***Excel Assignment – 16***

1. What is a Macro? How is it useful in excel or in your daily work?

Answer - **Excel Macro** is a record and playback tool that simply records our Excel steps and the macro will play it back as many times as we want. VBA Macros save time as they automate repetitive tasks. It is a piece of programming code that runs in an Excel environment but we don’t need to be a coder to program macros.

Macro in Excel helps we to achieve repeated task easily. In a layman’s language, a macro is defined as a recording of our routine steps in Excel that we can replay using a single button.

For example, we are working as a cashier for a water utility company. Some of the customers pay through the bank and at the end of the day, we are required to download the data from the bank and format it in a manner that meets our business requirements.

We can import the data into Excel and format. The following day you will be required to perform the same ritual. It will soon become boring and tedious. **Macros solve such problems by automating such routine tasks**. We can use a macro to record the steps of

* Importing the data
* Formatting it to meet our business reporting requirements.

1. What is VBA? Write its full form and briefly explain why VBA is used in excel?

Answer - VBA is an event-driven tool, which means that we can use it to tell the computer to initiate an action or string of actions. To do this, we build custom macros—short for macroinstructions—by typing commands into an editing module. A macro is essentially a sequence of characters whose input results in another sequence of characters (its output) that accomplishes specific computing tasks. We do not need to purchase the VBA software because VBA is the version of Visual Basic that ships with Microsoft Office.

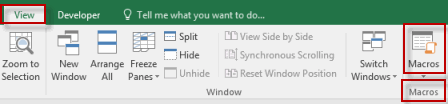
* **Visual Basic for Applications** is a computer programming language developed and owned by Microsoft.
* With VBA we can create macros to automate repetitive word- and data-processing functions, and generate custom forms, graphs, and reports.
* VBA functions within MS Office applications; it is not a stand-alone product.
* VBA is accessed in Excel by hitting **Alt + F11** while having an Excel workbook present.
* VBA leverages objects, variables, properties, projects, logical operators, and modules to make statements recognizable by debugging processes.

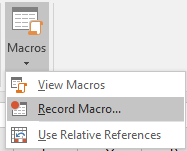
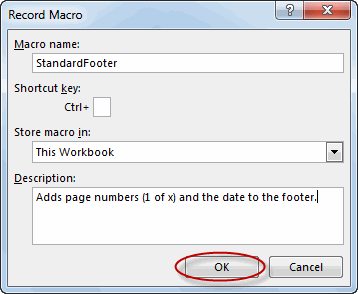
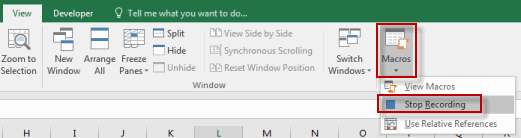
1. How do you record a macro? Write detailed steps to create a macro to automatically make the following table in bold and to create borders for it in excel.

hi 78

hello 69

ineuron 45

Answer- On the **View** tab, in the **Macros** group, click the **Macros** command.

* Click **Record Macro**.
* In the **Record Macro** dialog box:
  1. Enter a **Macro name**. Choose a name that clearly identifies the macro. You may not use spaces.
  2. Enter a **Shortcut key**. You will be able to run the macro using this shortcut key.
  3. Choose where to store the macro. You will normally accept the default (**This Workbook**).
  4. Enter a **Description**. Briefly explain what the macro does.
  5. Click **OK**.
* After clicking **OK**, every keystroke is recorded.
* To stop recording, on the **View** tab, in the **Macros** group, click the **Macros** command and then click **Stop Recording**.

To make table and bold data –

* 1. Click on developer.
  2. Click on start macro.
  3. Go into sheet, then select cell.
  4. Go into home menu ,click on all border from drop down.
  5. Click on bold option .
  6. Go on developer ribbon click on stop recording.

1. What do you mean when we say VBA Editor?

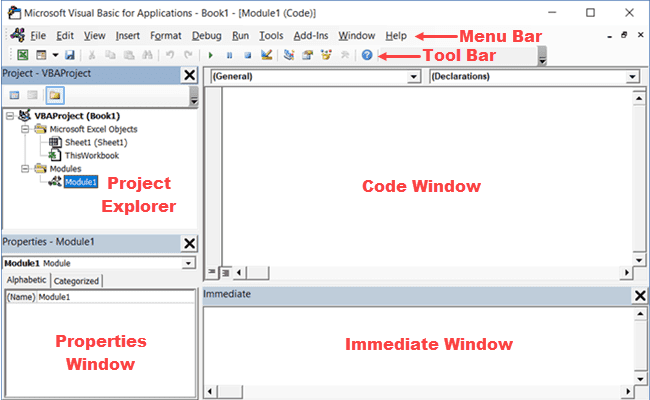
Answer - Visual Basic Editor is an application (a separate one) in which we can write and save all the VBA codes. In simple words, it’s a code editor for Excel in which we can write all the macros and store them. Even though it is a separate application (VB Editor) we can only use it with Excel .Yes, that’s right. we can’t run VBE separately; there must be an Excel workbook open for using VBE. Visual Basic Editor is the only way to write a VBA code in Excel. In fact, all Microsoft applications that host VBA use the Visual Basic Editor for script writing (writing code).

