی

برای اطمینان از اتصال، دکمه Test Connection را بزنید تا پیام موفقیت دریافت نمایید. سپس در زبانه RecordSource از ADO در قسمت Command Type گزینه 2-adCmdTable و در قسمت Table نام جدول مورد نظر خود را انتخاب می کنیم.

نکته: تنظیمات رشته اتصال را می توان به کمک برنامه نویسی نیز انجام داد. که در این صورت برنامه شما به راحتی روی هر کامپیوتری بدون نیاز به تنظیمات مجدد اجرا خواهد شد. برای اینکار مطابق مراحل زیر عمل کنید.

۱. پوشه ای برای ذخیره برنامه در نظر بگیرید.
۲. فایل بانک اطلاعاتی خود را در پوشه ای به نام DB در پوشه مسیر ذخیره برنامه قرار دهید.

فرض کنید نام بانک اطلاعاتی School.mdb و نام جدول مورد نظر Student و دارای فیلدهای زیر باشد که در محیط VB در برنامه Visual data Manager آنها را طراحی کرده اید.

نام فیلد به انگلیسی	نام فیلد به فارسی
ID	شماره
Fname	نام
Lname	نام خانوادگی
major	رشته
avg	معدل

۳. کنترل ADO و دیگر کنترلهای لازم برای نمایش فیلدهای جدول مورد نظر(Text) روی فرم قرار دهید. و مشخصه DataSource کنترلها را به نام ADO تنظیم نمایید.
۴. در رویداد Form_Load دستورات زیر را برای تنظیمات رشته اتصال بنویسید:

```
Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " & _
    "Data Source= " & App.Path & "\DB\School.mdb; Persist Security Info=False"
Adodc1.CommandType = adCmdTable
Adodc1.RecordSource = "Student"
```

Adodc1.Refresh → بازخوانی داده ها

در این مرحله می توانید نام فیلدهای مرتبط را برای هر کنترل تنظیم کنید. پس در ادامه رویداد Form_Load دستورات زیر را بنویسید.

```
Text1.DataField = "ID"          → شماره
Text2.DataField = "Fname"       → نام
Text3.DataField = "Lname"       → نام خانوادگی
Text4.DataField = "major"       → رشته
Text5.DataField = "avg"         → معدل
```

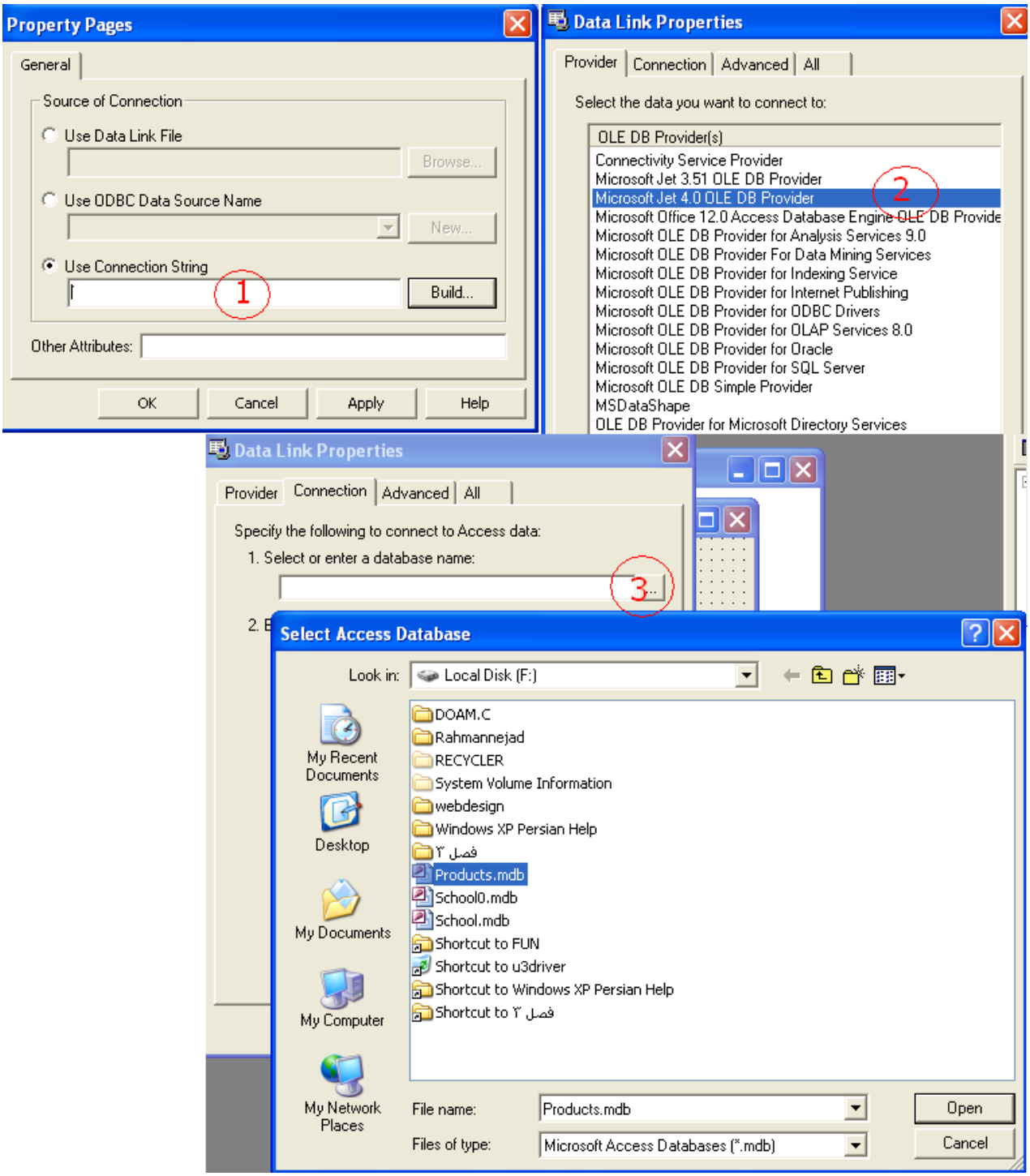
توضیح: App.Path مسیر ذخیره برنامه را تعیین می کند.

نکته: گاهی با اجرای برنامه کنترل Grid خطایی را اعلام می کند. با تنظیم خصوصیت TabStop با مقدار False ، مشکل حل می شود.

