

VBA Training

VBA stands for visual basic for application. VBA is a programming language to use automate manual task in any office applications(Excel, Access, Outlook, Word). Here we will discuss about excel vba. To automate any task in excel we have to write script which is called macro. We can also design interactive form to automate task. Macro is created in module, Sheet, Class module or workbook editor. which is called VBE. VBE stands for visual basic editor. Most of the script written in module.

Basic

We are going to cover some one liner tasks in vba. Which will be helpful to create script to automate any assignment. We start writing code with Sub means Subroutine

```
Sub BasicMacro()  
    write any script here  
End Sub
```

End sub is closing of Sub

Add New Sheet

Sheets.add

Sheets.add after:=Sheets("Sheet3")

Sheets.add before:=Sheets("Sheet3")

Sheets.add after:=Sheets(Sheets.Count)

Activate Any Sheet

Sheets("Employee").activate

Sheets("Sheet4").select

Delete the sheet

Sheets("Employee").Delete

Application.Displayalert=False

Sheets("Employee").Delete

Application.Displayalert=True

Macro speed up

```
Application.ScreenUpdating = False
```

Rename the Sheet

```
Sheets("Sheet3").Name="Employee"
```

Hide and Unhide sheet

```
Sheets("Sheet4").Visible = False
```

```
Sheets("Sheet47").Visible = xlSheetHidden
```

```
Sheets("Sheet44").Visible = xlSheetVeryHidden
```

```
Sheets("Sheet4").Visible = True
```

Find Word

```
Range("A1:A10").Find("Mohan").Select
```

```
Range("A1:A10").Find(What:="Mohan", MatchCase:=True, Lookat:=xlWhole).Select
```

Delete row, rows, column, columns

`Range("C1").EntireColumn.Delete`

`Range("A4").EntireRow.Delete`

`Range("A2,A4,A6,A8").EntireRow.Delete`

Task

`Range("B1:E1").EntireColumn.Delete`

`Range("A4:A6").EntireRow.Delete`

Input value in cell and Environ function

Range("A1") = "System Id"

Range("A2").Value = Environ("username")

Range("A3").Value = Environ("userprofile")

Pick path of this workbook

Range("A1")=Thisworkbook.path

Range("A2")=Thisworkbook.fullname

Autofit cell, row, rows, column, columns

Range("A1").entireColumn.AutoFit

Range("A1:A10").EntireRow.AutoFit

Autofill

Range("F2") = "=Sum(B2:E2)"

Range("F2").AutoFill Range("F2:F7")

Copy and Paste

`Range("A1").copy Range("H1")`

`Range("A1:A10").copy Range("H1")`

`Sheets("Employee").range("A1:A10").copy Sheets("Sheet10").range("H1")`

CurrentRegion:- Currentregion is used to cover all those data which is in same group

`Range("A1").CurrentRegion.Copy Range("M10")`

Used Range:-Used range is used to cover all the data from sheet.

`Sheets("Employee").usedrange.copy Sheets("Sheet47").range("A1")`

Add New workbook and save as

Workbooks.Add

ActiveWorkbook.SaveAs "C:\Users\skum254\Documents\Test\MyFirstWb.xlsx"

Workbooks("MyFirstWb.xlsx").Close TRUE

Workbooks.Add

ActiveWorkbook.SaveAs "C:\Users\skum254\Documents\Test\MyFirstWb.xlsm",

xlOpenXMLWorkbookMacroEnabled

Range("A1") = "Test"

Workbooks("MyFirstWb.xlsm").Close True

Open existing workbook and save as

```
Workbooks.Open ("C:\Users\skum254\Documents\Test\Executive.xlsx")  
ActiveWorkbook.SaveAs ("C:\Users\skum254\Documents\Test\Manager.xlsx")  
Range("A1") = "Name"  
Range("B1") = "Salary"  
Windows("Manager.xlsx").Close True
```

Save as in pdf

```
Sheets("Sheet15").ExportAsFixedFormat xlTypePDF, ThisWorkbook.FullName
```

Close Application

```
Application.Quit
```

Offset:-Offset is used to move pointer (Left, Right, Up, Down) as per requirement.

`Range("H10").offset(1,0).select`

`Range("H10").offset(-1,0).select`

`Range("H10").offset(0,1).select`

`Range("H10").offset(0,-1).select`

Find blank row

```
Range("A1").end(xldown).offset(1,0).select  
Range("A1").end(xltoRight).offset(0,1).select  
Range("A10000").end(xlup).offset(1,0).select  
Range("XFD1").end(xlToLeft).offset(0,1).select
```

Dim Rcount as integer

```
Rcount=worksheetfunction.Counta(Range("A:A"))+1
```

or

```
Rcount=Range("A10000").end(xlup).row+1  
Range("A"&Rcount).select
```

Find blank column

Dim Lcol as integer

Lcol =worksheetfunction.Counta(Range("1:1"))+1

Cells(1,Lcol).select

Or

Range("A1").offset(1, Lcol).select

Special Cell

Range("A1:A32").SpecialCells(xlCellTypeBlanks).Select

Timer to stop code for specific time interval

Application.Wait (Now + TimeValue("00:00:10"))

Variable

Variables are used to store some values in computer memory or storage system and those stored values are used anywhere in code. We need to declare variable before using variable in code if we have applied option explicit in module. Declaration starts with Dim keyword means dimension.

Option Explicit:- Option explicit makes variable declaration mandatory if we use variable in our code.

Option Implicit:- Without writing option explicit is option implicit. In option implicit we can use variable without declaration. Data type of the variable is variant in option implicit.