## KEY POINTS

* Visual Basic Editor is a code editor for VBA.
* It’s a separate application but we can only use it with Excel.
* We need to have the developer tab on the ribbon to access it.
* We can also use the keyboard shortcut (**Alt + F11**).
* It also stores the macros that we recode with the macro recorder.
* It has multiple tools to help we to write and manage all the codes.

5. Briefly describe the interface of a VBA editor? What is properties window? And what is watch window? How do you display these windows?

Answer - Below is an image of the different components of the VB Editor. These are then described in detail in the below sections -



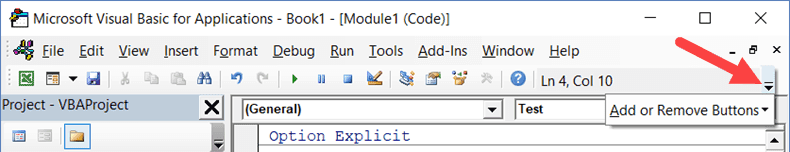
### **Menu Bar**

This is where you have all the options that you can use in the VB Editor. It is similar to the Excel ribbon where you have tabs and options with each tab. You can explore the available options by clicking on each of the menu element. You will notice that most of the options in VB Editor have keyboard shortcuts mentioned next to it. Once you get used to a few keyboard shortcuts, working with the VB Editor becomes really easy.

### **Tool Bar**

By default, there is a toolbar in the VB Editor which has some useful options that you’re likely to need most often. This is just like the Quick Access Toolbar in Excel. It gives you quick access to some of the useful options.

You can customize it a little by removing or adding options to it (by clicking on the small downward pointing arrow at the end of the toolbar).



In most cases, the default toolbar is all you need when working with the VB Editor.

You can move the toolbar above the menu bar by clicking on the three gray dots (at the beginning of the toolbar) and dragging it above the menu bar.

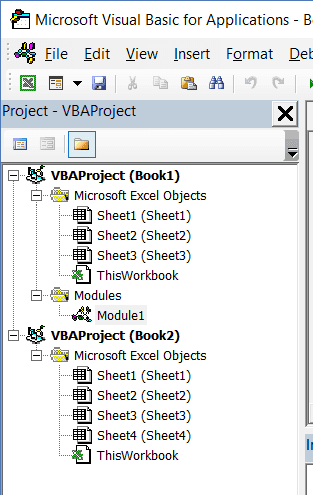
### **Project Explorer**

Project Explorer is a window on the left that shows all the objects currently open in Excel.

When you’re working with Excel, every workbook or add-in that is open is a project. And each of these projects can have a collection of objects in it.

For example, in the below image, the Project Explorer shows the two workbooks that are open (Book1 and Book2) and the objects in each workbook (worksheets, This Workbook, and Module in Book1).

There is a plus icon to the left of objects that you can use to collapse the list of objects or expand and see the complete list of objects.



The following objects can be a part of the Project Explorer:

1. All open Workbooks – within each workbook (which is also called a project), you can have the following objects:
   * Worksheet object for each worksheet in the workbook
   * This Workbook object which represents the workbook itself
   * Chart sheet  object for each chart sheet (these are not as common as worksheets)
   * Modules – This is where the code that is generated with a macro recorder goes. You can also write or copy-paste VBA code here.
2. All open Add-ins

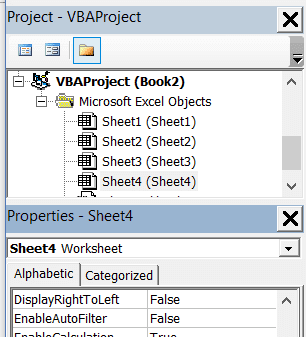
Consider the Project Explorer as a place that outlines all the objects open in Excel at the given time.

The keyboard shortcut to open the Project Explorer is **Control + R** (hold the control key and then press R). To close it, simply click the close icon at the top right of the Project Explorer window.

