ADVANCE EXCEL ASSIGNMENT 21

1. **Write a VBA code to enter your name in A1 Cell using Input Box and once you enter the name display a message box that says the name has been entered.**

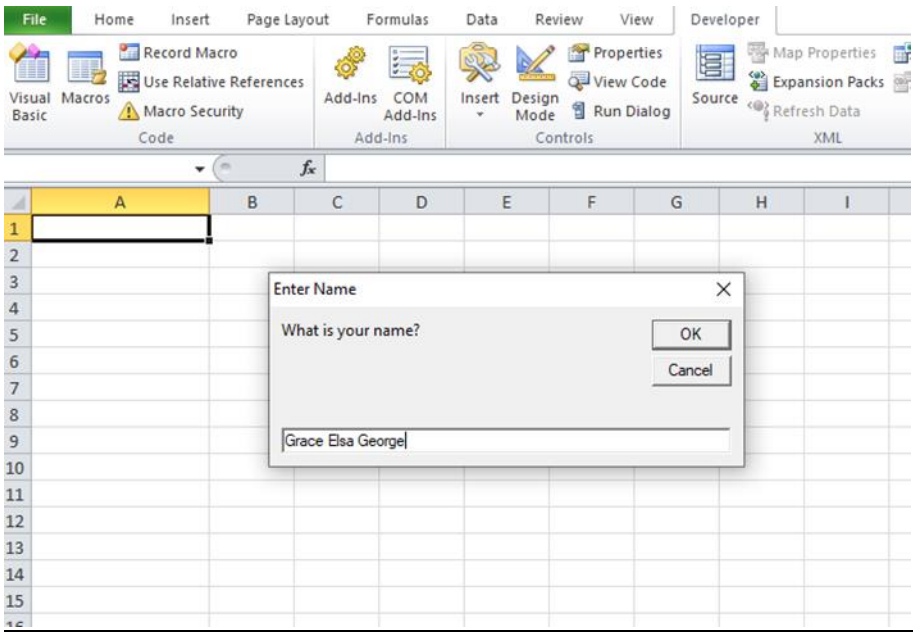
VBA Code:

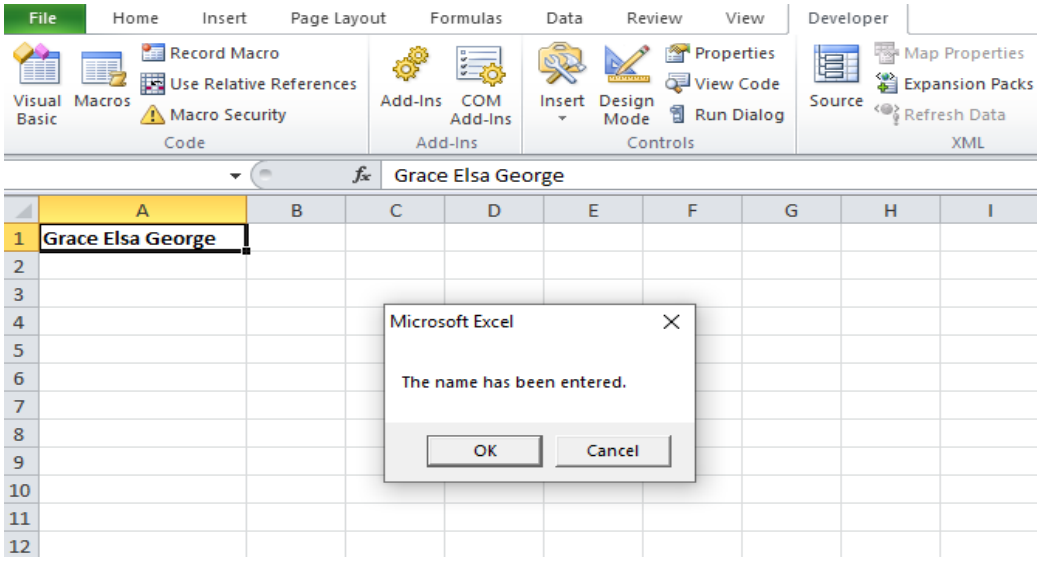
excelass21Q1.xlsm - ThisWorkbook (Code)

(General) | vba_input_box

```
Sub vba_input_box()

'variable to store the count of rows.
Dim iRow As Long

'get the count cells that are filled
iRow = WorksheetFunction.CountA(Range("A:A"))

'get to the next blank cell in column A
Cells(iRow + 1, 1).Select

'enter the value from the input box into the active cell
ActiveCell = InputBox("What is your name?", "Enter Name")
MsgBox "The name has been entered.", vbOKCancel

End Sub
```

Output:





2. **What are Userforms? Why are they used? How to fill a list box using for loop?**

A UserForm object is a window or dialog box that makes up part of an application's user interface. Userform in VBA are customized user-defined forms made to take input from a user in a form format. Although it has different sets of controls to add, such as text boxes, checkboxes labels, etc., to guide a user to input a value and store the value in the worksheet, every part of the UserForm has a unique code with it.