Data Type

Data type is used to restrict your storage limitation. Data type makes your error reduction because of restriction in storage type. We have number of data types in vba that will be explained here.

Variable and data type is dependent on each other as if we declare variable then we need to put data type otherwise it takes as variant data type. Variant data type is global data type to store any types of values (String, Number, Date etc.)

Variable declaration with data type:-

Byte:- Byte data type is for numeric data type to store maximum 255 numeric value.

```
Dim I As Byte
```

```
I = 255
```

```
MsgBox I
```

```
Range("M1") = I
```

```
Range("M1:M10") = I
```

Integer:- Integer data type is for numeric data type to store minimum -32768 to 32767 whole number. which is 2 bytes storage limit.

```
Dim I As Integer
```

```
I = 32767
```

```
MsgBox I
```

Long:- Long data type is for numeric data type to store approx 9 digit whole number. which has 4 bytes storage limit.

```
Dim I As Long
```

```
I = 1987635
```

```
MsgBox I
```

Single:- Single data type is used to store number with decimal places to round 4 decimal places. Storage limit is 4 bytes in single data type.

```
Dim I As Single
```

```
I = 345.987
```

```
MsgBox I
```

Double:- Double data type is used to store number with decimal places to round 12 decimal places. Storage limit is 8 bytes in double data type.

```
Dim I As Double
```

```
I = 2345.3957384
```

```
MsgBox I
```

String:- String data type stores only text value.

```
- Dim I As String
```

```
I = "Ram"
```

```
I = "Ram10000"
```

```
I = "10000"
```

```
MsgBox I
```

```
Range("M1:M30") = I
```

Date:- Date data type stores date in variable. # is mandatory to write before and end of date to store any date.

```
Dim DOJ As Date
```

```
DOJ = #10/1/2020#
```

```
MsgBox DOJ
```

Variant:- Variant is global data type to store any type of data. If we are not aware about data to store in variable then need to used variant data type.

```
Dim I As Variant
```

```
Dim I
```

```
I = 90
```

```
I = "Ram"
```

```
I = 234.237
```

```
I = #1/1/2020#
```

```
Range("M1:M10") = I
```

```
MsgBox I
```


Boolean:- Boolean is logical data type where we can store logical value TRUE or False.

```
Dim MyVal As Boolean
if Range("A1")>0 then
    MyVal=True
else
    MyVal=False
end if
msgbox "Logic of this code " & MyVal
```

Object:- An object is element of any application. Such as Application, Workbook, Worksheet, Range etc.

```
Dim Rng as Range
Set Rng=Range("A1:A10")
Rng.value=10
```

```
Dim Ws As Worksheet
Set Ws = Sheets("Sheet2")
Ws.Activate
```

```
Dim Wb as Workbook
Set Wb=Workbooks("Training.xlsx")
Wb.Activate
```

Record Macro

Recording a macro in VBA (Visual Basic for Applications) allows you to automate a series of actions in applications like Microsoft Excel, Word, or PowerPoint.

Loops In VBA

We have number of loops in vba or any of the programming language. Which is useful to repeat any assignment until condition is true or false. We have following loops in vba:-

- For

- For Each

- Do While

- Do Until

We will discuss here one by one loop.

For Loop

For loop is useful to repeat control. Ending of loop is next. Basic structure of for loop is as below:-

```
Sub ForLoopExample()
```

```
Dim I as integer
```

```
For I=1 to 10
```

```
Next I
```

```
end sub
```

Option Explicit

Sub SelectCells()

 Sheets("Sheet11").Select

 Dim Counter As Byte

 For Counter = 1 To 10

 Range("A"&Counter).select

 Cells(Counter, 1).Select

 Range("A1").offset(Counter-1,1).select

 Cells(Counter,Counter).select

 Cells(Counter, 1) = Counter

 Next Counter

End Sub

```
Sub Print10to1()
```

```
'TASK
```

```
    Dim I,R As Integer
```

```
    Sheets("Sheet11").Select
```

```
    Range("C1").Select
```

```
    R=1
```

```
    For I = 10 To 1 Step -1
```

```
        Range("A"&R)=I
```

```
        R=R+1
```

```
    Next I
```

```
End Sub
```

```
Sub SheetsAdd()
```

```
'TASK
```

```
Dim i As Byte
```

```
For i = 1 To 5
```

```
    Sheets.Add after:=Sheets(Sheets.Count)
```

```
Next
```

```
End Sub
```

```
Sub ColorCells()
```

```
'TASK
```

```
Sheets("sheet8").Select
```

```
Dim i As Integer
```

```
For i = 1 To 55
```

```
    Cells(i, 1).Interior.ColorIndex = i
```

```
    Cells(i, 1) = i
```

```
Next
```

```
End Sub
```

```
Sub TransposeDataRowToColumnAndColumnToRow
```

```
'TASK
```

```
End Sub
```

```
Sub TransposeDataRowToColumnAndRowToColumn
```

```
'TASK
```

```
End Sub
```



```
Sub PrintGrade()  
    Sheets("Student").Select  
    Dim Counter As Integer  
    Range("G1") = "Grade"  
    For Counter = 2 To 101  
        Range("F" & I) = WorksheetFunction.Sum(Range("B" & I & ":" & "E" & I))  
  
        If Range("F" & Counter) > 300 Then  
            Range("G" & Counter) = "A"  
        ElseIf Range("F" & Counter) > 250 Then  
            Range("G" & Counter) = "B"  
        ElseIf Range("F" & Counter) > 200 Then  
            Range("G" & Counter) = "C"  
        Else  
            Range("G" & Counter) = "Fail"  
        End If  
    Next Counter  
End Sub
```

```
Sub ConcatenateFnameAndLname()
```