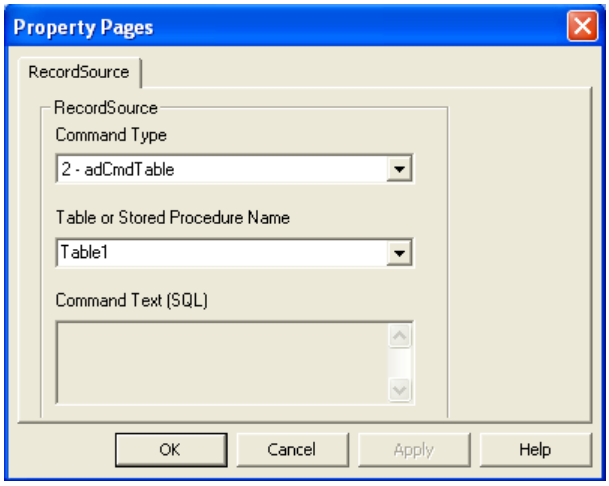
خلاصه مراحل ایجاد اتصال برنامه VB به بانک اطلاعاتی

۱. ایجاد پروژه جدید : File→new → Standard exe
۲. اضافه کردن کنترل ADO به ابزارها Project→ Components
۳. قرار دادن کنترل روی فرم و انجام تنظیمات زیر:

- رشته اتصال (ConnectionString): برای برقراری ارتباط با بانک اطلاعاتی با انتخاب آن در قسمت Properties و زدن دکمه... سپس در پنجره ظاهر شده گزینه Use Connection String را انتخاب و دکمهbuild را بزنید .مراحل شکل را دنبال کنید و با زدنOK پنجره ها را ببندید.

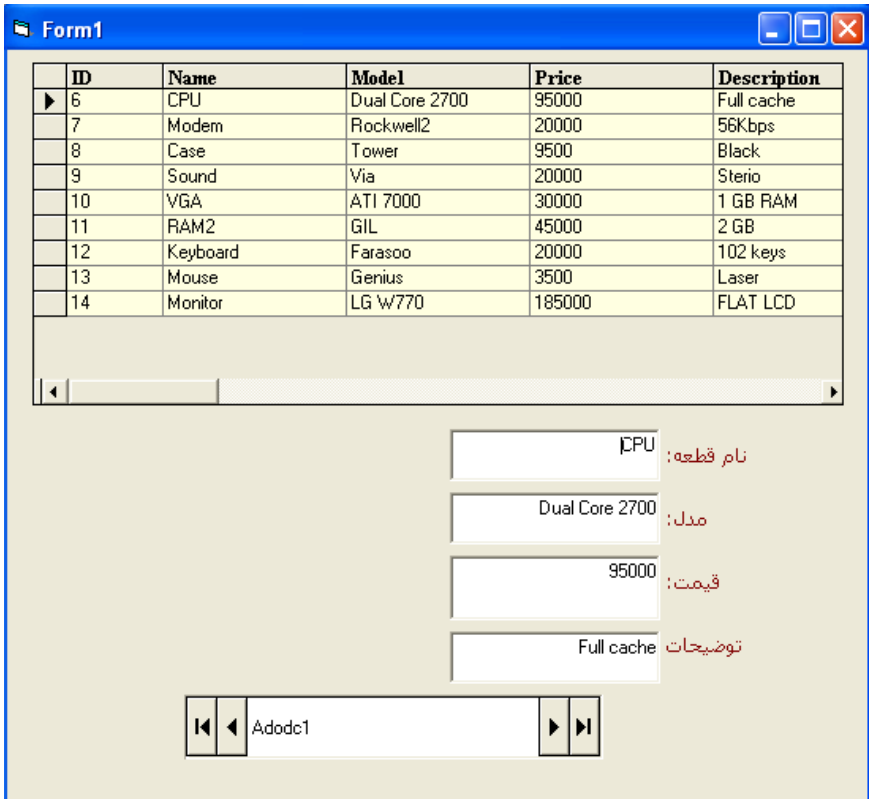


- انتخاب نام جدول در مشخصهRecordSource



۴. قرار دادن کنترلهایLabel, TextBox به تعداد فیلدهای جدول انتخاب شده روی فرم. سپس مشخصات زیر را برای TextBox ها تنظیم کنید:
   - DataSource: آنرا با نام کنترل ADO مقدار دهی کنید.
   - DataField: نام فیلد مورد نظر را به آن مربوط نمایید.

*** در روش فوق برای هر جدول بانک اطلاعاتی یک کنترلADO لازم است. و در هر لحظه فقط یک رکورد در TextBox ها قابل نمایش است. در صورتیکه بخواهید همه رکوردها را مشاهده نمایید می توانید از کنترل Data Grid استفاده کنید. این کنترل را نیز مثل مرحله2 به ابزارها اضافه نموده و سپس آنرا روی فرم قرار دهید. کافی است مشخصهDataSource آنرا به نام کنترل ADO مرتبط نمایید.



دستورات کار با رکوردها: کافی است در رویداد هر دکمه مورد نظر دستور مربوطه را نوشت:

دستور ADO.Recordset.نام

۱. Update برای ذخیره تغییرات که با حرکت بین رکوردها نیز انجام می شود (►یا ◄):

```
Adodc1.Recordset.Update
```

۲. CancelUpdate برای لغو تغییرات :

```
Adodc1.Recordset.CancelUpdate
```

۳. AddNew برای درج رکورد جدید که ثبت تغییرات با دستور Update یا حرکت بین رکوردها توسط (►: رفتن به رکورد بعدی) یا (◄: رفتن به رکورد قبلی) انجام می شود:

```
Adodc1.Recordset.AddNew
```

۴. EOF : بررسی اینکه آیا به آخرین رکورد رسیده ایم یا نه

۵. BOF : بررسی اینکه آیا رکورد جاری اولین رکورد است یا نه

اگر هر دو نتیجه TRUE برگردانند، یعنی هیچ رکوردی در Recordset نیست و خالی است.

```
if (Adodc1.Recordset.EOF = TRUE) and (Adodc1.Recordset.BOF = TRUE) then
        msgBox "NO Records!"
end if
```

۶. Filter: توسط این مشخصه شرط خاصی برای نمایش رکوردها تعیین می شود

```
Adodc1.Recordset.Filter = "شرط"
```

که شرط عبارت است از مقایسه نام یک فیلد با مقداری مشخص، مثلاً در دستور زیر فقط دانش آموزانی که معدل(Avg) آنها بیشتر از 15 است نمایش داده می شوند:

```
Adodc1.Recordset.Filter = "Avg > 15"
```

۷. Delete : برای حذف رکورد جاری یا گروهی از رکوردها بکار می رود.

```
Adodc1.Recordset.Delete  نوع حذف
```

که نوع حذف دو حالت دارد:

- adAffectCurrent: فقط رکورد جاری (که قابل مشاهده است)حذف می گردد و مقدار پیش فرض است.
- adAffectGroup: تمام رکوردهایی که با مشخصه Filter مطابقت دارند، حذف خواهند شد.

