ماکرو نویسی در اکسل VBA

برای انجام عملیات تکراری و جستجوی داده های مورد نظر می توان از ابزار قدرتمند ماکروها در اکسل استفاده کرد. ورود به ساختار برنامه نویسی ویژوال بیسیک در اکسل که با عنوان VBA یا VISUAL BASIC FOR APPLICATION در اکسل و یا آفیس از آن یاد می گردد ، از مسیر زیر انجام می گیرد :

TOOLS | MACRO | VISAUL BASIC EDITOR

در حقیقت ماکرو ها، همان کدهایی هستد که توسط کاربر یا ماشین تولید می شوند. اگر شما یک ماکرو را با استفاده از ابزار ماکرو سازی طراحی و اجرا کنید آنرا توسط ماشین ساخته اید ولی اگر بخواهید انعطاف بیشتری به آن بدهید باید آنرا بنویسید(کد نویسی). زبان وی بی ای در حقیقت معماری درونی اکسل و ابزاری قدرتمند برای نوشتن برنامه های پیشرفته و حلقوی است . این زبان که یک زبان شئ گراست ، اجزای درونی خود را به شکل اشیائی در نظر می گیرد که نرم افزار اکسل را تشکیل می دهند، مثل فایل (کارپوشه WORK BOOK) ، کاربرگ یا WORK SHEET و یا سلول RANGE .در حقیقت کارپوشه ها از اشیاء کاربرگ تشکیل شده اند و کاربرگ ها با اشیاء سلول کامل می شوند . هر شیء دارای یک سری خاصیت است . مثلا شما یک سنگ را در نظر بگیرید ، رنگ ، وزن ، شکل و... خواص سنگ محسوب می شوند .در وی بی ای VBA نیز همینطور است ، مثلا یک سلول دارای خواصی مثل محتویات ، اندازه فونت ، نام فونت ، رنگ ، فرمول ، کادر و .... می باشد. کاربران در برنامه نویسی در حقیقت این خواص را تغییر می دهند .

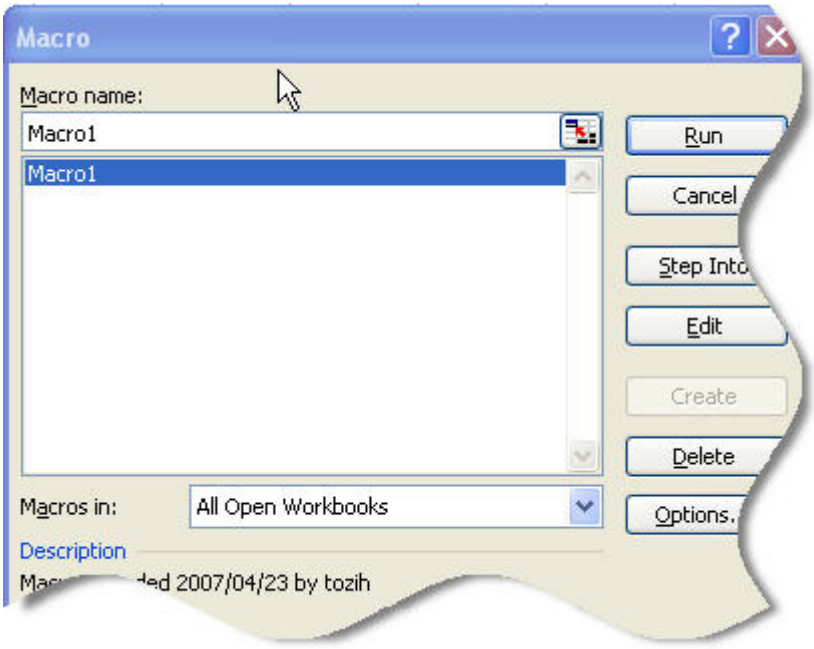
بیایید یک تمرین را باهم انجام دهیم :

ابتدا نوار ابزار ویژال بیسیک را فعال کنید ، سپس ماکرویی را طراحی کنید که در سلول A1از کاربرگ 2 وارد شود .