Task

```
    Sheets("Sheet11").Select
```

```
    Dim I As Integer
```

```
    For I = 2 To 7
```

```
        Range("C" & I) = Range("A" & I) & " " & Range("B" & I)
```

```
    Next I
```

```
End Sub
```

Sub ConditionalFormattingAndDataFilter()

Sheets("Sheet10").Select

Dim I, J, R, S, T As Byte

For I = 1 To 20

For J = 1 To 8

If Cells(I, J) > 5000 Then

Cells(I, J).Interior.ColorIndex = 4

Cells(R, 10) = Cells(I, J)

R = R + 1

Elseif Cells(I, J) > 3000 Then

Cells(I, J).Interior.ColorIndex = 6

Cells(S, 11) = Cells(I, J)

S = S + 1

Else

Cells(I, J).Interior.ColorIndex = 3

Cells(S, 12) = Cells(I, J)

T = T + 1

End If

Next J

Next I

End Sub

Do until loop

Do until will execute the block of code only if the condition evaluates true.

```
Sub Print1to10000ScreenUpdateUse()  
  Sheets("Sheet14").Select  
  Dim I As Integer  
    I = 1  
    Range("A1") = "Data"  
  Do Until I = 10001  
    Cells(I + 1, 1) = I  
    I = I + 1  
  Loop  
End Sub
```

```
Sub UseofDoLoopUntil()  
  Sheets("Sheet14").Select  
  Dim I As Integer  
  I = 1  
  Do Until I = 11  
    Cells(I, 1).Select  
    I = I + 1  
  Loop  
End Sub
```

```
Sub CopyDataMoreThan30000Salary()  
  Sheets("Sheet10").Select  
  Dim I, K As Integer  
  I = 2  
  K = 1  
  Do Until IsEmpty(Range("A" & I))  
    If Range("G" & I) > 30000 Then  
      Range("M" & K) = Range("A" & I)  
      K = K + 1  
    End If  
    I = I + 1  
  Loop  
End Sub
```

```
Sub ConcatenateTwoColumnsValue()  
Sheets("Sheet14").Select  
Range("D2").Select  
Do Until IsEmpty(ActiveCell.Offset(0, -3))  
    'Range("D" & I) = Range("A" & I) & " " & Range("C" & I)  
  
    Cells(I, 4) = Cells(I, 1) & " " & Cells(I, 2)  
    I = I + 1  
  
    ActiveCell.Value = ActiveCell.Offset(0, -3) & " " & ActiveCell.Offset(0,  
-1)  
    ActiveCell.Offset(1, 0).Select  
Loop  
End Sub
```

```
Sub GradeUSESelectCase()  
Sheets("Student").Select  
Dim I As Integer  
I = 2  
Do Until IsEmpty(Range("A" & I))  
    Select Case Range("F" & I)  
        Case 300 To 400  
            Range("H" & I) = "A"  
        Case 250 To 299  
            Range("H" & I) = "B"  
        Case 200 To 249  
            Range("H" & I) = "C"  
        Case Else  
            Range("H" & I) = "Fail"  
    End Select  
    I = I + 1  
Loop  
End Sub
```


Do while loop

Do while loop will execute the block of code only if condition evaluates false

```
Sub ExampleofDowhile()  
    Sheets("Student").Select  
    Dim I As Integer  
    I = 1  
    Do While I < 10  
        Cells(I, 1).Select  
        I = I + 1  
    Loop  
End Sub
```

```
Sub Print1to10000ScreenUpdateUse()  
Dim Counter As Integer  
Counter = 1  
Do While Counter <= 10  
    Cells(Counter, 1) = Counter  
    Cells(1, Counter) = Counter  
    Cells(Counter, Counter) = Counter  
    Counter = Counter + 1  
Loop  
End Sub
```

```
Sub Print10to1()  
Dim Counter As Integer  
Counter = 1  
Do While Counter <= 10  
    Cells(Counter, 1) = 11 - Counter  
    Cells(1, Counter) = 11 - Counter  
    Cells(Counter, Counter) = 11 - Counter  
    Counter = Counter + 1  
Loop  
End Sub
```

```
Sub UseofIsEmpty()  
  Sheets("Sheet9").Select  
  Dim I As Integer  
  I = 1  
  Do While Not IsEmpty(Range("A" & I))  
    Range("C" & I) = Range("A" & I)  
    I = I + 1  
  Loop  
End Sub
```

```
Sub FileNameInOurSheet()  
Dim Fname, Fpath As String  
Dim K As Integer  
K = 1  
  
Fpath = "C:\Users\shishir kumar\Desktop\TEST\  
Fname = Dir(Fpath & "*.xlsx")  
    Do While Fname <> ""  
        Cells(K, 2) = Fname  
        K = K + 1  
        Fname = Dir()  
    Loop  
End Sub
```

For Each

For each loop works with object collection. Such as workbooks, worksheets, Range etc. Objects are as:-