۸. MoveLast: حرکت به رکورد آخر

۹. MoveFirst: حرکت به اولین رکورد

۱۰. MoveNext : حرکت به رکورد بعدی، که البته اگر از آخرین رکورد رد شدیم باید کنترل نماییم، مثلاً به اولین رکورد برگردیم:

```
Adodc1.Recordset.MoveNext
if (Adodc1.Recordset.EOF)  then  Adodc1.Recordset. MoveFirst
```

۱۱. MovePrevious : حرکت به رکورد قبلی، که البته اگر از اولین رکورد رد شدیم باید کنترل نماییم، مثلاً به آخرین رکورد برگردیم:

```
Adodc1.Recordset. MovePrevious
if (Adodc1.Recordset.BOF)  then  Adodc1.Recordset. MoveLast
```

۱۲. Requery بازخوانی مجدد رکوردها، مثلاً بعد از اعمال فیلتر و شرط

```
Adodc1.Recordset.Filter = "Avg > 15"
Adodc1.Recordset.Requery
```

۱۳. RecordCount : اطلاع از تعداد رکوردهای موجود در Recordset

```
Adodc1.Recordset.RecordCount
```

تمرین: یک بانک اطلاعاتی جدید به نام TelBook به کمک Visual Data Manager بسازید و در پوشه DB از مسیر برنامه ذخیره کنید. یک جدول به نام Tel با مشخصات زیر در آن طراحی نمایید.

کنترل ADO و به تعداد لازم Text و Label قرار دهید و در رویداد Form_Load به کمک رشته اتصال ADO را به بانک و جدول مربوط کنید. فرم را به شکل مقابل طراحی کرده و در هر دکمه رویداد مربوطه را بنویسید

نام فیلد به انگلیسی	نام فیلد به فارسی
ID	شماره
Fname	نام
Lname	نام خانوادگی
City	شهر
Address	آدرس
Tel	تلفن
Mobile	موبایل

دستورات Form1:

دکمه درج جدید:
```
Adodc1.Recordset.AddNew
```

دکمه ذخیره
```
Adodc1.Recordset.Update
```

دکمه لغو ذخیره:
```
Adodc1.Recordset.CancelUpdate
```

دکمه حذف:
```
Dim r As Integer
r = MsgBox("Delete Current Record?", vbYesNo + vbQuestion)
If r = vbYes Then
   Adodc1.Recordset.Delete
   Adodc1.Recordset.MoveNext
   If Adodc1.Recordset.EOF Then Adodc1.Recordset.MoveLast
End If
```

دکمه فیلتر:
```
Adodc1.Recordset.Filter = "Lname= "  &  " ' "  & Text8.Text & " ' "
Adodc1.Recordset.Requery
```

دکمه لغو فیلتر
```
Adodc1.Recordset.Filter = ""
Adodc1.Recordset.Requery
```

دکمه اولین
```
Adodc1.Recordset.MoveFirst
```

دکمه آخرین
```
Adodc1.Recordset.MoveLast
```

دکمه بعدی
```
Adodc1.Recordset.MoveNext
If (Adodc1.Recordset.EOF) Then Adodc1.Recordset.MoveFirst
```

دکمه قبلی
```
Adodc1.Recordset.MovePrevious
If (Adodc1.Recordset.BOF) Then Adodc1.Recordset.MoveLast
```

دکمه لیست
```
Form2.Show
```

رویداد Form_Load()
```
Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " & _
    "Data Source= " & App.Path & "\DB\TelBook.mdb;" & _
    "Persist Security Info=False"
Adodc1.CommandType = adCmdTable
Adodc1.RecordSource = "Tel"
Adodc1.Refresh

Text1.DataField = "ID"
Text2.DataField = "Fname"
Text3.DataField = "Lname"
Text4.DataField = "City"
Text5.DataField = "Address"
Text6.DataField = "Tel"
Text7.DataField = "Mobile"
```

توجه: form2را به پروژه اضافه کنید و کنترل ADO و DataGrid را به آن اضافه نمایید.
Components→ Microsoft DataGrid Control

سپس دستورات زیر را در Form_Load آن بنویسید:

```
Dim i As Integer
Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0; " & _
    "Data Source= " & App.Path & "\DB\TelBook.mdb;Persist Security Info=False"
Adodc1.CommandType = adCmdTable
Adodc1.RecordSource = "Tel"
Adodc1.Refresh
DataGrid1.DataSource = Adodc1
DataGrid1.TabStop = False
DataGrid1.AllowUpdate = False
For i = 0 To DataGrid1.Columns.Count - 1
  DataGrid1.Columns(i).Width = 1300
Next
```

استفاده از دستور بازیابی Select در بازیابی و نمایش اطلاعات جدولهای بانک اطلاعاتی

Select نام ستونهای مورد نظر
From نام جدول مربوطه
Where شرط

- نام ستونهای مورد نظر با کاما از هم جدا می شوند. اگر همه ستونهای جدول مورد نظر باشد، به جای نوشتن نام همه آنها کافی است یک علامت * فرار دهیم.
- شرط اغلب عبارت است از مقایسه یک ستون و مقداری خاص که انتخاب سطرهایی را انجام می دهد که در شرط صدق می کنند ، اگر شرطی تعیین نشود و همه سطرها مورد نیاز باشد، عبارت Where کلاً حذف می گردد.
- می توان برای ستونها نام جدید انتخاب کرد:

نام جدید AS نام ستون

توجه: نام جدید چند کلمه ای داخل [ ] نوشته شود

مثال:

```
Select Name As [نام کالا], Model as مدل, Price As قیمت
from Parts
Where Price > 50000
```