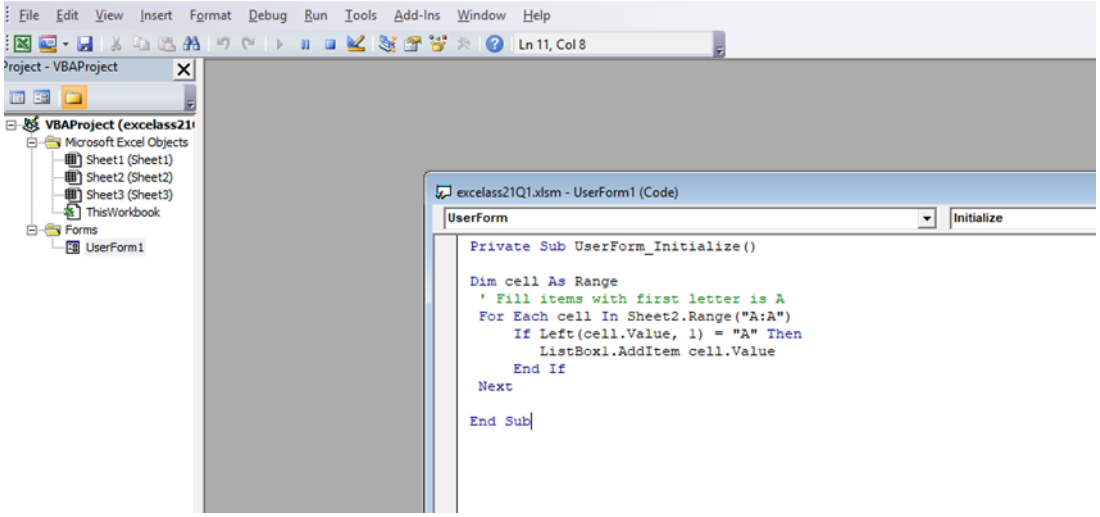
The UserForms collection is a collection whose elements represent each loaded UserForm in an application. The UserForms collection has a Count property, an Item method, and an Add method. Count specifies the number of elements in the collection; Item (the default member) specifies a specific collection member; Add places a new UserForm element in the collection.

Syntax
UserForm UserForms [.Item] (index)

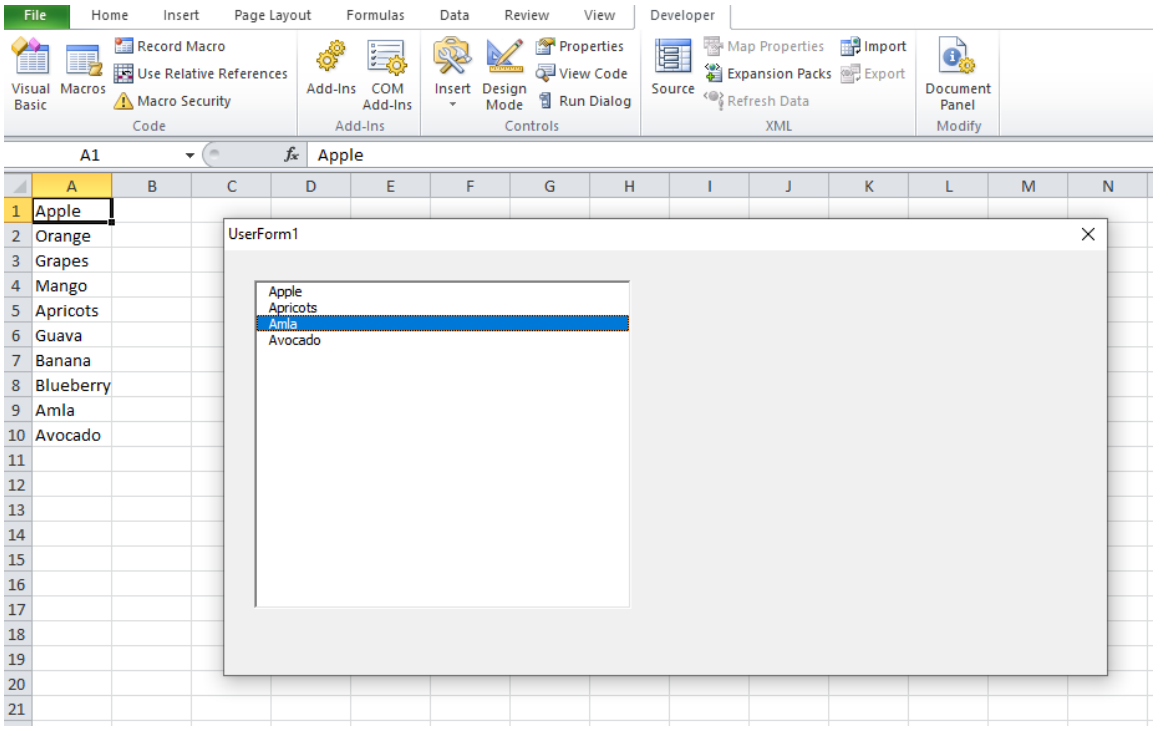
The placeholder index represents an integer with a range from 0 to UserForms.Count - 1. Item is the default member of the UserForms collection and need not be specified.