Application—Workbook—Worksheet—Range--Cells

```
Sub SelectCellWithRng()  
  Sheets("Sheet9").Select  
  Dim Rng As Range  
  For Each Rng In Range("A1:A10")
```

```
    Next Rng  
End Sub
```

```
Sub SelectCellWithRng()  
  Sheets("Sheet9").Select  
  Dim Rng As Range  
  Dim I as integer  
  I=1  
  For Each Rng In Range("A1:A10")  
    Rng.Select  
    Rng.Value=I  
    I=I+1  
  Next Rng  
End Sub
```

```
Sub RngCopyDataOnecolumnToAnother()  
  Sheets("Sheet18").Select  
  Dim Rng As Range  
  For Each Rng In Range("A1:A19")  
    Rng.Offset(0, 3) = Rng.value  
  Next Rng  
  MsgBox "Done"  
End Sub
```



```
Sub RngCopyDataOnCondition()  
  Sheets("Sheet18").Select  
  Dim Rng As Range  
  Dim i As Byte  
  i = 1  
  For Each Rng In Range("A1:A19")  
    If Rng.value > 50 Then  
      Cells(i, 3) = Rng.value  
      i = i + 1  
    End If  
  Next Rng  
  
End Sub
```

```
Sub ApplyCountifByRng()  
Dim Rng As Range  
Range("C1:C50").Copy Range("J1")  
Range("J1:J50").RemoveDuplicates 1  
  
For Each Rng In Range("J2:J9")  
    Rng.Offset(0, 1) = WorksheetFunction.CountIf(Range("C2:C50"), Rng)  
    Rng.Offset(0, 2) = WorksheetFunction.SumIf(Range("C2:C50"), Rng, Range("H2:H50"))  
Next  
End Sub
```

```
Sub ApplyVlookupByRng()  
  Sheets("Sheet15").Select  
  Dim r, Rng, rng1, crirng As Range  
  Set crirng = Range("i14:i24")  
  Set rng1 = Range("a2:g50")  
  For Each r In crirng  
    r.Offset(0, 1) = WorksheetFunction.VLookup(r, rng1, 7, 0)  
    r.Offset(0, 2) = WorksheetFunction.VLookup(r, rng1, 2, 0)  
  Next r  
End Sub
```

```
Sub SheetsActivate()  
Dim Ws As Worksheet  
For Each Ws In Worksheets  
    Ws.Activate  
    Ws.Protect "123"  
    Ws.Unprotect "123"  
    Ws.Range("m1") = "Name"  
    Ws.Range("n1") = "Designation"  
Next Ws  
End Sub
```

```
Sub CollectAllSheet1NameInOneSheet()  
Dim Ws As Worksheet  
Dim r As Integer  
r = 1  
For Each Ws In Worksheets  
    Sheets("Salary").Range("A" & r) = Ws.Name  
    r = r + 1  
Next Ws  
End Sub
```

Charts

```
Sub changecharttype()  
  Sheets("sheet19").Select  
    Dim ch As Chart  
    Range("A1").CurrentRegion.Select  
    Set ch = Charts.Add  
    ch.ChartType = xl3DAreaStacked  
End Sub
```

```
Sub createchart1()  
    Sheets("Sheet19").Select  
    Sheets("Sheet19").ChartObjects.Add 500, 30, 600, 200  
    Sheets("Sheet19").ChartObjects(1).Chart.SetSourceData Range("A1").CurrentRegion  
    ActiveSheet.ChartObjects.Delete  
End Sub
```

Array

An array is capable to store more than one value. Each value is stored in each elements of the array. Such as define array as below:-

```
Dim MyArray(3) as string  
    MyArray(0)="Ram"  
    MyArray(1)="Akash"  
    MyArray(2)="Mohan"  
    MyArray(3)="Johnson"
```

If we declare as below:-

```
Dim MyArray(1 to 3) as String  
    MyArray(1)="Ram"  
    MyArray(2)="Mohan"  
    MyArray(3)="Akash"
```

Option Base 1:- Start your array with 1 instead 0

```
Dim MyArray(3)          This array defination is just as below  
Dim MyArray(1 to 3)
```


Option Base 1

Sub Arraytest()

Dim Emp_Name(3) As String

Emp_Name(1) = "Ram"

Emp_Name(2) = "Mohan"

Emp_Name(3) = "Shyam"

Sheets("Employee").Activate

Range("K1") = Emp_Name(1)

Range("K2") = Emp_Name(2)

Range("K3") = Emp_Name(3)

End Sub

```
Sub SplitOnArray()  
Dim LStr As String  
Dim larray() As String
```

```
LStr = "Shishir2009kumar@gmail.com"
```

```
larray = split(LStr, "@")  
MsgBox larray(0)  
MsgBox larray(1)
```

```
End Sub
```

```
Sub joinOnArray()  
Dim fname As String  
Dim Ename(1 To 3) As String  
    Ename(1) = "Ram"  
    Ename(2) = "Kumar"  
    Ename(3) = "Singh"  
  
    'FName = Join(Ename)  
    fname = join(Ename, ",")  
  
    MsgBox fname  
End Sub
```

```
Sub QuickDynamicArray()  
Dim TopFilms()  
Sheets("Employee").Activate  
TopFilms = Range("A2").CurrentRegion  
Worksheets.Add  
  
'Range(ActiveCell, ActiveCell.Offset(UBound(TopFilms, 1) - 1, UBound(TopFilms, 2) - 1)) = TopFilms  
Range("A1", Range("a1").Offset(UBound(TopFilms, 1) - 1, UBound(TopFilms, 2) - 1)) = TopFilms  
  
Erase TopFilms  
End Sub
```