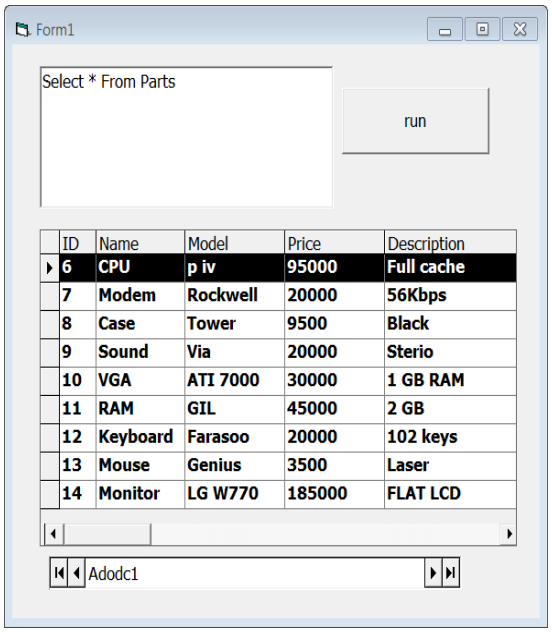
مثال:

فرض کنید بانکی به نام Productsداریم که جدولی به نام Parts برای ذخیره مشخصات قطعات کامپیوتر با مشخصات زیر در آن تعریف شده باشد.

نام فیلد به انگلیسی	نام فیلد به فارسی
ID	شماره
Pname	نام
Model	مدل
Price	قیمت
Specifications	مشخصات فنی

۱ - بانک اطلاعاتی خود را در مسیر ذخیره پروژه قرار دهید

۲ - پروژه ای طراحی کنید به شکل زیر:




```
Private Sub Command1_Click()
    Adodc1.CommandType = adCmdText
    Adodc1.RecordSource = Text1.Text
    Adodc1.Refresh
End Sub

Private Sub Form_Load()
  Adodc1.ConnectionString =
"Provider=Microsoft.Jet.OLEDB.4.0;" & _
"Data Source=" & App.Path & "\Products.mdb;" & _
"Persist Security Info=False"
End Sub
```

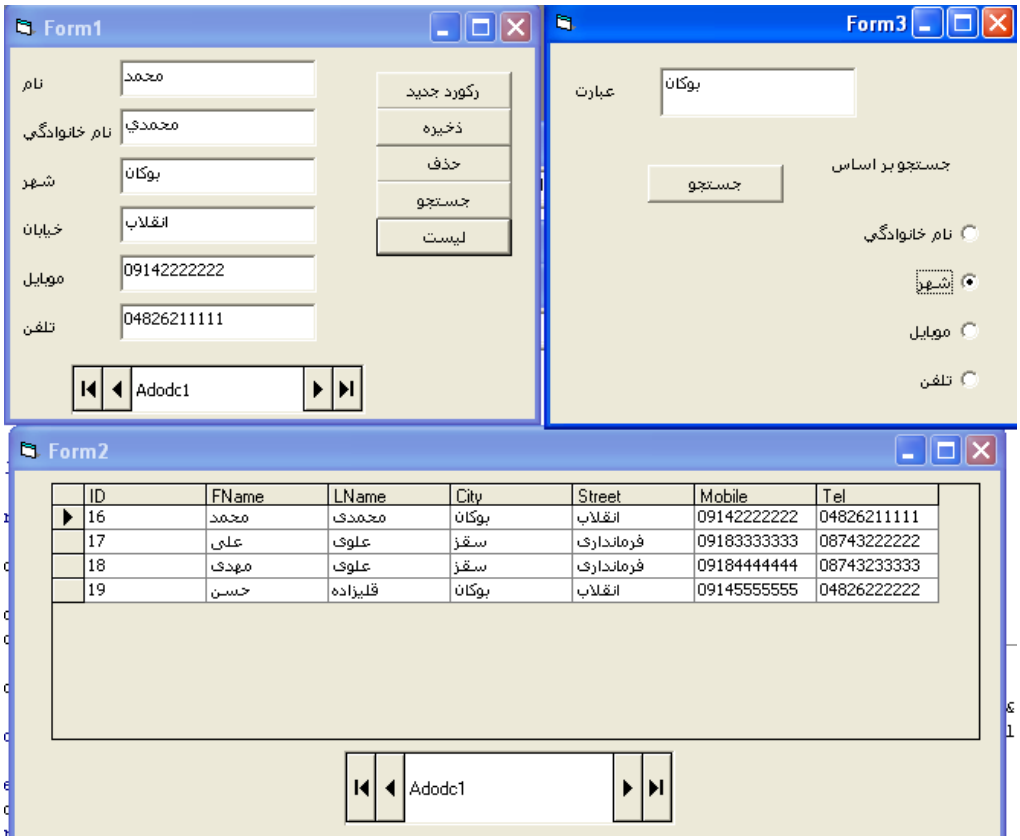
مثال:

فرض کنید بانکی به نام TelBookداریم که جدولی به نام Tel برای ذخیره دفترچه تلفن افراد با مشخصات زیر در آن تعریف شده باشد.

نام فیلد به انگلیسی	نام فیلد به فارسی
Fname	نام
Lname	نام خانوادگی
City	شهر
Street	خیابان
Mobile	موبایل
Tel	تلفن

۱ - بانک اطلاعاتی خود را در مسیر ذخیره پروژه قرار دهید

۲ - پروژه ای حاوی 3 فرم طراحی کنید به شکل زیر:



دستورات فرم اول:

```
Private Sub cmdDelete_Click()  → حذف
   Dim r As Integer
    r = MsgBox("Delete Current Record?", vbYesNo + vbQuestion)
    If r = vbYes Then
        Adodc1.Recordset.Delete
        Adodc1.Recordset.MoveNext
        If Adodc1.Recordset.EOF Then Adodc1.Recordset.MoveLast
    End If
End Sub
```

```
Private Sub cmdList_Click    () → لیست
        Form2.Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;"  & _
        " Data Source=" & App.Path & "\TelBook.mdb;Persist Security Info=False"
        Form2.Adodc1.CommandType = adCmdText
        Form2.Adodc1.RecordSource = "Select * from Tel"
        Form2.Adodc1.Refresh
        For i = 0 To Form2.DataGrid1.Columns.Count - 1
            Form2.DataGrid1.Columns(i).Width = 1200
        Next
        Form2.Show
End Sub
```

```
Private Sub CmdNew_Click()→ رکورد جدید
        Adodc1.Recordset.AddNew
End Sub
```

```
Private Sub cmdSave_Click()→ ذخیره
        Adodc1.Recordset.Update
End Sub
```

```
Private Sub CmdSearch_Click()→ جستجو
        Form3.Show
End Sub
```

```
Private Sub Form_Load() → فرم 1
        Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        " Data Source=" & App.Path & "\TelBook.mdb;Persist Security Info=False"
        Adodc1.CommandType = adCmdTable
        Adodc1.RecordSource = "Tel"
        Adodc1.Refresh
        Text1.DataField = "Fname"
        Text2.DataField = "Lname"
        Text3.DataField = "City"
        Text4.DataField = "Street"
        Text5.DataField = "Mobile"
        Text6.DataField = "Tel"
        Adodc1.Refresh
End Sub
```