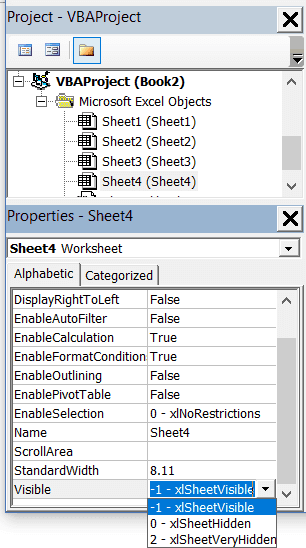
### **Properties Window**

Properties window is where you get to see the properties of the select object. If you don’t have the Properties window already, you can get it by using the keyboard shortcut F4 (or go to the View tab and click Properties window).

Properties window is a floating window which you can dock in the VB Editor. In the below example, I have docked it just below the Project Explorer.



Properties window allows us to change the properties of a selected object. For example, if I want to make a worksheet hidden (or very hidden), I can do that by changing the Visible Property of the selected worksheet object.



### **Code Window**

There is a code window for each object that is listed in the Project Explorer. You can open the code window for an object by double-clicking on it in the Project Explorer area.

Code window is where you’ll write your code or copy paste a code from somewhere else.

When you record a macro, the code for it goes into the code window of a module. Excel automatically inserts a module to place the code in it when recording a macro.

# **Watch Window**

A watch is a variable or expression that has been placed in the window to enable you to monitor its value.  
Let’s we watch the values of variables and expressions as your code executes.  
When your application enters break mode, the watch expressions you select appear in a window allowing you to observe their values etc. It is also possible to set up conditional watches.  
This window is automatically updated after each line of code is executed.

### **Watches Window**

Although it is labelled as Watch Window the actual window displays Watches Window.

|  |
| --- |
| alt text |

**Expression** - Lists the watch expression with the Watch icon, on the left.  
**Value** - List the value of the expression at the time of the transition to break mode.  
**Type** - Lists the expression type.  
**Context** - Lists the context of the watch expression.  
You can close the window by clicking the Close box. If the Close box is not visible, double-click the Title bar to make the Close box visible, then click it.

This window shows all the watches that have been created.  
Let’s we pause the program when a variable value changes.

Appears automatically when watch expressions are defined in the project.  
You can change the size of the column headers by dragging its border to the right to make it larger or to the left to make it smaller.  
Close the window by clicking the Close box. If the Close box is not visible, double-click the Title bar to make the Close box visible, then click it.  
You can edit a value and then press ENTER, the UP-ARROW key, the DOWN ARROW key, TAB, SHIFT+TAB, or click somewhere on the screen to validate the change. If the value is illegal, the Edit field remains active and the value is highlighted. A message box describing the error also appears. Cancel a change by pressing ESC.  
If the context of the expression isn't in scope when going to break mode, the current value isn't displayed.

### Add A Watch

3 different types of watches  
highlight a variable and drag it straight in  
To add a watch select (Debug > Add Watch).

|  |
| --- |
| alt text |

**Expression** -  
**Context** - After adding a watch you can use these drop-downs to change the scope to "Full scope" if originally entered as local scope.  
**Watch Expression** - Adds the expression so the value can be watched during execution. Same as Local Window.  
**Break When Value is True** - Adds the expression so the value can be watched and execution will enter break mode if the value of the expression is true  
**Break When Value Changes** - Add the expression so the value can be watched and execution will enter break mode if the value of the expression changes

### Drag and Drop

Variables can be added by dragging and dropping them into the window. It is easy to watch individual variables.  
Drag a selected variable to the Immediate window or the Watch window.

### **Quick Watch**

This is a feature you can use to check the value of a variable or expression quickly while in break mode.  
Place the insertion point over the variable name and select (Debug > Quick Watch) or alternatively press (**Shift + F9**).  
Another way to get values for expressions and variables quickly is to enable the Auto Data Tips from the (Tools > Options)(Editor tab).  
With this feature enabled when you place the mouse over a variable or select an expression and place the mouse over the expression a tool tip will appear after a sort delay displaying the current value.

To view the value of variables during execution or break mode hover the cursor over the variable within the procedure or expression. Entire expressions can be evaluated by highlighting the expression with the cursor and hovering.

6. What is an immediate Window and what is it used for?

Answer - **To display the Immediate window**

* From the **View** menu, choose Immediate window (CTRL+G).