Files and Folders

```
Sub CreateFolderExample()
```

```
""Set Library "Microsoft Scripting Runtime" before working on Files and Folder
```

```
Dim fso As FileSystemObject
```

```
Set fso = New FileSystemObject
```

```
    fso.CreateFolder ThisWorkbook.Path & "\" & "Test"
```

```
End Sub
```

```
Sub CreateFolderWithCondition()
```

```
Dim fso As FileSystemObject
```

```
Set fso = New FileSystemObject
```

```
If Not fso.FolderExists(ThisWorkbook.Path & "\" & "Test") Then
```

```
    fso.CreateFolder ThisWorkbook.Path & "\" & "Test"
```

```
Else
```

```
    MsgBox "Folder already exist"
```

```
End If
```

```
End Sub
```

```
Sub FileInformation()  
Dim fso As FileSystemObject  
Set fso = New FileSystemObject  
Dim fpath As String  
Dim fil As Scripting.File  
  
fpath = "C:\Users\shishir kumar\Desktop\TEST\Finance.xlsx"  
  
If fso.FileExists(fpath) Then  
    Set fil = fso.GetFile(fpath)  
    Debug.Print fil.Name, fil.Size, fil.Type, fil.Path, fil.DateCreated,  
    fil.DateLastModified, fil.DateLastAccessed  
Else  
    MsgBox "File Not Exist"  
End If  
End Sub
```

```
Sub CreateFileCopy()  
Dim fso As FileSystemObject  
Set fso = New FileSystemObject  
Dim OFol, NFol As String  
  
OFol = "C:\Users\shishir kumar\Desktop\TEST\Finance.xlsx"  
NFol = "C:\Users\shishir kumar\Desktop\Fn.xlsx"  
  
If fso.FileExists(OFol) Then  
    fso.copyfile OFol, NFol  
Else  
    MsgBox "File Not Exist"  
End If  
End Sub
```

Sort

```
Sub SortDataAscDesc()  
  Sheets("Employee").Select  
  Range("A1:H50").Sort Range("A1"), xlAscending, Header:=xlYes  
  Range("A1:H50").Sort Range("A1"), xlDescending, Header:=xlYes  
  
  Range("A1:H50").Sort Key1:=Range("B1"), Key2:=Range("A1"), _  
    Order1:=xlAscending, Order2:=xlDescending, Header:=xlYes  
  
End Sub
```


Filter and Advance Filter

```
Sub FilterApply()  
    Sheets("Employee").Range("A1").AutoFilter  
End Sub
```

```
Sub FilterData()  
    Sheets("Employee").Range("A1").CurrentRegion.AutoFilter 3, "HR"  
End Sub
```

```
Sub FilterData1()  
    Sheets("Sheet6").Range("A1").CurrentRegion.AutoFilter 3, "HR", xlOr, "Sales"  
End Sub
```

```
Sub FilterData2()  
    Sheets("Sheet6").Range("A1").CurrentRegion.AutoFilter 8, ">20000", xlAnd, "<30000"  
End Sub
```

```
Sub FilterData3()  
    With Sheets("Sheet6").Range("A1").CurrentRegion  
        .AutoFilter 3, "HR"  
        .AutoFilter 8, ">30000"  
    End With  
End Sub
```

Pivot Table

```
Sub CreatePivotTableExample()
```

```
  Sheets("Employee").Select
```

```
  Dim PT As PivotTable
```

```
  Dim PC As PivotCache
```

```
  Set PC = ThisWorkbook.PivotCaches.Create(xlDatabase, Range("A1").CurrentRegion)  
  Sheets.add after:=Sheets(Sheets.count)
```

```
  Set PT = ActiveSheet.PivotTables.Add(PC, Range("B5"))
```

```
  With PT
```

```
    .PivotFields("Name").Orientation = xlRowField
```

```
    .PivotFields("Department").Orientation = xlPageField
```

```
    '.PivotFields("Department").Orientation = xlcolumnField
```

```
    .PivotFields("Total Salary").Orientation = xlDataField
```

```
  End With
```

```
End Sub
```

```
PivotTables("PivotTableName").PivotCache.Refresh
```

Function

```
Function NumIdentity(MyVal As Long) As String
    If MyVal < 0 Then
        NumIdentity = "Negative"
    ElseIf MyVal = 0 Then
        NumIdentity = "Zero"
    Else
        NumIdentity = "Positive"
    End If
End Function
```

Task

Function gradeidentity(myval As Integer) As String

If myval > 300 Then

 gradeidentity = "A"

Elseif myval > 250 Then

 gradeidentity = "B"

Elseif myval > 200 Then

 gradeidentity = "C"

Else

 gradeidentity = "Fail"

End If

```
Function OnlyText(rng As Range)
```

```
Dim i As Integer
```

```
For i = 1 To Len(rng)
```

```
    If VBA.Mid(rng, i, 1) Like "[A-Z]" Or VBA.Mid(rng, i, 1) Like "[a-z]" Then
```

```
        Onlytext = Onlytext & VBA.Mid(rng, i, 1)
```

```
    End If
```

```
Next
```

```
End Function
```

Task

```
Function OnlyNumber(rng As Range)
```

```
Dim i As Integer
```

```
For i = 1 To Len(rng)
```

```
    If VBA.Mid(rng, i, 1) Like "[0-9]" Then
```

```
        onlynumber = onlynumber & VBA.Mid(rng, i, 1)
```

```
    End If
```

```
Next
```

```
End Function
```

Function Tax(Income As Long)

