Excel Assignment - 17

***1. What are modules in VBA and describe in detail the importance of***

***creating a module?***

**modules in VBA**

* 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. While a module header could be directly written by a human programmer, more typically a VBA implementation will mechanically generate module headers based upon the programmer’s usage of implementation specific tools.
* A module body consists of actual VBA Language source code and most typically is directly written by a human programmer.
* VBA supports two kinds of modules, procedural modules and class modules, whose contents MUST conform to the grammar productions procedural-module and class-module.

**Importance of creating a module :-**

* Code Reusability: With VBA modules, We can write reusable code snippets that can be called from multiple places in our Excel application. It's like having a repertoire of cooking recipes that we can use in different dishes.
* Easier Debugging: By compartmentalizing our code into modules, we can isolate and troubleshoot issues more efficiently. It's like finding the exact screw that's causing our clock to malfunction instead of disassembling the entire mechanism.
* Improved Readability: VBA modules allow us to break down our code into smaller, more digestible sections. It's like dividing a complex novel into chapters, making it easier to follow the story.
* Flexible Execution: Modules enable us to control when and how our code executes. It's like having the power to turn on specific lights in our house at different times of the day, creating the perfect ambiance.

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

***Module and a Module?***

**Class modules** are like the superheroes of VBA programming, bringing object-oriented capabilities to our Excel projects.

They allow us to create custom objects with their own properties, methods, and events. Think of class modules as blueprints for creating unique objects that possess specific characteristics and abilities.

To understand class modules, let's step into the realm of fiction. Imagine we're a master toy maker. W design and build different types of toys, each with its own distinct features, functions, and personality. Each toy is a unique instance, created based on a specific blueprint. Similarly, class modules in VBA allow us to define the characteristics and behaviors of our custom objects.

Diﬀerence between a Class Module and a Module :-

Modules: VBA modules are like independent units of code that store procedures, functions, and variables. They act as containers for code segments and provide a means to organize and manage our Excel macros and functions. It's like having individual compartments in a toolbox, each holding specific tools for different tasks.

Class Modules: Class modules, on the other hand, take the concept of code organization to the next level. They allow us to define custom objects with their own properties, methods, and events. Class modules introduce the principles of object-oriented programming (OOP) into your Excel projects, enabling us to create sophisticated applications with modular and reusable code. It's like having a factory where you can design and manufacture unique products, each with its own specifications and functionalities.

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

**Procedures :-**

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.

Procedures are useful for performing repeated or shared tasks, such as frequently used calculations, text and control manipulation, and database operations.We can call a procedure from many different places in our code, so you can use procedures as building blocks for your application.

Structuring our code with procedures gives you the following benefits:

Procedures allow us to break your programs into discrete logical units. we can debug separate units more easily than we can debug an entire program without procedures.

After we develop procedures for use in one program, we can use them in other programs, often with little or no modification. This helps us avoid code duplication.

**Function procedures :-**

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.

We can define a Function procedure in a module, class, or structure. It is Public by default, which means we can call it from anywhere in our application that has access to the module, class, or structure in which we defined it.

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

**Property Procedures :-**

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 we access the property in an expression.

A Set procedure sets a property to a value, including an object reference. It is called when we assign a value to the property.

We usually define property procedures in pairs, using the Get and Set statements, but we can define either procedure alone if the property is read-only (Get Statement) or write-only (Set Statement).

We can omit the Get and Set procedure when using an auto-implemented property.

We can define properties in classes, structures, and modules. Properties are Public by default, which means We can call them from anywhere in our application that can access the property's container.

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

Same As Above

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

**Sub procedures :-**

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.

We can define a Sub procedure in modules, classes, and structures. By default, it is Public, which means We can call it from anywhere in your application that has access to the module, class, or structure in which we defined it. The term method describes a Sub or Function procedure that is accessed from outside its defining module, class, or structure. For more information, see Procedures.

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

**Parts of a sub procedure :**

1. **Sub procedure-name**

Sub procedure\_name()  
.  
.  
End Sub

Sub procedure-name is the declaration statement and the first line of the procedure. This is paired with the End Sub statement. All code statements must be written between these two statements

2. **Argument\_list :** If arguments or parameters are included they operate in the same way as function arguments.

3. **Public | Private :**

Public:

The procedure will be listed on the Excel Macro list.

The procedure is available to all modules in the project.

Private :

The procedure is not listed in the Excel Macro dialog box

The procedure is only available to other procedures in the same module.

4. **Exit Sub :**

Normally associated with logical statements or error checking. Allows code execution to terminate, and the remaining code statements are not run.

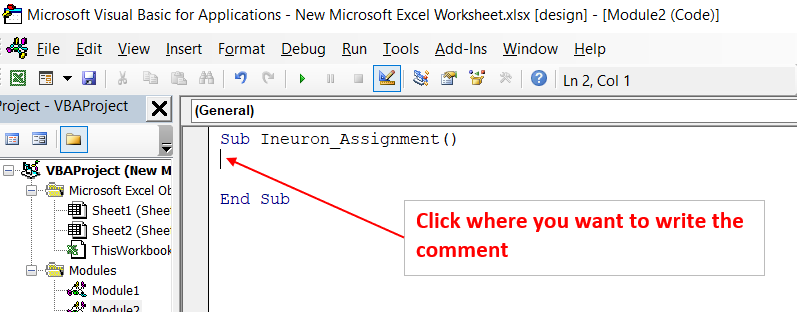
***6. How do you add comments in a VBA code? How do you add multiple lines of comments in a VBA code?***

**VBA Comments :**

Comments are the lines in the code that are ignored while executing the code. These are represented as green text in the code. The comments help describe the written code. Knowing the correct use of comments is very important because while working with long and complex code, comments help us identify which part of code does what. It is very helpful for development purposes.

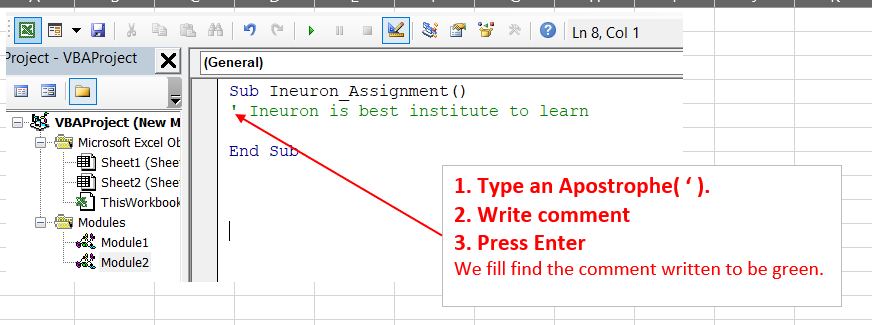
**Adding Single Line comment in VBA :**

Step 1: Click on the line where you want to insert a comment.



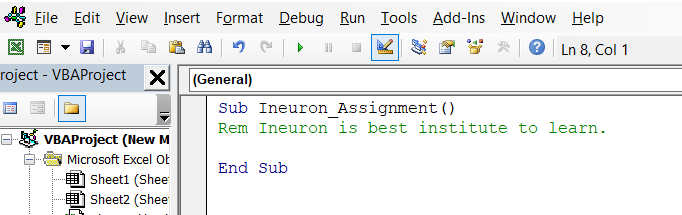
Step 2: Type an Apostrophe( ‘ ) at the start of a line and Write the comment we want.

Step 3: Press Enter and we fill find the comment written to be green.



**Use Rem to Comment**

At the start of the comment use the keyword Rem to comment on a line.



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

***Same as Above.***