**To execute code in the Immediate window**

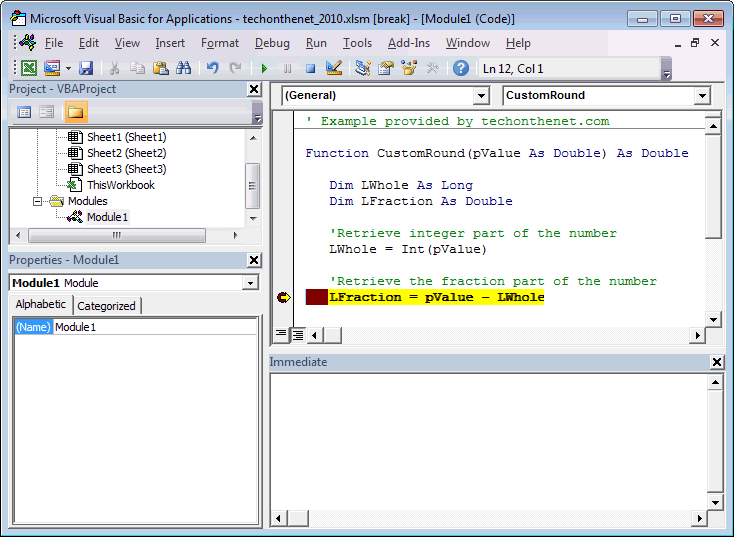
1. Type a line of code in the Immediate window.
2. Press ENTER to execute the statement.

Use the Immediate window to:

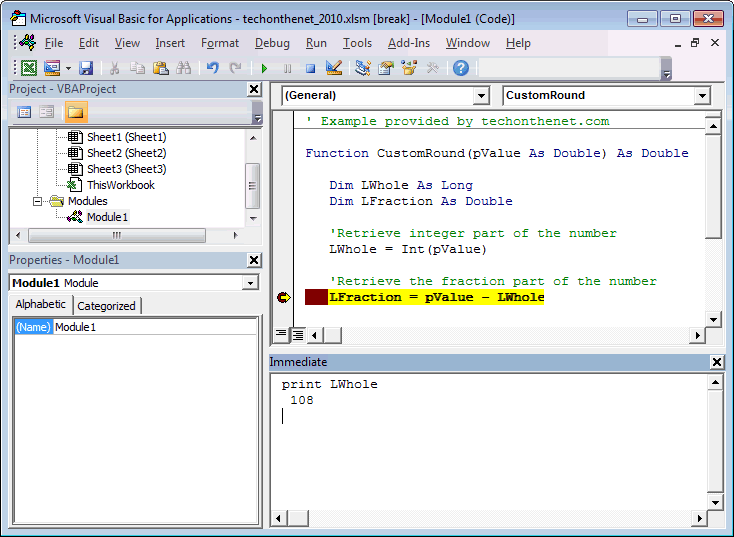
* Test problematic or newly written code.
* Query or change the value of a variable while running an application. While execution is halted, assign the variable a new value as you would in code.
* Query or change a property value while running an application.
* Call procedures as you would in code.
* View debugging output while the program is running.

## Using the Immediate Window

In Excel 2010, the Immediate window can be used to debug your program by allowing you to enter and run VBA code in the context of the suspended program.



We've found the Immediate window to be the most help when we need to find out the value of a variable, expression, or object at a certain point in the program. This can be done using the print command.

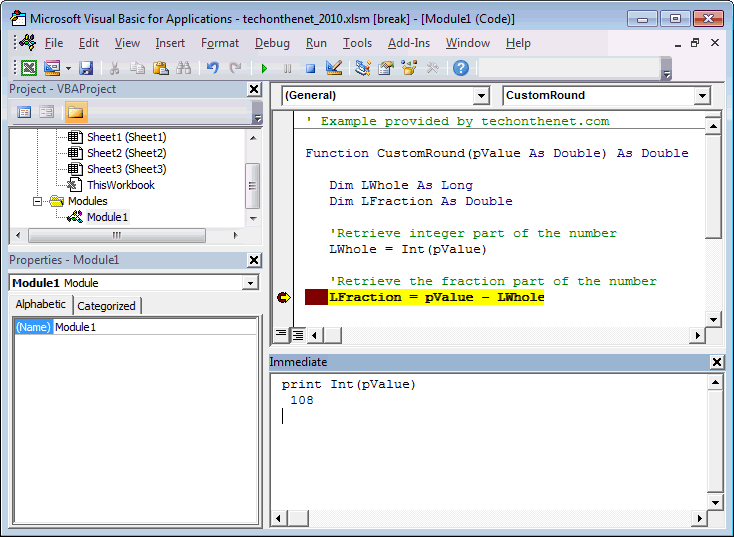


In this example, we typed **print LWhole** in the Immediate window and pressed ENTER.

print LWhole

The Immediate window displayed the result in the next line. In this case, the print LWhole command returned **108**.

You can also type more complicated expressions in the Immediate window. (Remember to press ENTER.) For example:



In this example, we typed **print Int(pValue)** in the Immediate window and pressed ENTER.

print Int(pValue)

The Immediate window displayed the result of **108** in the next line.

The Immediate window can be used to run other kinds of VBA code, but bear in mind that the Immediate window can only be used when debugging so any code that you run is for debugging purposes only. The code entered in the Immediate window does not get saved and added to your existing VBA code