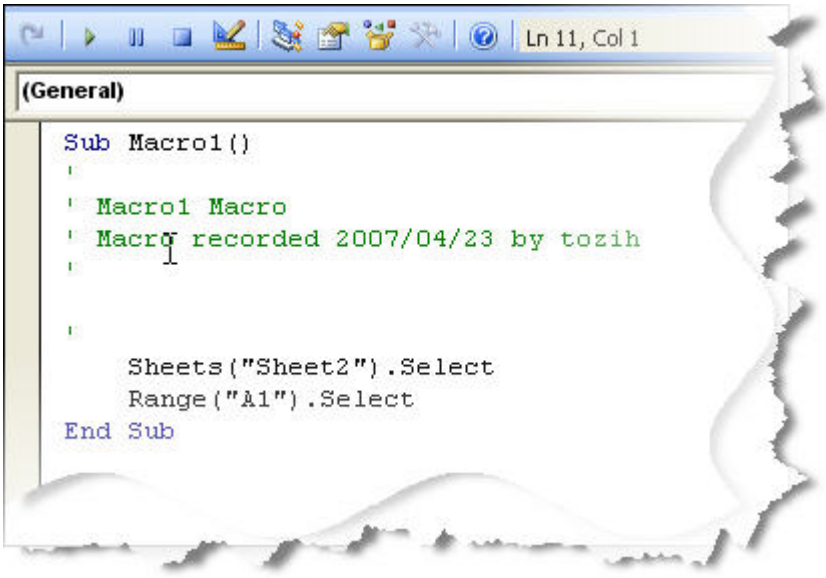
مراحل ساخت ماکرو :

ابتدا کلید RECORD MACRO را می فشاریم و سپس به کاربرگ 2 و سلول A1 وارد می شویم و در آخر کلید STOPرا می فشاریم .
حال برای دیدن کد ماکرو مسیر زیر را دنبال کنید :

کلید RUN را فشرده تا اسامی ماکرو ها ظاهر شود سپس ماکروی مورد نظر را انتخاب کرده کلید EDIT را می فشاریم تا متن ماکرو (کد) نمایش داده شود .



همانطور که مشاهده می کنید ماکرو با SUB و نام ماکرو ،شروع و با END SUB تمام می شود و در بین آنها عبارات برنامه نویسی نوشته شده است . سه رنگ در بدنه ی ماکرو بکار رفته 1- سبز: که معرف توضیحات برنامه است و هیچ تاثیری بر عملکرد ماکرو ندارد 2- آبی : کلمات کلیدی 3- سیاه : دستورات

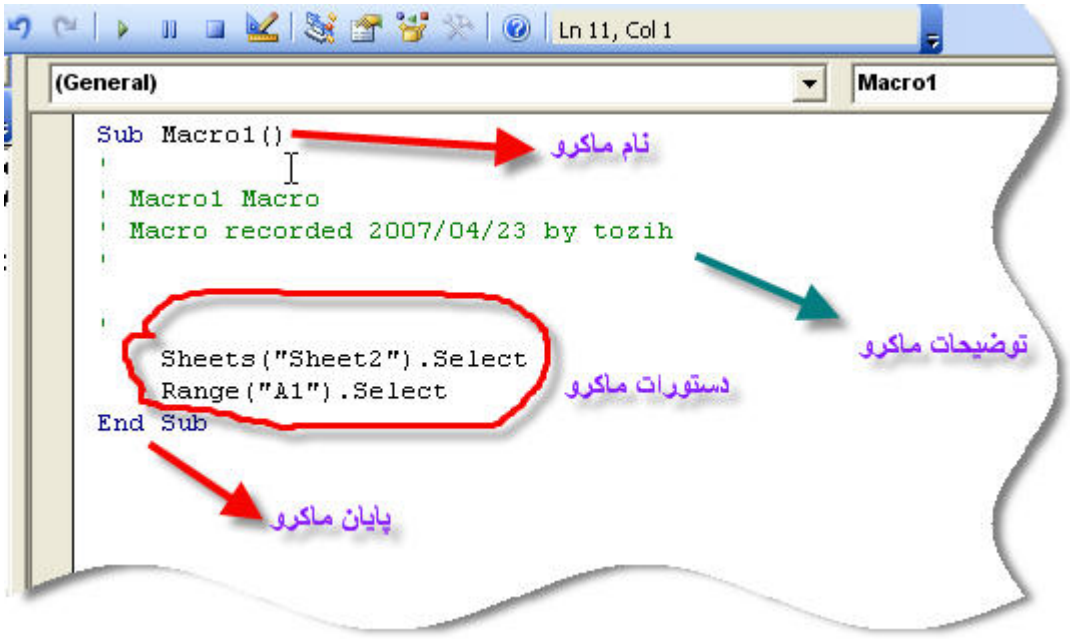


Sub Macro1()