Select Case Income

Case 2000000 To 5000000

Tax = (Income - 2000000) * 0.3

Tax = Tax + (2000000 - 1000000) * 0.2

Case 1000000 To 1999999

Tax = (Income - 1000000) * 0.1

Tax = Tax + (1000000 - 500000) * 0.05

Case 500000 To 1000000

Tax = (Income - 500000) * 0.05

Case Else

Tax = 0

End Select

End Function

Create function to pick month name if we pass parameter month number

Task

Connection

```
Const MyConStr As String = "Provider=Microsoft.ace.oledb.12.0;Data Source=C:\Users\shishir  
kumar\Desktop\Smruti.accdb"  
Sub ImportDataInExcelFromAccess()  
Dim ConStr As ADODB.Connection  
Dim Rs As ADODB.Recordset  
Dim RsField As ADODB.Field  
Dim rng As Range  
Dim i As Integer  
i = 1  
Set ConStr = New ADODB.Connection  
Set Rs = New ADODB.Recordset  
ConStr.ConnectionString = MyConStr  
ConStr.Open  
With Rs  
.ActiveConnection = ConStr  
.Source = "Employee"  
.LockType = adLockOptimistic  
.CursorType = adOpenForwardOnly  
.Open  
End With  
Sheets.Add after:=Sheets(Sheets.Count)  
For Each RsField In Rs.Fields  
Cells(1, i) = RsField.Name  
i = i + 1  
Next RsField  
Range("A2").CopyFromRecordset Rs  
End Sub
```



```
Sub ExportDataInMsAccess()  
  Sheets("Sheet15").Select  
  Dim ConStr As ADODB.Connection  
  Dim Rs As ADODB.Recordset  
  Dim rng As Range  
  
  Set ConStr = New ADODB.Connection  
  Set Rs = New ADODB.Recordset  
  
  ConStr.ConnectionString = MyConStr  
  ConStr.Open  
  
  With Rs  
    .ActiveConnection = ConStr  
    .Source = "Employee"  
    .LockType = adLockOptimistic  
    .CursorType = adOpenForwardOnly  
    .Open  
    .MoveLast  
  
    For Each rng In Range("A2:A50")  
      .AddNew  
      .Fields("Name") = rng.value  
      .Fields("Department") = rng.Offset(0, 1).value  
      .Fields("Salary") = rng.Offset(0, 2).value  
      .Update  
    Next rng  
  End With  
End Sub
```

```
Sub ImportDataFromSQLToExcel()
```

```
Dim cn As ADODB.Connection
```

```
Dim Rs As ADODB.Recordset
```

```
Set cn = New ADODB.Connection
```

```
Set Rs = New ADODB.Recordset
```

```
cn.ConnectionString = "Provider=sqloledb; data source=(local);initial catalog=Training;Integrated Security=SSPI;"
```

```
cn.Open
```

```
Rs.ActiveConnection = cn
```

```
Rs.Open "Emp"
```

```
Sheets("Sheet19").Select
```

```
Range("A1").CopyFromRecordset Rs
```

```
End Sub
```

```
Sub ConnectSQLToExportData()
```

Step 1 Create and Open connection

```
Dim cn As ADODB.Connection
```

```
Dim rs As ADODB.Recordset
```

```
Set cn = New ADODB.Connection
```

```
Set rs = New ADODB.Recordset
```

```
cn.ConnectionString = "Provider=sqloledb;Data Source=(loc2l);initial catalog=Training;Integrated Security=SSPI"
```

```
cn.Open
```

Step 2 Read data from Excel file

```
Worksheets("Sheet3").Select
```

```
Dim eName, eDepartment, sqlqry As String
```

```
Dim eSalary As Long
```

```
Dim Rng As Range
```

```
For Each Rng In Range("A2:A50")
```

```
    eName = Rng.Value
```

```
    eDepartment = Rng.Offset(0, 1).Value
```

```
    eSalary = Rng.Offset(0, 2).Value
```

Step 3 Create SQL query and update data in Table

```
    sqlqry = "Insert into EmployeeTest(Name,Department,Salary)Values('" & eName & "','" & eDepartment & "','" & eSalary & "')" >
```

```
    cn.Execute sqlqry
```

```
Next
```

```
MsgBox "Data exported into SQL"
```

```
End Sub
```

```
Sub sendmailitem()  
Dim olapp As Outlook.Application  
Dim olemail As Outlook.MailItem  
  
Set olapp = New Outlook.Application  
Set olemail = olapp.CreateItem(olMailItem)  
With olemail  
    .Display  
    .To = "shishir2009kumar@gmail.com"  
    .BCC = ""  
    .CC = ""  
    .Subject = "testmail"  
    .Body = "Hello World! " & vbCrLf & "Where you can enjoy in your life and live with  
your family"  
    Workbooks.Open ("C:\Users\shishir kumar\Desktop\sasfd.xlsx")  
    .Attachments.Add ActiveWorkbook.FullName  
    .Send  
End With  
End Sub
```

```
Sub createnewpptapp()  
Dim ppApp As PowerPoint.Application  
Dim ppPres As PowerPoint.Presentation  
Dim ppSlide As PowerPoint.Slide  
Sheets("Employee").Activate  
Set ppApp = New PowerPoint.Application  
Set ppPres = ppApp.Presentations.Add  
Set ppSlide = ppPres.Slides.Add(1, ppLayoutTitle)  
ppApp.Visible = True  
ppApp.Activate  
ppSlide.Shapes(1).TextFrame.TextRange = ActiveCell.value  
ppSlide.Shapes(2).TextFrame.TextRange = "My Name is Aryan"  
Set ppSlide = ppPres.Slides.Add(2, ppLayoutBlank)  
ppSlide.Select  
Range("A1").CurrentRegion.Copy  
ppSlide.Shapes.PasteSpecial ppPasteOLEObject  
ppSlide.Shapes(1).Width = ppPres.PageSetup.SlideWidth  
ppSlide.Shapes(1).Height = ppPres.PageSetup.SlideHeight  
ppSlide.Shapes(1).Width = 500  
ppSlide.Shapes(1).Height = 300  
ppSlide.Shapes(1).Left = 6  
  
Dim MyChart As Excel.ChartObject  
Set MyChart = Sheets("Employee").ChartObjects(1)  
Set ppSlide = ppPres.Slides.Add(3, ppLayoutBlank)  
MyChart.Copy  
ppSlide.Shapes.Paste  
ppSlide.Shapes(1).Width = 700  
ppSlide.Shapes(1).Height = 500  
End Sub
```

Web Scraping

Web scraping in VBA (Visual Basic for Applications) refers to the process of extracting data from websites using VBA programming within Microsoft Excel.