The VBA ListBox is a very useful control. If you are creating any kind of UserForm application you will most likely use it. The ListBox is used to display a list of items to the user so that the user can then select one or more. The ListBox can have multiple columns and so it is useful for tasks like displaying records.

VBA code to fill the listbox using for loop:



Output:



3. **What is an array? Write a VBA code to enter students and their marks from the below table.**

A VBA array is a type of variable. It is used to store lists of data of the same type. An example would be storing a list of countries or a list of weekly totals.

In VBA a normal variable can store only one value at a time. In the following example we use a variable to store the marks of a student:

```
' Can only store 1 value at a time
Dim Student1 As Long
Student1 = 55
```

If we wish to store the marks of another student then we need to create a second variable. In VBA, an array is a variable that can store multiple values. You can access all the values from that array at once or you can also access a single value by specifying its index number which is the position of that value in the array. You can store all this information in an array, not just for one student but for hundreds.

Static array declaration example:

```
Dim Students(1 To 3) As Long
```

Dynamic array declaration example:

```
Dim Students () As Long
```

To create a multiple-dimensional array, you need to define the dimensions while declaring the array. Well, you can define as many dimensions as you need (VBA allows 60 dimensions) but you will probably not need to use more than 2 or 3 dimensions of any of the arrays. Using a two-dimensional array is like having rows and columns.

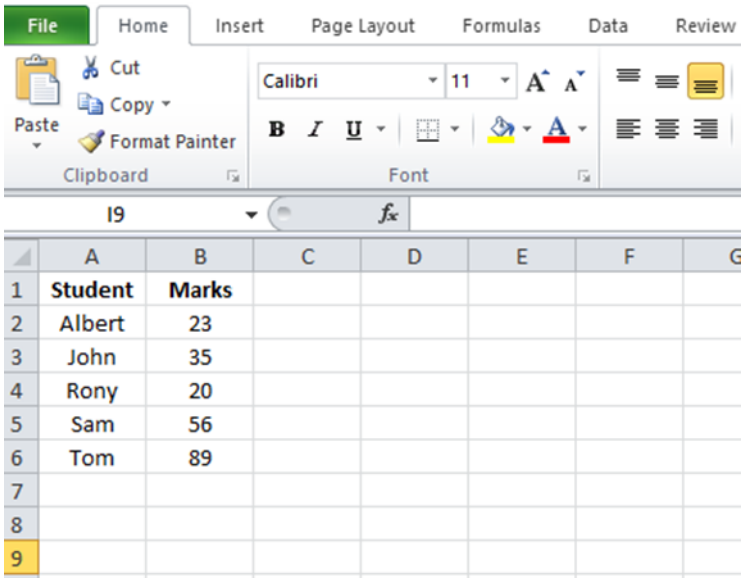
<u>Create a Multi-Dimensional Array in VBA</u>