' Macro1 Macro
' Macro recorded 2007/04/23 by tozih

Sheets("Sheet2").Select
 Range("A1").Select

End Sub



همانطور که از کد ها مشخص است ماکرویی بنام MACRO1 به کاربرگ 2 رفته و سلول A1 را در آن انتخاب می کند . در ساختار برنامه نویسی VBA تغییر خواص به صورت توارثی انجام می پذیرد ، یعنی هر خاصیت باید به شئ مورد نظر، با یک نقطه متصل باشد .

```
Range("A1").Select
```

در حقیقت برای تغییر خاصیت اشیاء باید ابتدا نام شئ را ذکر کرده سپس نقطه را قرار داده و آنگاه خاصیت مورد نظر را ذکر کنیم و مقادیر آنرا تغییر دهیم . مثلا :

```
RANGE("A2:A10").FONT.SIZE = 16
```

اندازه فونت محتویات سلول های A1 تا A10 را به 16 تغییر می دهد .
مثال دیگر ، محتویات سلول D2 از کاربرگ 3 را عبارت ALI TOZIH قرار دهید .

```
SHEETS("SHEET3").RANGE("D5").VALUE = "ALI TOZIH"
```

در ضمن خاصیت VALUE پیش فرض است و می توان آنرا حذف کرد :

```
SHEETS("SHEET3").RANGE("D5") = "ALI TOZIH"
```

چند قاعده

این قواعد را هنگام ساختن ماکروها به یادداشته باشید:
نامگذاری ماکـــــــــــرو:در نام ماکرو نباید از فضای خالی(Space) استفاده کنید. بهتر است برای آن نامی منطقی انتخاب کنید که گویای عملکرد آن باشد.

فعال کردن ماکرو: زمانی که ترکیبی از کلیدها را در نظر گرفتید تا بعدا آنها را برای اجرای ماکرو به کار ببرید، دقت کنید که این ترکیب قبلا به مورد دیگری اختصاص داده نشده باشد، مانند ترکیب کلیدهای Ctrl+C که بـــرای فرمان Copyدر نظر گرفتـــــه شده است .در چنین شرایطی، کادر محاورهای Record Macro به شما هشدار میدهد که ترکیب کلیدی مورد نظر را انتخاب نکنید چون برای فرمان دیگری در نظر گرفته شده است.
انتخاب یک لایه امنیتی :از آنجا که اکثر افراد، ویروسها را از طریق ماکـــتروها انتشار میدهند، به همین دلیل XP یک ویژگی امنیتی با سه سطح بـــالا، متوسط و پایین در اختیار کاربـــــــران میگذارد .برای تنظیم سطح امنیتی، یک بـــرنامه کاربردی Office را باز کرده و سپس به ترتیب گزینه هایTools، MacroوSecurityرا انتخاب کنید تا کادر محاورهای Security ظاهر شود .اگر گزینه High را انتخاب کنید، دیگر نمیتوانید هیچ ماکرویی را اجرا کنید .اگر مایلید ماکروها را اجرا کنید، لازم است سطوح امنیتی Medium و یا Low را در نظر بگیرید.
اگـــــر هنگام اجرای ماکرو نمی خواهید اخطارها را مشاهده کنید، سطح امنیتی Medium بهترین انتخاب است و اگر از ماکروها زیاد استفاده می کنید و نمی خواهید به هیچ وجه اخطاری دریافت کنید، گزینه Low را برگزینید.

ساده اما کامل

در پایان، بهترین کار اینست که ماکروهای خود را هر چه ساده تر ایجاد کنید، زیرا ماکروهای پیچیده، همیشه آن طور که می خواهید عمل نمی کنند. به علاوه ساده ترین ماکروها، مفیدترین هم خواهند بود.

INPUTBOXجعبه دریافت داده:

هر گاه کاربر قصد دریافت داده ها را از کاربرگ نداشته باشد و بخواهد آنها را مستقیما دریافت کند از این ابزار استفاده می کند :

```
Sub ALI()
  I = InputBox("عدد مورد نظر را وارد کنید    ")
End Sub
```

MSGBOXجعبه پیام:

هر گاه کاربر قصدداشته باشد پیامی را بر صفحه نمایش دهد از این دستور استفاده می کند.

```
Sub ALI()
  MsgBox"عدد وارد شده"
End Sub
```

انواع متغیر ها و دامنه کاربردآنها:

