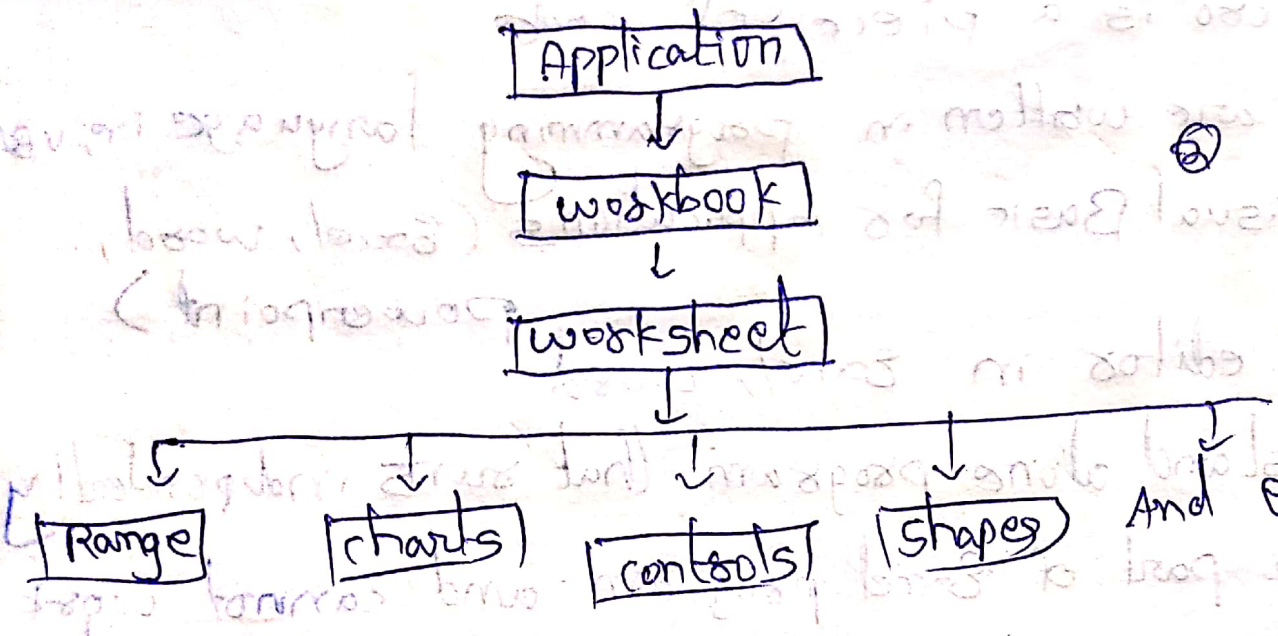
-> A macro is a piece of code
-> Macros are written in programming language i.e; VBA
-> VBA is Visual Basic for Applications (Excel, word, powerpoint)
-> VBA is editor in Excel, word
-> VB is stand alone program that runs indipendently
-> VBA is part of Excel program and cannot work -alone
-> VBS is variant of visual Basic language used for Internet Applications.
-> why Macros is required?
A) Repetitive task
Develop new formulae
complex task
-> synonyms of Macros
A) subroutine
procedure
program
-> Macros is object oriented program
-> In Excel, anything and everything is object
Ex: workbook, sheets, cells, charts etc.

object model for Excel



Record a Macro :-

=> Developer → 1) Use Relative References
2) Record Macro

=> Macro name —

Shortcut key —

Store Macro in —

Description

Save

Macro - .xlsm

without Macro - .xlsx

```
sub first_macro()
msgBox "ssh"
End sub
```

cell referencing

```
sub cell_referencing()
ActiveCell.value = "ssh"
ActiveCell.value = 40
[b5].value = 70
[c1:c10] = "ssh"
cells(8,2).value = "ssh"
Range("a1").value = "ssh"
Range("a2:a10") = "ssh"
End sub
```

copy paste

```
sub copy_paste()
Range("a1:a10") = "ssh"
```

'1st method

```
Range("b1:b10") = Range("a1:a10").value
```

'2nd method

```
Range("a1:a10").copy
Range("d1:d10").pasteSpecial
Application.cutcopymode = False
End sub
```