دستورات فرم3:

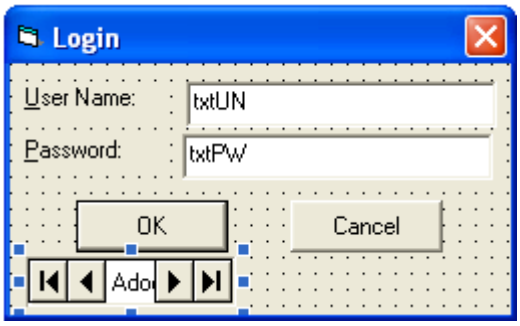
```
Private Sub CmdSearch_Click() → دکمه جستجو
        Dim s As String
        If Option1.Value Then
            s = "Lname"
        ElseIf Option2.Value Then
            s = "City"
        ElseIf Option3.Value Then
            s = "Mobile"
        ElseIf Option4.Value Then
            s = "Tel"
        End If
        Form2.Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0 ;" & _
        " Data Source=" & App.Path & "\TelBook.mdb;Persist Security Info=False"
        Form2.Adodc1.CommandType = adCmdText
        Form2.Adodc1.RecordSource = "Select * from Tel Where " & s & " Like '%" & _
                                    Text1.Text & "%' "
        Form2.Adodc1.Refresh
        For i = 0 To Form2.DataGrid1.Columns.Count - 1
            Form2.DataGrid1.Columns(i).Width = 1200
        Next
        Form2.Show
End Sub
```

اضافه کردن فرم ورود کاربر:

۱ - جدولی جدید با نام UserInfo و مشخصات زیر طراحی کنید:

نام فیلد به انگلیسی	نام فیلد به فارسی
ID	شماره
UserName	نام کاربری
PassWord	رمز عبور

۲ - فرم جدیدی با نام FrmLogin به پروژه اضافه نمایید. و دو کنترل Text به نامهای txtUN و txtPW و نیز کنترل ADO



۳ - دستورات زیر را برای آن بنویسید:

۴ - توجه کنید که فرم FrmLogin به عنوان فرم اصلی اولیه تنظیم شود:

Project→ property→ startup Object

```
Private Sub cmdOK_Click()  →OK دکمه
        Dim s As String
        If (txtPw.Text = “”) Or (txtUN.Text = “”) Then
          MsgBox "لطفاً نام و رمز را وارد کنید!", vbQuestion
          Exit Sub
        End If
        s = "(UserName = ‘” & txtUN.Text & “’ )”
        s = s & " AND (Password = “ ’ & txtPw.Text & “’ )”
        Adodc1.Recordset.Filter = s
        If Adodc1.Recordset.EOF Then
          MsgBox "نام کاربری یا رمز اشتباه است!", vbExclamation
          Exit Sub
        Else
          MsgBox "خوش آمدید"
          frmLogin.Hide
          Form1.Show
        End If
End Sub
```

```
Private Sub Form_Load()
        Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
          "Data Source=" & App.Path & "\Products.mdb;Persist Security Info=False"
        Adodc1.CommandType = adCmdTable
        Adodc1.RecordSource = "UserInfo"
        Adodc1.Refresh
End Sub
```

جستجوی رکوردها به کمک متد Find: به کمک این متد رکوردی که با معیار مورد نظر ما تطبیق دارد را یافته و مکان نما روی آن قرار می گیرد.

نام ADO.Recordset.Find معیار جستجو, 0 , جهت حرکت برای جستجو, نقطه شروع

- **معیار جستجو**: بر اساس مقایسه یک یا چند فیلد با مقادیر مورد نظر می باشد. اگر مقدار رشته ای باشد باید داخل علامت تک گیومه قرار گیرد. مثلاً Name = 'cpu' یا Price > 100
- **پارامتر جهت جستجو** : تعیین می کند که جستجو به کدام سمت انجام شود که سه مقدار دارد:
  - adSearchForward یا مقدار صفر: به جلو یا به سمت آخرین رکورد <-- عدم موفقیت با EOF
  - adSearchBackward یا مقدار 1 : به عقب یا به سمت اولین رکورد <-- عدم موفقیت با BOF

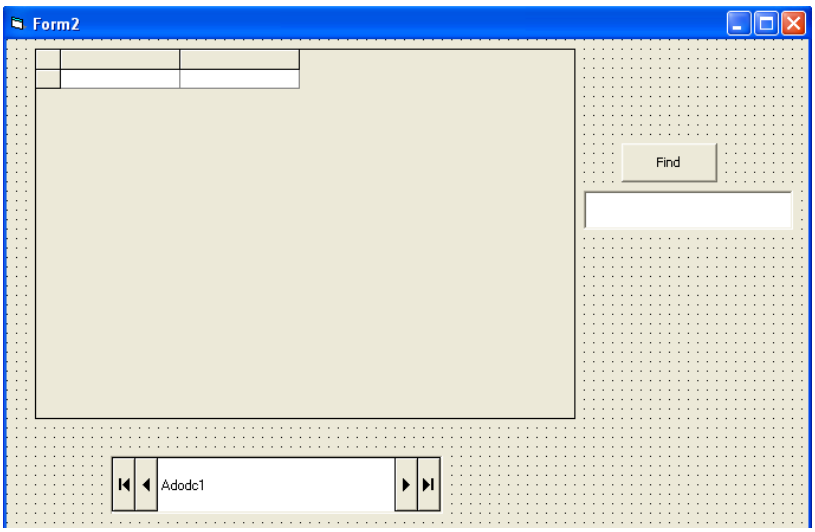
- **نقطه شروع**: محل شروع جستجو را مشخص می کند:
  - adBookMarkCurrent یا مقدار صفر: شروع از محل فعلی
  - adBookMarkFirst یا مقدار 1: شروع از محل اولین رکورد
  - adBookMarkLast یا مقدار 2: شروع از محل آخرین رکورد