Data type	Storage size	Range
Byte	1 byte	0 to 255
Boolean	2 bytes	**True** or **False**
Integer	2 bytes	-32,768 to 32,767
Long (long integer)	4 bytes	-2,147,483,648 to 2,147,483,647
Single (single-precision floating-point)	4 bytes	-3.402823E38 to -1.401298E-45 for negative values; 1.401298E-45 to 3.402823E38 for positive values
Double (double-precision	8 bytes	-1.79769313486231E308 to -4.94065645841247E-324 for negative

floating-point)		values; 4.94065645841247E-324 to 1.79769313486232E308 for positive values
Currency (scaled integer)	8 bytes	-922,337,203,685,477.5808 to 922,337,203,685,477.5807
Decimal	14 bytes	+/- 79,228,162,514,264,337,593,543,950,335 with no decimal point; +/-7.9228162514264337593543950335 with 28 places to the right of the decimal; smallest non-zero number is +/-0.0000000000000000000000000001
Date	8 bytes	January 1, 100 to December 31, 9999
Object	4 bytes	Any **Object** reference
String (variable-length)	10 bytes + string length	0 to approximately 2 billion
String (fixed-length)	Length of string	1 to approximately 65,400
Variant (with numbers)	16 bytes	Any numeric value up to the range of a **Double**
Variant (with characters)	22 bytes + string length	Same range as for variable-length **String**

برای استفاده ا ز متغیر ها در VBA بهتر است که آنها را تعریف کنیم ، دو نوع متغیر عددی و غیر عددی وجود دارد که عددی ها عبارتند از :

1. BYTE
2. INTEGER
3. LONG
4. SINGLE
5. DOUBLE
6. CURRENCY
7. DECIMAL

و غیر عددی عا عبارتند از:

1. BOOLEAN

2. OBJECT
3. STRING
4. VARIANT
5. DATE

روش تعریف متغیر:

روش معرفی متغیر ها استفاده از کلمه ی کلیدی DIM بعد از نام ماکروست . به مثال زیر توجه کنید:

I به عنوان یک عدد صحیح (متغیر) معرفی شده است.

```
SUB ALI()
 DIM I AS INTEGER
 I  = INPUTBOX (" ENTER  A NUMBER INTEGER:")
END SUB
```

شرط ها و ساختار های تصمیم گیری درVBA:

یکی از مهمترین بخش های هـــر زبان بـــــرنامه نویسی ساختار های تصمیم گیری در آنهاست .

IF

IFیکی از پرکاربرد ترین دستورات VBA می باشد ، ساختار آن عبارت است از:

```
IF   [condition]  THEN
[statement]
END IF
```

مثال : می خواهیم ساختار شرطی ایجاد کنیم که اگر محتویات سلول A1 از عدد 10 کوچکتر بود یک پیام مردودی نمایش دهد . (فقط ساختار IF را نمایش می دهیم)

```
IF  RANGE("A1")<10 THEN
 MSGBOX" مردود"
```

END IF

اگر قرار باشد دردستورشرطی ،دوشرط را کنترل کنیم می توانیم دستور را به صورت زیر بنویسیم:

IF [condition1] and [condition2] THEN

[statement]

END IF

ماکرویی طراحی کنید که نمره ی دانش آموزی را از کاربر گرفته اگر

نمره کمتر یا مساوی 7 باشد ،ضعِف

نمره بیشتر از 7وکمتریا مساوی 14 ،متوسط

نمره بیشتر از 14 ،خوب

پیام های مقابل آنها را چاپ نماید.

```
Sub NOMREH()

Dim NOM As Single

NOM = InputBox("نمره را وارد کنید")

If NOM >= 0 And NOM <= 7 Then MsgBox "BAD"

If NOM > 7 And NOM <= 14 Then MsgBox " NOT BAD"

If NOM > 14 And NOM <= 20 Then MsgBox "GOOD"

End Sub
```

به این نکته توجه داشته باشید که اگر فرامین IF فقط یک دستور باشند دیگر نیازی به END IF نیست البته دستور مورد نظر را باید در پشت سر THEN نوشت .

If...Then...Else Statement

If *condition* **Then** [*statements*] [**Else** *elsestatements*]

Or, you can use the block form syntax:

If *condition* **Then**
[*statements*]

[**ElseIf** *condition-n* **Then**
[*elseifstatements*] . . .

[**Else**
[*elsestatements*]]

End If

حلقه سازی :

بدون حلقه سازی ، برنامه ها بسیار خسته کننده ودردسرساز خواهند بود.حلقه سازی اجازه میدهد تا درزمان برقرار بودن یک شرط یا رسیدن به یک مقدار مشخص ،یک بلوک کد اجرا شود.برای مثال فرض کنید که می خواهید اعداد 1 تا 6 را نمایش دهید ،می توانید برنامه را بصورت زیر بنویسید:

```
MsgBox”1”
MsgBox”2”
MsgBox”3”
MsgBox”4”
MsgBox”5”
MsgBox”6”
```

این برنامه جواب می دهد ولی کارآمد نیست اگر بخواهیم اعداد بیشتری را چاپ کنیم می بایست تعداد زیادی خط دیگر را به این کد اضافه کرد.

