ADVANCE EXCEL ASSIGNMENT 17

1. **What are modules in VBA and describe in detail the importance of creating a module?**

   A module is the fundamental syntactic unit of VBA source code. The physical representation of a module is implementation dependent but logically a VBA module is a sequence of Unicode characters that conform to the VBA language grammars.

   A module consists of two parts: a module header and a module body.

   The module header is a set of attributes consisting of name/value pairs that specify the certain linguistic characteristics of a module.

   A module body consists of actual VBA Language source code and most typically is directly written by a programmer.

   A VBA module is used to store any VBA code that you have written in the VBE (Visual Basic Editor). The modules are contained within a VBA Project and when the file is saved – be it an Excel workbook, Word document or Access database, the module or modules are saved within that file – that file is essentially the parent application of the module. Modules can also be exported out of the parent file and saved as their own individual files. This is useful when you want to re-use code in a different file, and therefore perhaps import that module into a new file.

2. **What is Class Module and what is the difference between a Class Module and a Module?**

   Class modules – this module is used to create objects at run time. Class modules are used by Advanced VBA programmers and will be covered at a later stage.

   When you insert a new module, you can see an option to insert a class module. But there's a slight difference between both modules. Class modules are special modules that can help you create your custom objects. You can also define methods, properties, and events for those objects. And when you create a new object from the class module, you can refer to it from the standard module as well.

   You can develop a robust building block which can be used in any number of different Excel applications.

   Once it is thoroughly tested, then is can be relied on to always produce the correct results in the same way as the built-in Excel objects. If updates are made to code elsewhere in the application, the new object will still continue to work in the same way. You can use your new object in other Excel applications as an add-in. The objects can be re-used in other applications and helps in debugging.

3. **What are Procedures? What is a Function Procedure and a Property Procedure?**

   A procedure is a block of Visual Basic statements enclosed by a declaration statement (Function, Sub, Operator, Get, Set) and a matching End declaration. All executable statements in Visual Basic must be within some procedure.

   A Function procedure is a series of Visual Basic statements enclosed by the Function and End Function statements. The Function procedure performs a task and then returns control to the calling code. When it returns control, it also returns a value to the calling code. Each time the procedure is called, its statements run, starting with the first executable statement after the Function statement and ending with the first End Function, Exit Function, or Return statement encountered. You can define a Function procedure in a module, class, or structure. It is Public by default, which means you can call it from anywhere in your application that has access to the module, class, or structure in which you defined it. A Function procedure can take arguments, such as constants, variables, or expressions, which are passed to it by the calling code.

A property procedure is a series of Visual Basic statements that manipulate a custom property on a module, class, or structure. Property procedures are also known as property accessors. Visual Basic provides for the following property procedures:

- A Get procedure returns the value of a property. It is called when you access the property in an expression.
- A Set procedure sets a property to a value, including an object reference. It is called when you assign a value to the property.

You usually define property procedures in pairs, using the Get and Set statements, but you can define either procedure alone if the property is read-only (Get Statement) or write-only (Set Statement). You can omit the Get and Set procedure when using an auto-implemented property. You can define properties in classes, structures, and modules. Properties are Public by default, which means you can call them from anywhere in your application that can access the property's container.

4. What is a sub procedure and what are all the parts of a sub procedure and when are they used?

A Sub procedure is a series of Visual Basic statements enclosed by the Sub and End Sub statements. The Sub procedure performs a task and then returns control to the calling code, but it does not return a value to the calling code. Each time the procedure is called, its statements are executed, starting with the first executable statement after the Sub statement and ending with the first End Sub, Exit Sub, or Return statement encountered. You can define a Sub procedure in modules, classes, and structures. By default, it is Public, which means you can call it from anywhere in your application that has access to the module, class, or structure in which you defined it. The term method describes a Sub or Function procedure that is accessed from outside its defining module, class, or structure.

A Sub procedure can take arguments, such as constants, variables, or expressions, which are passed to it by the calling code.

Declaration syntax:-

The syntax for declaring a Sub procedure is as follows:

```
 [modifiers] Sub SubName[(parameterList)]
    ' Statements of the Sub procedure.
End Sub
```

The modifiers can specify access level and information about overloading, overriding, sharing, and shadowing.

The syntax for each parameter in the parameter list is as follows:

```
[Optional] [ByVal | ByRef] [ParamArray] parameterName As DataType
```

Sub procedures are one of the major cornerstones of VBA. A Sub procedure does things. They perform actions such as formatting a table or creating a pivot table. The majority of procedures written are Sub procedures. All macros are Sub procedures.

5. **How do you add comments in a VBA code? How do you add multiple lines of comments in a VBA code?**

A VBA COMMENT is a green line of text that helps you to describe the written code. In simple words, a comment is a line of text which is not a code and VBA ignores it while executing the code. It's a good practice to add comments in your VBA codes.

Method 1: Steps you need to follow to add a comment in a VBA code:

a. First, click on the line where you want to insert the comment.
b. After that, type an APOSTROPHE using your keyboard key.
c. Next, type the comment that you want to add to the code.
d. In the end, hit enter to move to the new line and the comment will turn green.

Method 2: Use the comment block button from the toolbar. This button simply adds an apostrophe at the start of the line. To use this button, first of all, you need to select the line of the code and then click on the button.

Method 3: You can also use the keyword Rem. In order to comment a line, you need to put this keyword at the beginning of a line:

```
Rem   Sheet1.Range("A1").Value = "Test"
```

Similarly to comment button, the Rem keyword allows you to comment just a whole line of a code.

To add comments in multiple lines, we use multi-line comments when we have to add points in our description or the description is long. The easiest way is to select all the lines and then use the comment button from the toolbar or you can also add an APOSTROPHE at the starting of each line.