QnA

Question1:-What is “Option Explicit” and use of it

Answer:- Option Explicit forces to declare all variables. Which need to use in code of vba.

Question2:- What is “Option Implicit” and it’s use.

Answer:- Non declaration of variable is Option Implicit. Data type of the option implicit variables are variant.

Question3:- What is “Option Base” and it’s use.

Answer:- Option base is used to starts array declaration with 1. By default of the array starts with 0.

QnA

Question4:-What is variant data type. Explain it.

Answer:- Data type restricts us to store value in variable whereas variant data type is global data type to store any type of value in variable. Such as Number, Text, Date etc.

Question5:- What are scopes of variables.

Answer:- Scopes of variable are as below

- ☐ Procedure Level
- ☐ Module Level
- ☐ Project Level

Question6:- What types of module available in VBE.

Answer:- There are three types of modules in VBE

- ☐ Module
- ☐ Class Module
- ☐ Form

QnA

Question7:- Explain how can you pass arguments to VBA functions.

Answer:- We have two number of ways to pass arguments in function

- ByVal
- ByRef

Question8:- What is the basic object model of excel.

Answer:- Application---->Workbook---->Worksheet----->Range or Chart is object model.

Question9:- What is shortcut to open VBE.

Answer:- Alt+F11

QnA

Question10:- How to declare object variable for workbook, worksheet and range.

Answer:- Declare object variable as below:-

Dim Wb as workbook

Dim Ws as Worksheet

Dim Rng as Range

Dim Ch as Chart

Question11:- What is an array.

Answer:- An array is a group of sequence variables. An array can store multiple values.

Question12:- What is option base 1 and preserve in array.

Answer:- **Option base** defines 1 starting point of the an array as below declaration

Dim MyArray(3) as String

Preserve is used to save pre-stored value of the array and add new values in the array.

QnA

Question13:- What is Lbound and Ubound in array.

Answer:- Lbound is a function which returns lowest subscript of the given dimension of the array.

Ubound is a function which returns highest subscript of the given dimension of the array.

Question14:- What is meant by data type.

Answer:- Data type is characteristics of variable or array that determines what kind of data can be hold in variable or array.

QnA

Question15:- What are different type of errors and error handling techniques.

Answer:- There are many types of error we get in vba as below:-

- Syntax errors
- Runtime errors
- Logical errors

Question16:- How to check file exist or not on specified location.

Answer:- Dim fso As FileSystemObject
Set fso = New FileSystemObject
Dim fpath As String
fpath = "C:\Users\shishir kumar\Desktop\TEST\Finance.xlsx"
If fso.FileExists(fpath) Then
msgbox "Yes"
Else msgbox "No"
end if

Question17:- How to copy file from one location to another location.

Answer:- Dim fso As FileSystemObject

Set fso = New FileSystemObject

Dim OFol, NFol As String

OFol = "C:\Users\shishir kumar\Desktop\TEST\Finance.xlsx"

NFol = "C:\Users\shishir kumar\Desktop\Fn.xlsx"

If fso.FileExists(OFol) Then

 fso.copyfile OFol, NFol

Else

 MsgBox "File Not Exist"

End If

Question18:- What is early binding and late binding in VBA.

Answer:- Early binding is known as static binding. Early binding is a best performer.

Late binding occurs when general object associations are made, such as the Object and Variant declaration types.

Question19:- What is difference between procedure and function.

Answer:- Function returns a value and does not change value of actual argument.

The subroutine performs some set of tasks and does not return a value like functions.

Functions are called by using a variable

Subroutine can be recalled from anywhere within the program in multiple types after the declaration

QnA

Question20:- What is variable and constant and how to declare variable and constant.

Answer:- Variable are used to store some values in computer memory or storage system and those stored values are used anywhere in code. Variable value can be change while code execution.

```
Dim Counter as Integer
```

```
Counter=20
```

Constant is also used to store fix value in computer memory which is not changeable while code execution.

```
Const Counter as integer=20
```

Question21:- Difference between CurrentRegion and UsedRange properties.

Answer:- CurrentRegion reads a group of data from active cell reference whereas Usedrange reads all data from sheet.

```
Range("A1").currentregion.select
```

```
Sheets("Sheet1").usedrange.select
```


QnA

Question22:- How to debug vba code.

Answer :- We use F8 key to debug our code step by step.

Question23:- how can you stop vba code when it goes into infinite loop.

Answer :- CTRL+Break key to press multiple time and we can press esc key.

Question24:- What is object, property and method in VBA.

Question25:- Describe redim and preserve.

Answer :- Redim restructure our array declaration whereas Preserve save our previous stored values and restructure our array to store other values also.

QnA

Question26:- List out lock type in ADO.

Answer :- There are many ADO lock type in VBA as below

- ❑ **Adlockreadonly**:- Read only records
- ❑ **AdLockPessimistic**:- Pessimistic locking, record by record, The provider lock records immediately after editing.
- ❑ **AdLockOptimistic**:- Optimistic locking, record by record. The provider lock records only when calling update.
- ❑ **adLockBatchOptimistic**:- Optimistic batch updates. Require for batch update mode.

QnA

Question27:- List out cursor type in ADO.