FOR…Next

با این روش می توان جهت انجام فعالیتهای تکراری حلقه تعرِف کرد.شکل دستور به شرح ذیل می باشد:

```
For n=a to b

[statements]

Next n
```

با استفاده از دستور حلقه سازی می توانیم کد فوق را ساده تر بنویسیم:

```
For n=1 to 6
   MsgBox n
```

```
Next n
```

دستورات بکار گرفته شده در بین FOR ,NEXTمی توانند به هر تعدادی باشندو حتی می توانیم در بین آنهاتوابع را نیز فرا خوانیم.مقادیر شروع و خاتمه در حلقه FOR…NEXT هم می توانند متفاوت باشند واجباری نیست که با 1 شروع وبا n خاتمه یابند.

گزینه step ،کـــــــارایی بیشتری را برای حلقه FOR…NEXT ایجاد میکند و این امکان را به شما می دهد که شمارنده حلقه بتواند افزایشی به غیر از یک واحد داشته باشد.به مثال زیر دقت کنید:

```
For n=3 to 12 step 3
  MsgBox n

Next n
```

خروجی کد زیر 3,6,9,12 خواهد بودچون گام حلقه 3 است.

برای این که ببینید step چطور به صورت معکوس عمل می کند به مثال زیر دقت کنید.

```
For n=10 to 1 step -1
  MsgBox n

Next n
```

Do Until

حلقه Do Until تا زمان برقرار بودن یک شرط خاص ، تکرار را ادامه می دهد.اکثر مواقع این شرط ،رسیدن یک متغیر به مقداری خاص است.وقتی شرط برقرار شود حلقه متوقف می شود و بـــــرنامه به اجرای دستورالعمل بعد از حلقه می پردازد.به مثال زیر توجه کنید:

```
Sub test_do ( )
x=0
Do Until x=100
   x=x+1
Loop
MsgBox x

End Sub
```

While… wend

در آخر ، حلقه While…Wend را مطرح می کنیم .اجرای این حلقه تا زمانی که یک شرط خاص برقرار باشد ادامه می یابدو به محض آن که شرط برقرار نباشد اجرای حلقه متوقف می شود.

```
Sub test_do ( )
x=0
While x<50
   x=x+1
Wend
MsgBox x

End Sub
```

زمانی که x دیگر کمتر از 50 نباشد متوقف می شود.

*** نکته: برای خروج زود هنگام از حلقه می توان از دستور Exit For استفاده کرد.**

یک مثال ساده:

```
Sub test_exit ( )

For x=1 to 100
        If x=50 then
                Exit for
        End if
Next x
  MsgBox x
End sub.
```

تابع چیست ؟

تابع یک عملگر است که ورودی را می‌گیرد و روی آن پردازشی را انجام می‌دهد و سپس نتیجه پردازش را تحت عنوان خروجی به ما می‌دهد.
مثلا همین تابع SUM را در نظر بگیرید ، چند تا عدد می‌گیرد و جمع آنها را به ما می‌دهد.

توابع در اکسل (1)
شکل کلی یک تابع در اکسل به صورت :
=نام تابع(ورودی ۱ ,ورودی ۲ ,...)
=sum(... , ..., ...)
=pmt(..., ... , ...)
ورودی ها با علامت , یا ; از همدیگر جدا می شوند.
24

اکسل بیش از 300 تابع دارد که اکثر کارهایی که ممکن است بخواهیم انجام دهیم با این توابع قابل انجام است ، در ضمن ترکیب این توابع نیز برای ما امکانات فراوانی را بهمراه دارد و این را هم مد نظر داشته باشیم که شرکتی به عظمت ماکروسافت و تجربه چندین ساله‌اش مطمئنا نیازهای تمامی کاربران در سطح دنیا را در نظر داشته و تا آنجایی که امکان داشته توابع مختلف را پیش بینی کرده است .