برای اطمینان از پاسخ باید EOF و BOF را بررسی کرد که رکورد یافت شده است یا نه.مثلاً دستورات زیر از محل فعلی به سمت جلو جستجو را انجام داده و یافت شدن یا نشدن قطعه ای با قیمت بیش از 1000 را بررسی می کند

```
Adodc1.Recordset.Find  "Price> 1000"  0, adSearchForward, adBookMarkCurrent
If Adodc1.EOF  OR Adodc1.BOF  Then  msgbox "یافت نشد!"
```

توجه: همه پارامترها غیر از معیار جستجو اختیاری هستند، یعنی می توان آنها را ننوشت و حالت پیش فرض از نقطه فعلی به سمت جلو می باشد.

```
Adodc1.Recordset.Find  "Price> 1000"
```



مثال: فرض کنید جدول Parts از بانک Products را در اختیار دارید. فرم زیر را طراحی کرده و دستورات رشته اتصال را در رویداد Form_load نوشته و سپس معیار جستجو را در Text وارد کرده و آنرا جستجو کنید:

```
Private Sub CmdFind_Click ()  → Find دکمه
        Adodc1.Recordset.Find  Text1.Text
        If Adodc1.Recordset.EOF  OR Adodc1.Recordset.BOF  Then  msgbox "یافت نشد!"
End Sub
```

```
Private Sub Form_Load()
```

```
        Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
        " Data Source=" & App.Path & "\Products.mdb;Persist Security Info=False"
        Adodc1.CommandType = adCmdTable
        Adodc1.RecordSource = "Parts"
        Adodc1.Refresh
        For i = 0 To DataGrid1.Columns.Count - 1
            DataGrid1.Columns(i).Width = 1000
        Next
End sub
```

توجه : در صورتیکه از محل فعلی جستجو را شروع کرده و جستجو نا موفق باشد با ذخیره محل و برگشت به آن می توان حالت قبلی را برگرداند. پس رویداد جستجو را اصلاح می کنیم:

```
Dim bk as Variant
Bk = adodc1.RecordSet.BookMark → ذخیره محل فعلی
Adodc1.Recordset.Find Text1.Text
If Adodc1.Recordset.EOF OR Adodc1.Recordset.BOF Then → اگر یافت نشد
adodc1.RecordSet.BookMark = bk → بازگشت به محل قبلی
msgbox "یافت نشد!"
end if
```

می توان توسط دکمه های رادیویی(Option) جهت جستجو را خود تعیین کنیم. پس دو دکمه رادیویی به فرم اضافه کرده و دستور را بازنویسی می کنیم

```
Dim bk as Variant
Bk = adodc1.RecordSet.BookMark → ذخیره محل فعلی
If Option1.Value then
        Adodc1.Recordset.Find Text1.Text , 0 , adSearchForward →جلو
Elseif Option2.Value then
        Adodc1.Recordset.Find Text1.Text , 0 , adSearchBackward →عقب
End if
If Adodc1.Recordset.EOF OR Adodc1.Recordset.BOF Then → اگر یافت نشد
adodc1.RecordSet.BookMark = bk → بازگشت به محل قبلی
msgbox "یافت نشد!"
end if
```

آشنایی با دو کنترل دیگر

DataList: این کنترل برای نمایش همه اطلاعات ستون خاصی از جدول استفاده می شود. دو خصوصیت زیر برای اتصال باید تنظیم شود:

- Rowsource : نام کنترل ADO به آن داده می شود که متناظر با نام جدول تنظیم شده است.
- ListField : نام ستون مورد نظر

DataCombo: این کنترل برای نمایش همه اطلاعات ستون خاصی از جدول استفاده می شود. مشابه DataList است فقط از نظر ظاهری با آن فرق دارد و شبیه Combo Box فقط یک مقدار را نشان می دهد و برای نمایش بقیه مقادیر باید فلش رو به پایین آنرا کلیک کنیم.

مثال: فرض کنید نام بانک اطلاعاتی Products و نام جدول Parts باشد و بخواهیم اطلاعات نام قطعه و قیمت را به کمک این دو کنترل ببینیم.

۱. پروژه جدید و ذخیره در مسیر مورد نظر
۲. طراحی بانک اطلاعاتی و ذخیره در همان مسیر پروژه
۳. افزودن کنترلهای لازم به ابزارها و سپس فرم

Components → M.S DataList Control 6.0, DataGrid , …

۴. تنظیمات زیر را برای کنترلها در Form_Load انجام دهید:

```
Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
  "Data Source=" & App.Path & "\Products.mdb;Persist Security Info=False"
Adodc1.CommandType = adCmdTable
Adodc1.RecordSource = "Parts"
Adodc1.Refresh
```

۵. تنظیمات اتصال کنترلها به Ado را انجام دهید.

DataGrid→ Data source: Adodc1

DataList→ Row source: Adodc1, ListField: نام ستون

DataCombo→ Row source: Adodc1, ListField: نام ستون

نکته: گاهی با اجرای برنامه کنترل Grid خطایی را اعلام می کند. با تنظیم خصوصیت TabStop با مقدار False حل می شود.

فصل پنجم- نرم افزار Crystal report

نرم افزار تهیه گزارش از اطلاعات ذخیره شده در بانک اطلاعاتی Access, SQL Server, Oracle

مراحل اتصال به بانک اطلاعاتی

۱. در اولین پنجره گزینه اول Using the report Expert
۲. در لیست بعدی گزینه اول Standard
۳. انتخاب دکمه Database
۴. در بخش Data Explorer به یکی از روشها مانند ODBC بانک اطلاعاتی خود را مشخص می کنیم
۵. بعد از انتخاب جدول مورد نظر با کلیک(Add) دکمه Close
۶. در مرحله بعدی(Next) انتخاب فیلدهای مورد نظر(Add All همه)
۷. زدن دکمه Finish
۸. با انتخاب زبانه Design در سمت چپ، می توان فونت، اندازه، محل، .... را تغییر داد. دقت کنید سطر اول عناوین ستونها و سطر دوم مقادیر آنهاست.

اجزای اصلی یک گزارش

1- Report Header: عنوان اصلی گزارش در اولین صفحه
2- Page Header: عنوان هر صفحه که در بالای آن نمایش داده می شود. مثل: نام ستونها، شماره صفحه، تاریخ
3- Detail: اصل گزارش و مقادیر ستونها
4- Report Footer: عنوان آخر گزارش در آخرین صفحه
5- Page Footer: در پایین هر صفحه مثل: شماره صفحه، تاریخ، ....