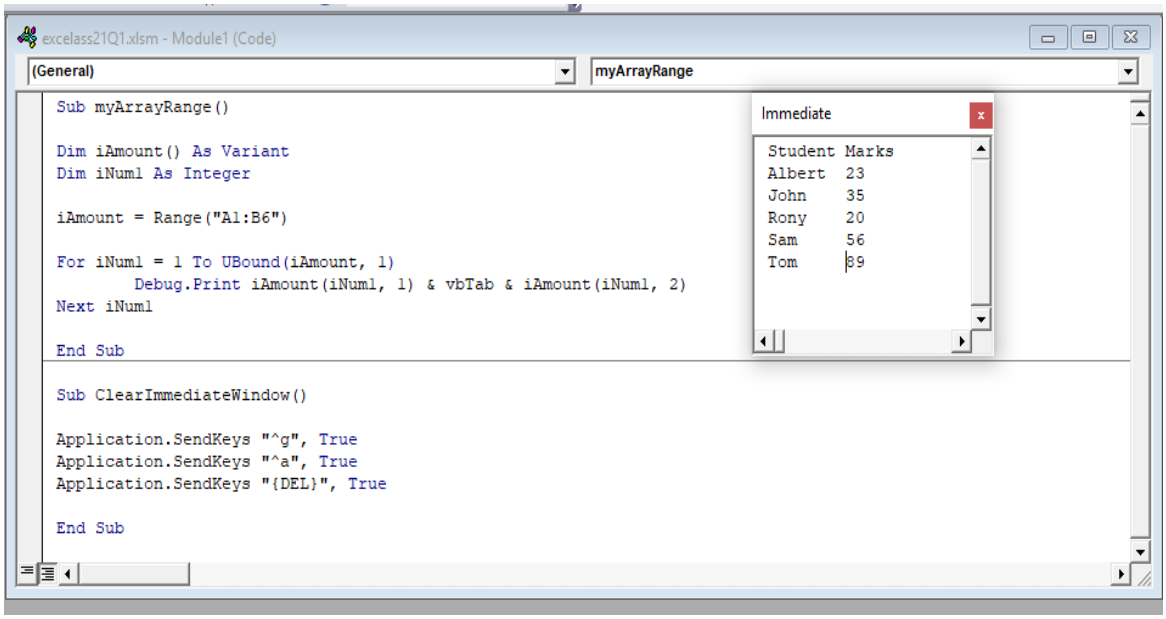
Use the Dim statement to declare the array with the name that you want to give. After that, enter a starting parenthesis and define the element count for the first dimension. Next, type a comma and enter a count of elements that you want to have in the second dimension, and close the parentheses. In the end, define the data type for the array as a variant or any data type you want.

```
Dim myArray(5, 2) As Variant
```

The above code for the array creates an array with 5 rows and 2 columns.

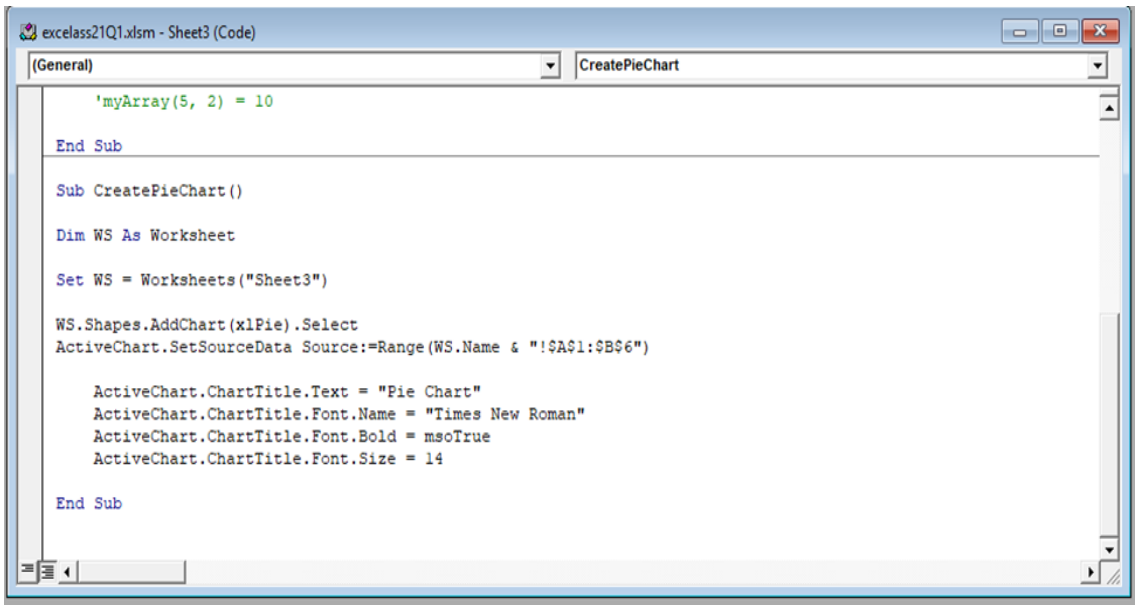
<u>VBA code to enter students and their marks from the below table:</u>