توابع جدید به چه کاری می‌آیند

البته این سوال ممکن است به ذهن شما متبادر شود که چرا باید تابع جدیدی اضافه کرد.
شاید دلایل زیر بتواند گوشه‌ای از ارزش تابع را برای ما بیان کند:

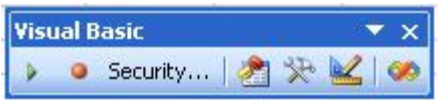
- جلوگیری از کارهای تکراری در اکسل
- انجام محاسبات پیچیده
- دسترسی به کلیه امکانات یک زبان برنامه نویسی مانند ویژوال بیسیک
- به اشتراک گذاشتن توابع با سایر کاربران
- استفاده سریعتر از نرم افزار
- جلوگیری از اشتباهات کاربران

آشنایی با ویژوال بیسیک

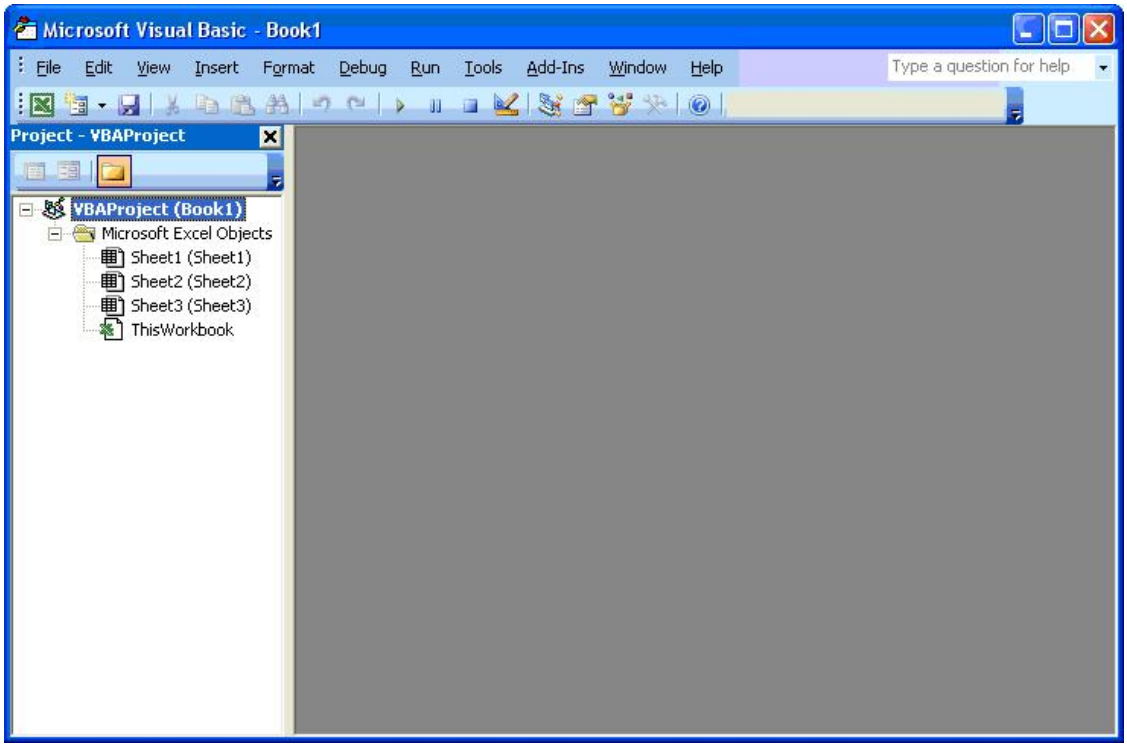
شما برای استفاده از VB در اکسل نیاز ندارید که نرم افزار VISUAL BASIC را نصب کنید ، همراه با نصب آفیس خود این نرم افزار نیز نصب می‌شود.

گام اول ورود به محیط ویژوال بیسیک

- ابتدا بایستی وارد محیط VB شویم. برای اینکار چندین راه وجود دارد که عبارتند از:
- زدن کلید ALT+F11
- از منوها : Tools → Macro → Visual Basic Editor
- از Toolbar :