مراحل کار با C.R در محیط برنامه نویسی VB

۱. در منوی Project گزینه Add Crystal Report با نصب برنامه فعال می شود
۲. با انتخاب آن، پنجره C.R Gallery باز می شود که در آن حالت Standard و Expert را انتخاب نمایید.
۳. در پنجره بعد، دکمه Project را بزنید که در پنجره جدید 4 روش اتصال به بانک اطلاعاتی را می بینید مثلاً گزینه ADO اتصال بصورت Connection String را فراهم می کند که با زدن دکمه Build و تنظیم و انتخاب بانک اطلاعاتی مورد نظر و زدن OK
۴. در پنجره ظاهر شده دو روش برای تعیین جدول وجود دارد:
   - انتخاب جدول(Table) در قسمت Object Type و انتخاب نام جدول مورد نظر در قسمت Objects
   - انتخاب گزینه SQL و نوشتن دستور مربوطه (Select) در کادر داده شده
۵. زدن دکمه Next و انتخاب فیلدها و سپس Finish
۶. در پنجره بعدی، دو انتخاب با حالت Yes/No داریم
   - آیا فرم جدید حاوی کنترل گزارش گیری CR اضافه شود؟
   - آیا فرم جدید، به عنوان فرم اصلی برنامه باشد؟

مثال:

فرض کنید بانکی به نام Productsداریم که جدولی به نام Parts برای ذخیره مشخصات قطعات کامپیوتر با مشخصات زیر در آن تعریف شده باشد.

نام فیلد به انگلیسی	نام فیلد به فارسی
ID	شماره
Pname	نام
Model	مدل
Price	قیمت
Number	تعداد

۱ - بانک اطلاعاتی خود را در مسیر ذخیره پروژه قرار دهید

۲ - در محیط VB از مسیر زیر برنامه Crystal را فعال کنید

Project→Add Crystal Report 8.5

۳ - مراحل تنظیمات گزارش را انجام دهید تا اطلاعات جدول Parts نمایش یابد.

۴ - روی فیلد Price در قسمت Design گزارش کلیک راست کنید و گزینه های زیر را دنبال کنید

Format → Font→ Color → X*2 دکمه

۵ - در پنجره مربوطه دستور زیر را بنویسید تا کالاهای با قیمت کمتر از 20000 قرمز شوند

If {ado.Price} < 20000 then crRed

۶ - در قسمت سمت چپ از پنجره گزارش روی Formula Field کلیک راست کرده و سپس گزینه New را می زنیم . در این بخش می خواهیم حاصلجمع کل قیمت هر کالا را با ضرب تعداد در قیمت نمایش دهیم. پس می نویسیم:

{ado.Price}*{ado.Number}

این فیلد جدید طبق فرمول داده شده بدست می آید و در گزارش نمایش داده می شود . ولی در بانک اطلاعاتی تغییری ایجاد نمی کند.

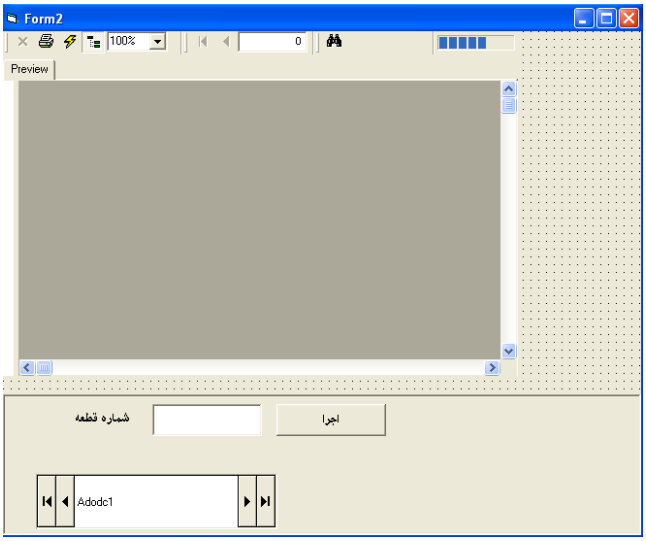
۷ - فیلد جدید را با نام Total ذخیره کرده و بعد از تعریف آن به کمک ماوس به قسمت Detail گزارش بعد از آخرین ستون می کشیم

مثال 2:نمایش کالای خاص که توسط کادر متن تعیین شود

۱ -پروژه جدیدی باز کرده و در مسیر ذخیره بانک Products ذخیره کنید.

۲ - در محیط VB از مسیر زیر برنامه Crystal را فعال کنید

Project→Add Crystal Report 8.5



۳ -مراحل تنظیمات گزارش را انجام دهید تا اطلاعات جدول Parts نمایش یابد.

۴ - روی فرم دوم که حاوی کنترل CRViewer است، کنترلهای زیر را قرار دهید

- PictureBox با تنظیم خصوصیت Align به Align Bottom
- کنترلهای Textbox, Label, Command, Ado درداخل PictureBox

در رویداد دکمه دستورات زیر را بنویسید:

```
Me.Adodc1.CommandType =adCmdText
Me.Adodc1.ConnectionString = "Provider=Microsoft.Jet.OLEDB.4.0;" & _
  "Data Source=" & App.Path & "\Products.mdb;Persist Security Info=False"
Me.Adodc1.RecordSource = “Select * from Parts where ID = ” & text1.text
Me.Adodc1.Refresh
Report.Database.SetDataSource  adodc1.Recordset
CRViewer1.ReportSource = Report
CRViewer1.ViewReport
```

برنامه را اجرا کنید و پس از وارد کردن کد محصول، دکمه را بزنید. برای نمایش نتیجه دکمه refresh از برنامه گزارش را بزنید. این دکمه با تنظیم زیر فعال میگردد:

EnableRefreshButton=true

فصل هفتم API

API (Application Programming Interface) به معنای رابط برنامه نویسی کاربردی است و به مجموعه ای از روالهای داخلی ویندوز گفته می شود که می توان آنها را در هر محیط برنامه نویسی استفاده کرد