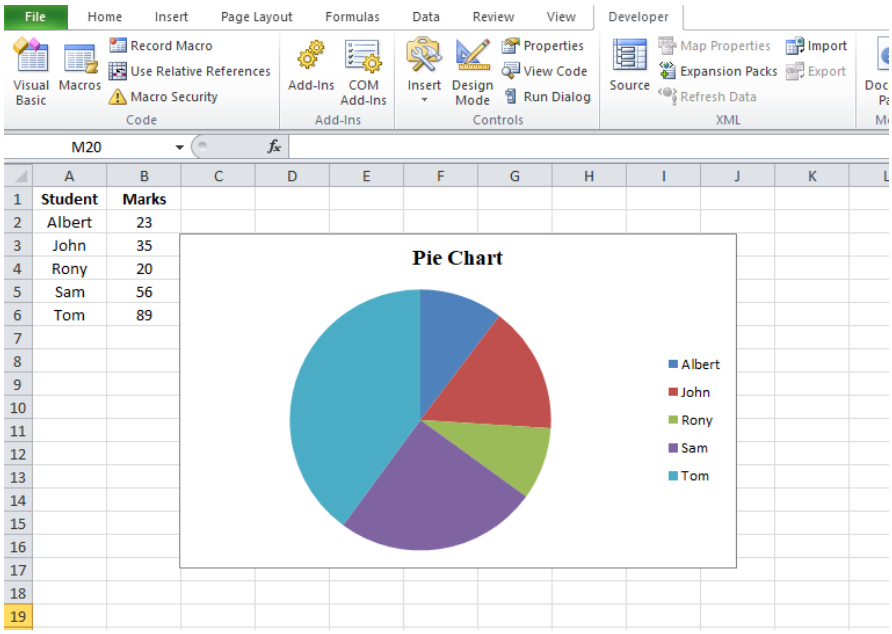


4. **Use the following data to create a pie chart using VBA code. Use Font - 'Times new Roman', Size -14, Bold, Title - Pie chart' and you are per to use colours as per your taste.**

<u>VBA code to create a pie chart for student table from question 2:</u>



Output:



5. **Check the dataset in the link given below and create a pivot table using VBA showing the sales for the year from stationary category.**

https://docs.google.com/spreadsheets/d/1IRSEnmgz8Ro276GslknRNk0zlrB5CZH1YrnT71kqFM/edit?usp=sharing