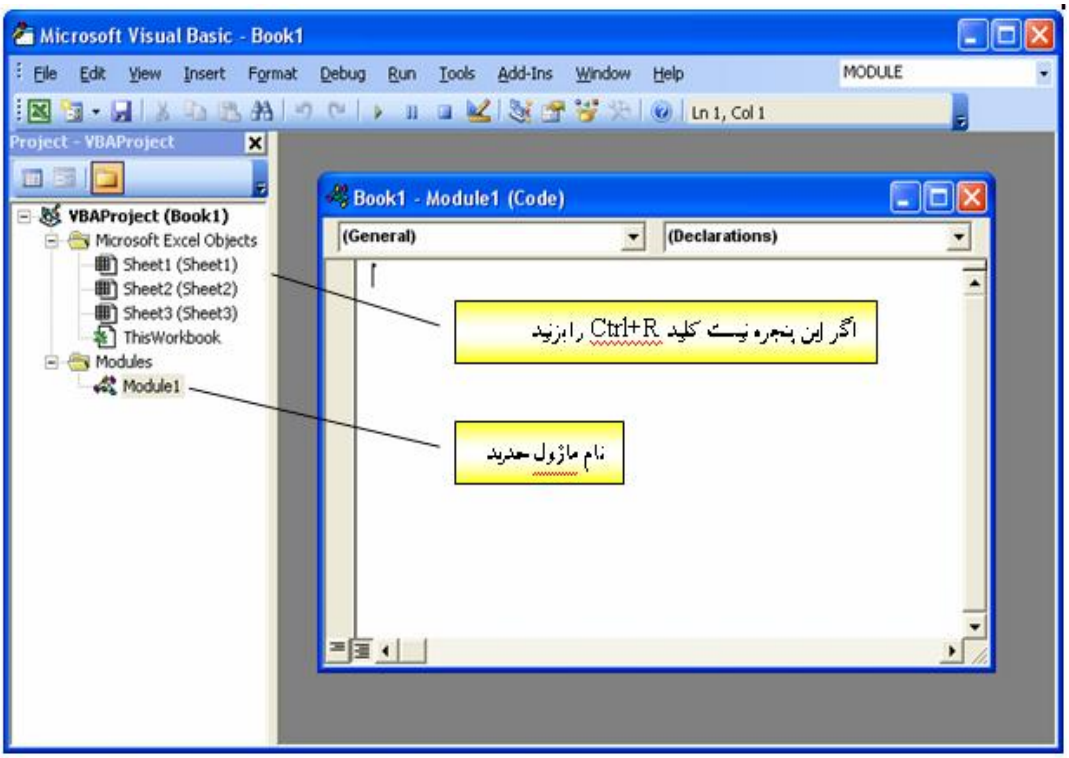


محیط ویژوال بیسک



گام دوم ایجاد یک ماژول

- شما باید دستورات تابع خود را در یک Module (ماژول) بنویسید ، از منوی Insert گزینه Module را بزنید . و اگر به project explorer نگاه کنید متوجه خواهید شد که یک ماژول جدید ایجاد شده است.



گام سوم ایجاد یک تابع در ماژول

- یک تابع در ویژوال بیسیک قواعد استانداردی دارد که شما باید از این قواعد اطاعت کنید .
- اولین قانون آن این است که یک تابع با دستورات استانداردی شروع و به پایان می‌رسد.
- قانون دوم این است که هر تابع یک نوع دارد و ورودیهای یک تابع در داخل پرانتز مشخص می‌شوند.
- قانون سوم ، نوع داده ورودیها (و خود تابع) باید مشخص شود.
- این دستورات عبارتند از :

Private Function Test(Num As Integer) as Double

End Function

- نام تابع ما test است و عبارت داخل پرانتز می‌گوید که این تابع یک ورودی دارد که نام آن ورودی Num است و integer بیانگر آن است که این ورودی عددی صحیح است . (32,768- تا 32,767)
- خروجی تابع از نوع double است و البته گذاشتن آن در همه موارد الزامی نیست ، گرچه بهتر است که مشخص شود. (برای اطلاع بیشتر به کتابهای برنامه نویسی مراجعه کنید.)
- عبارت Private Function نشانگر شروع تابع و End Function برای پایان تابع است.