Answer :- There are many ADO lock type in VBA as below:-

- ❑ **AdOpenForwardOnly:-** Default. Uses a forward-only cursor. Identical to a static cursor, except that you can only scroll forward through records. This improves performance when you need to make only one pass through a recordset .
- ❑ **adOpenKeyset:-** Uses a keyset cursor. Like a dynamic cursor, except that you can't see records that other users add, although records that other users delete are inaccessible from your recordset. Data changes by other users are still visible.
- ❑ **adOpenDynamic:-** Uses a dynamic cursor. Additions, changes, and deletions by other users are visible, and all types of movement through the Recordset are allowed, except for bookmarks, if the provider doesn't support them.
- ❑ **adOpenStatic:-** Uses a static cursor. A static copy of a set of records that you can use to find data or generate reports. Additions, changes, or deletions by other users are not visible.

QnA

Question28:- Types of loop in VBA.

Answer :- There are many types of loop in VBA as below:-

- ☐ For Loop
- ☐ For Each Loop
- ☐ Do Loop
- ☐ Do While Loop
- ☐ Do Until Loop

Question29:- Types of Data Type in VBA.

Answer :- There are many data types in vba as below:-

- ☐ Byte
- ☐ Integer
- ☐ Long
- ☐ Double
- ☐ Single
- ☐ String
- ☐ Date
- ☐ Variant
- ☐ Boolean
- ☐ Object

QnA

Question30:- Write five types of vba script to find last used row and last used column.

Answer :- There are number of ways to find last used data in row or column as below:-

```
Range("A1").end(xldown).offset(1,0).select
```

```
Range("A10000").end(xlup).offset(1,0).select
```

```
Dim Rcount as integer
```

```
Rcount=worksheetfunction.Counta(Range("A:A"))+1
```

```
Range("A"&Rcount).select
```

```
Rcount=worksheetfunction.Counta(Range("A:A"))+1
```

```
Range("A"&Rcount).select
```

```
Rcount=worksheetfunction.Counta(Range("A:A"))+1
```

```
Cells(Rcount,1).select
```

```
Lrow = Range("A1").CurrentRegion.Rows.Count
```

```
Range("A"&Lrow+1).select
```

```
Lrow=Range("A10000").end(xlup).row
```

```
Range("A1").offset(Lrow,1).select
```

QnA

```
Range("A1").end(xltoRight).offset(0,1).select
```

```
Range("XFD1").end(xlToLeft).offset(0,1).select
```

```
Lrow=Range("A10000").end(xlup).row+1  
Cells(Lrow,1).select
```

```
Lrow=Range("A1").SpecialCells(xlCellTypeLastCell).Row+1  
Range("A"&Lrow).select
```

QnA

Question31:- What is difference between Thisworkbook and Activeworkbook.

Answer :- Thisworkbook is workbook where code is written and activework is a workbook which is currently active.

Question32:- Write code to hide sheet so user can't unhide in excel.

Answer :- Sheets("Employee").visible=xlsheetveryhidden

Question33:- What is difference between ActiveX and Form control.

Form Control:- Form controls are the build-in feature in excel.

ActiveX Control:- ActiveX Controls sometimes need to be added manually by the user

Form Control:- Form control is much simpler.

ActiveX Control:- They have a more flexible design than Form Control

Form Control:- Form Controls don't have any properties settings

ActiveX Control:- ActiveX Control has properties settings.

QnA

Question34:- What is difference between Functions and Subroutines.

Answer :- There are some measure difference between functions and subroutines are as below:-

1. Function returns a value whereas subroutine performs a set of tasks but does not return a value.
2. Functions are called by using variable whereas subroutine can be recalled from anywhere within the program in multiple types after the declaration
3. Functions can be used as formula in sheets whereas subroutine can't be used as formula in sheets.

QnA

Question35:- Write a connection string to connect vba to SQL.

Answer :-

```
Sub ImportDataFromSQLToExcel()
```

```
Dim cn As ADODB.Connection
```

```
Dim Rs As ADODB.Recordset
```

```
Set cn = New ADODB.Connection
```

```
Set Rs = New ADODB.Recordset
```

```
cn.ConnectionString = "Provider=sqloledb;
```

```
data source=(local);
```

```
initial catalog=Training1;
```

```
Integrated Security=SSPI;"
```

```
cn.Open
```

```
Rs.ActiveConnection = cn
```

```
Rs.Open "Emp"
```

```
End Sub
```

QnA

Question36:- Use of union, join and split function in VBA

Answer :-

Question37:- Use of Dir(), Mkdir () function in VBA.

Answer :- Dir() function returns first file from given path within directory

Dim fname, fpath As String

fpath = "C:\Users\shishir kumar\Desktop\TEST\"

fname = Dir(fpath & "*.xlsx")

Mkdir() function creates a directory as per given path within directory.

Below code will create TEST folder on desktop.

Mkdir Fpath

QnA

Question38:- How to delete all files from any folder.

Answer :- VBA Kill statement deletes a file or files based on the specified path name in Excel VBA. You can specify wildcard characters either * and ? to delete multiple files.

```
Dim sPath As String
```

```
sPath = "C:\Someswari\VBAF1\VBA Functions\VBA Text
```

```
Functions\Temp.xlsm"
```

```
Kill (sPath)
```

Question39:- What is difference between do while and do until loop.

Question40:- What is web scraping and write code to open any website to fetch data.

Created by:- Shishir Kumar

Experience:- 13 Years in Mis, Software Engineer, Data Analyst, Data Science

Contact:- 9899282365/9871616527

Location:- Gurgaon

Website:-www.misanalytics.in

Training (Online and Offline):-

- Advance Excel
- VBA with Excel
- Power Point
- Ms Access
- VBA with Ms Access
- SQL Server
- SSRS
- SSIS
- Oracle
- Tableau
- Power BI
- Data Science with R
- Data Science and Development with Python

Note:- Call me without any hesitation for any help.