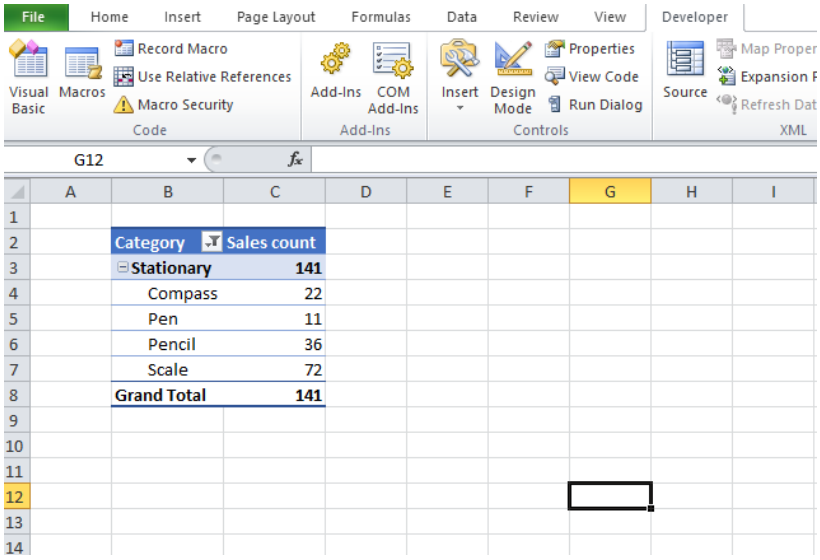
VBA Code to create the pivot table:

```
Sub InsertPivotTable()

'Declare Variables
    Dim PSheet As Worksheet
    Dim DSheet As Worksheet
    Dim PCache As PivotCache
    Dim PTable As PivotTable
    Dim PRange As Range
    Dim LastRow As Long
    Dim LastCol As Long

'Insert a New Blank Worksheet
    On Error Resume Next
    Application.DisplayAlerts = False
    Worksheets("PivotTable").Delete
    Sheets.Add Before:=ActiveSheet
    ActiveSheet.Name = "PivotTable"
    Application.DisplayAlerts = True
```

```
    Set PSheet = Worksheets("PivotTable")
    Set DSheet = Worksheets("Dataset")

'Define Data Range
    LastRow = DSheet.Cells(Rows.Count, 1).End(xlUp).Row
    LastCol = DSheet.Cells(1, Columns.Count).End(xlToLeft).Column
    Set PRange = DSheet.Cells(1, 1).Resize(LastRow, LastCol)

'Define Pivot Cache
    Set PCache = ActiveWorkbook.PivotCaches.Create _
            (SourceType:=xlDatabase, SourceData:=PRange). _
            CreatePivotTable(TableDestination:=PSheet.Cells(2, 2), _
            TableName:="SalesPivotTable")

'Insert Blank Pivot Table
    Set PTable = PCache.CreatePivotTable _
            (TableDestination:=PSheet.Cells(1, 1), TableName:="SalesPivotTable")

'Insert Row Fields
    With ActiveSheet.PivotTables("SalesPivotTable").PivotFields("Category")
            .Orientation = xlRowField
            .Position = 1
    End With

    With ActiveSheet.PivotTables("SalesPivotTable").PivotFields("Category")
          .PivotItems("Footwear").Visible = False
    End With

    With ActiveSheet.PivotTables("SalesPivotTable").PivotFields("Product")
            .Orientation = xlRowField
            .Position = 2
    End With

'Insert Data Field
    With ActiveSheet.PivotTables("SalesPivotTable").PivotFields("Product")
            .Orientation = xlDataField
            .Function = xlCount
            .NumberFormat = "#,##0"
            .Name = "Sales count "
    End With
    ActiveSheet.PivotTables("SalesPivotTable").CompactLayoutRowHeader =
    "Category"

'Format Pivot Table
```

```
    ActiveSheet.PivotTables("SalesPivotTable").ShowTableStyleRowStripes = True
    ActiveSheet.PivotTables("SalesPivotTable").TableStyle2 = "PivotStyleMedium9"

'to show the pivot table in Tabular form
    pvsheet.PivotTables("SalesPivotTable").RowAxisLayout xlTabularRow

    Application.DisplayAlerts = True
    Application.ScreenUpdating = True
End Sub
```

Output:



6. **Write step by step procedure to protect your workbook using a password.**

Step 1: Select the Sheet which needs to be protected

The first step is to decide which sheet we need to protect using a password to protect the sheet. Next, we need to call the sheet by name using the VBA Worksheet Object. For example, assume you want to protect the "Master Sheet" sheet, then you need to mention the worksheet name below.

Step 2: Define Worksheet Variable

After mentioning the worksheet name, put a dot, but we don't see any IntelliSense list to work with. So, it makes the job difficult. To access the IntelliSense list, define the variable as a worksheet.

Code:

```
Sub Protect_Example1()
  Dim Ws As Worksheet
End Sub
```

Step 3: Give Worksheet Reference

Now, set the worksheet reference to the variable as Worksheets("Master Sheet").

Code:

```
Sub Protect_Example1()
  Dim Ws As Worksheet
  Set Ws = Worksheets("Master Sheet")
End Sub
```

Now, the variable “Ws” holds the reference of the worksheet named “Master Sheet.” By using this variable, we can access the IntelliSense list.

Step 4: Select Protect Method

Select the “Protect” method from the IntelliSense list.

Step 5: Enter Password

Specify the password in double-quotes.

Code:

```
Sub Protect_Example1()
  Dim Ws As Worksheet
  Set Ws = Worksheets("Master Sheet")
  Ws.Protect Password:="MyPassword"
End Sub
```

Step 6: Run the Code

Run the code manually or use the shortcut key F5. Then, it will protect the sheet named “Master Sheet.”

When the sheet is protected, if we want to modify it, it shows an error message.

We need to use loops if you wish to protect more than one sheet. Below is the example code to protect the sheet.

```
Sub Protect_Example2()
 Dim Ws As Worksheet

 For Each Ws In ActiveWorkbook.Worksheets
 Ws.Protect Password:="My Passw0rd"
 Next Ws

End Sub
```