- در فایلهایی به نام DLL ذخیره می شوند
- فایلهای DLL در پوشه Windows, System ذخیره می گردند
- هنگام نصب ویندوز، نصب میشوند
- DLL مخفف Dynamic Link Library به معنای کتابخانه پیوند پویاست.
- پسوند فایلهای DLL بصورت dll. یا exe. هستند.
- سه فایل مهم DLL عبارتند از:
  - User32.dll: کنترل رابط کاربر، ماوس، منوهاو پنجره ها
  - Gdi32.dll: کنترل خروجی به صفحه نمایش و ...
  - Kernel32.dll: کنترل سخت افزار و نرم افزار داخلی ویندوز
- توابع Dll ها در زمان اجرا در دسترس هستند و در زمان ترجمه با برنامه ترکیب نمی شوند

نحوه استفاده از توابع API

۱. به کمک ابزار API Text Viewer در مسیر زیر آنها را پیدا میکنیم
Program files → Microsoft Visual studio → … Tools
۲. در محیط برنامه API Viewer گزینه File → Load Text File را بزنید و فایل Win32API.txt را انتخاب نمایید.
۳. در قسمت API Type سه گزینه وجود دارد:
   - Constants: همه ثابتهای نام گذاری شده مورد استفاده توابع API
   - Declares: فهرست اعلانهای لازم برای توابع API
   - Types: همه انواع داده های مورد استفاده توابع API
۴. کلیه روالها و موارد مرتبط در قسمت Available Items ظاهر می شوند. با تایپ چند حرف اول نام تابع مورد نظر ، آنرا در لیست پیدا میکنید.
۵. بعد روی نام تابع مورد نظر کلیک کرده(یا دکمه Add) تا اعلان آن در قسمت Selected Items قرار گیرد
۶. حالت Public یا private (استفاده در Module) و private (برای استفاده در فرم) را انتخاب نموده و سپس دکمه Copy را بزنید.
۷. به محیط برنامه نویسی برگشته و در قسمت General آنرا Paste کنید.
۸. در محیط برنامه نام تابع را فراخوانی کنید تا کار مورد نظر شما انجام شود.
۹. در اعلان یک تابع API چند قسمت وجود دارد:
   - نام تابع
   - نام فایل Dll
   - نام مستعار تابع
   - لیست پارامترهای تابع با یکی از دو کلمه رزرو شده زیر:
     - ByVal: تابع نمی تواند مقدار متغیر ارسال شده به تابع را تغییر دهد
     - ByRef: تابع می تواند مقدار متغیر ارسال شده به تابع(پارامتر) را تغییر دهد.

مثال: پخش صدای بوق و باز و بسته شدن درب CD_Rom در برنامه
۱ - پروژه جدید به همراه سه دکمه و ذخیره
۲ - برنامه API Viewer را باز کرده و در قسمت Declare تابع Messagebeep و mciSendString را یافته و Copy کنید و در محیط برنامه Paste کنید.
۳ - در دکمه ها دستورات زیر را بنویسید:
در دکمه اول: عدد مثبت یعنی پخش از کارت صوتی → Messagebeep 100
دکمه دوم → mciSendString "Set CDAudio Door Open Wait", 0&,0&,0&
دکمه سوم → mciSendString "Set CDAudio Door Closed Wait", 0&,0&,0&

مثال 2: نمایش نام کاربری خود

۱ - یک دکمه و Text در محیط برنامه
۲ - در برنامه API Viewer ، GetUserName را کپی می کنیم
۳ - آنرا در محیط برنامه Paste کرده
۴ - در روال دکمه دستورات زیر را بنویسید:

```
Dim buffer As String → محل ذخیره نام کاربری
buffer = String(255, 0) →255 رشته خالی به طول
GetUserName buffer, 255 →API فراخوانی
buffer = Left(buffer, InStr(buffer, Chr(0)) - 1) → 0 کپی کارکترهای قبل از کد کاراکتر
Text1.Text = buffer → نمایش نام کاربر
```

مثال 3: تشخیص کارکتر ورودی(بزرگ یا کوچک) یا عددی
از توابع زیر استفاده می شود:

- IsCharAlphaNumeric: تشخیص حرفی یا عددی
- IsCharAlpha: تشخیص حرفی
- IsCharLower: آیا حرف وارد شده کوچک است
- IsCharUpper: آیا حرف وارد شده بزرگ است

این توابع API را به برنامه اضافه نمایید و در رویداد KeyPress از فرم دستورات زیر را بنویسید:

```
Dim s as string
Cls
If IsCharAlphaNumeric(KeyAscii) then s = "عددی یا حرفی"
If IsCharAlpha (KeyAscii) then s = "حرفی"
If IsCharLower(KeyAscii) then s = s + "کوچک"
If IsCharUpper(KeyAscii) then s = s + " بزرگ"
Print s
```

مثال 4: مخفی کردن Task Bar

۱ - از قسمت Declare از API Viewer دو تابع FindWindow, SetWindowPos را اضافه نمایید
۲ - از قسمت Constant از API Viewer دو ثابت زیر را نیز اضافه نمایید.

```
SWP_HIDEWINDOW = &H80
SWP_SHOWWINDOW = &H40
```

۳ - دو دکمه برای ظاهر کردن و مخفی کردن به فرم اضافه کنید
۴ - در قسمت General :

```
Private Hwndl As Long
```

۵ - در دکمه «ظاهر کردن» دستور زیر:

```
Call setWindowPos(Hwndl,0,0,0,0,0,SWP_SHOWWINDOW)
```

۶ - در دکمه «مخفی کردن» دستورات زیر:

```
Hwndl=FindWindow("Shell_Traywnd","")
Call setWindowPos(Hwndl,0,0,0,0,0,SWP_HIDEWINDOW)
```

مثال 5: نمایش مدت زمان روشن بودن کامپیوتر

۱ - روی فرم کنترل Timer و دو برچسب قرار دهید(نام یکی را به LblTime تغییر دهید.)
۲ - تابع API را به برنامه اضافه کنید: GetTickCount
۳ - در رویداد Timer دستورات زیر:

```
Private Sub Timer1_Timer()
  Dim t As Long, h As Integer, m As Integer, s As Integer
  t = GetTickCount → مدت زمان روشن بودن کامپیوتر برحسب میلی ثانیه
  t = t \ 1000 → برحسب ثانیه
  h = t \ 3600 → ساعت
  t = t Mod 3600 → باقی مانده از ساعت
  m = t \ 60 → دقیقه
  s = t Mod 60 → ثانیه
  lblTime.Caption = h & " : " & m & " : " & s
End SubVB
```