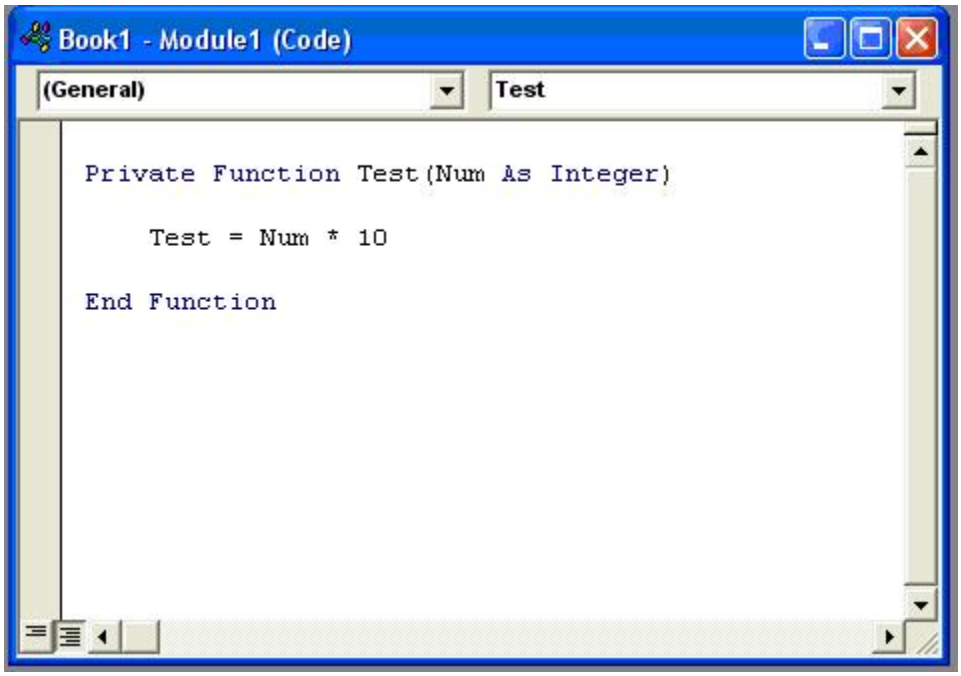
گام چهارم – نوشتن تابع

فرض کنید می‌خواهیم تابعی بنویسیم که یک عدد را بگیرد و آنرا در 10 ضرب کند!
اول باید تصمیم بگیریم که اسم این تابع را چه بگذاریم ، در حقیقت این اسم همان کلمه‌ای است که در اکسل برای استفاده از این تابع استفاده خواهیم کرد.
خوب اسم آنـــــرا Test می‌گذاریم و می‌دانیم که ایـــن تابع باید یک ورودی داشته باشد و نـــــوع این ورودی را هم Integer می‌گذاریم.
باید نامی برای این ورودی در نظر بگیریم ، این نام نباید یک نام آشنا ! برای VB باشد و بهتر است نامی با مسما در نظر بگیریم ، اینجا اسم این ورودی را Num می‌گذاریم.
پس در ماژول خود خواهیم نوشت :

```
Private Function Test(Num As Integer)

  Test = Num * 10

End Function
```



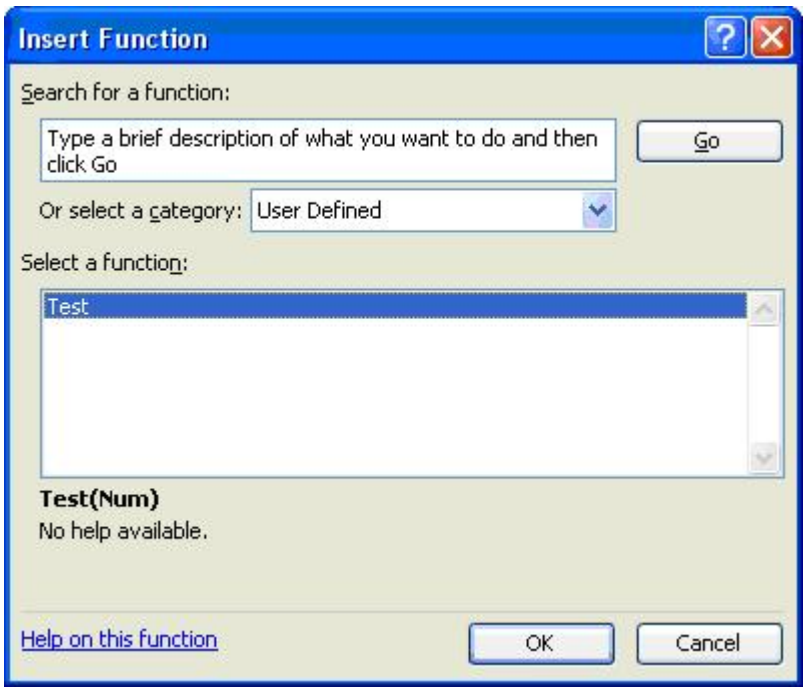
حال از ویژوال بیسیک خارج می‌شویم (Alt + Q) و به اکسل بر می‌گردیم .

گام پنجم – استفاده از تابع

مثل توابع استاندارد اکسل می‌توان از این تابع هم استفاده کرد مثلا بنویسید :

= test(8)
= test(A1)

اگر به جای کلمه Private ، Public بنویسیم، می‌توانیم نام تابع جدیدمان را در UserFunction ببینیم.



مثال:

تابعی بنویسید که فاکتوریل عدد را محاسبه کند.

```
Function FACT1(N)
S = 1
For I = 2 To N
S = S * I
Next I
FACT1 = S
End Function
```