→ no command so paste everything

→ copied area

Font

```
sub Font()
Range("a1:a10") = "ssh"
Range("a1:a10").font.Name = "ssh"
Range("a1:a10").font.Bold = True/False
Range("a1:a10").font.size = 20
Range("a1:a10").font.Italic = True/False
Range("a1:a10").font.Underline = True/False
End sub
```

→ Arial

with Block

```
sub with_block()
Range("a1:a10") = "ssh"
with Range("a1:a10").Font
    .Name = "arial"
    .Bold = True/False
    .Italic = True/False
    .Size = 10
End with
End sub
```

(10)

Borders

```
sub borders()
Range("a1:a10").Borders.Linestyle = xlDot
Range("a1:a10").borders.color = vbGreen
Range("a1:a10").borders.weight = 3
End sub
```

xlDash
xlContinuous
xlDouble
xlNone

(11)

Alignment

```
sub alignment()
Range("a1:c5").HorizontalAlignment = xlLeft
Range("a1:c5").VerticalAlignment = xlLeft/xlRight/
                                   xlCenter
End sub
```

(2)

xlRight
xlCenter

Font color

```
sub font_color()
Range("a1:a10").font.color = vbwhite/vbBlack/
    vbyello/vbRed/vbGreen/vbBlue/
    vbCyan/vbmagneta } 8 colors
                        standard
Range("a1:a10").font.ColorIndex = 1
                                  "
                    (1 to 56)
End sub.
```

(3)

Background color

(14)

```
sub cell_background_color()
Range("a1:a10").Interior.color = vbRed
```

// 8 standard color

```
Range("a1:a10").Interior.ColorIndex = 1
```

// 1 to 56

```
End sub
```

Paste special

(15)

```
sub pastespecial()
Range("a1:a10").copy
Range("b1:b10").pastespecial xlPasteFormats
```

xlPasteColumnWidths
XlPasteValues

```
Application.cutcopyMode = False
End sub
```

orientation

```
sub orientation()
Range("a1").orientation = 0
End sub
```

10/20/30/40/50/60/70/80/90...

(16)

---

wrap Text

(17)

```
sub wrap_text()
Range("a1:a5").wrapText = True/False
End sub
```

---

merge unmerge

(18)

```
sub merge_unmerge()
Range("a1:d10").Merge
Range("a1:d1").Unmerge
End sub
```

---

clear cells

```
sub clear()
Range("a1:a10").clearFormats
Range("a1:a10").clearComments
Range("a1:a10").clearHyperlinks
Range("a1:a10").clear
End sub
```

Delete cells


```
sub delete_cells()
Range("b3").Delete
Range("a1:a10").Delete
Range("b6").EntireRow.Delete
Range("c4").EntireRow.Delete
Range("B4").EntireColumn.Delete
End sub
```

Rows & columns Insert

(21)

```
sub insert_row_column()
Range("c:c").Insert
Range("1:1").Insert
Range("b5").EntireColumn.Insert
Range("b5").EntireRow.Insert
End sub
```

Rows & columns Delete

(22)

```
sub delete_row_column()
Range("c5").EntireRow.Delete
Range("a1:a3").EntireRow.Delete
Range("c5").EntireColumn.Delete
Range("a1:c1").EntireColumn.Delete
End sub
```

column width

(23)

```
sub column_width()
Range("a1").ColumnWidth = 15
Range("a1").Columns.ColumnWidth = 15
Range("a1").Columns.AutoFit
End sub
```

Row Height

(24)

```
sub row_height()
Range("a1").RowHeight = 100
Range("a1:a5").RowHeight = 10
Range("a1:a5").Rows.RowHeight = 10
Range("a1:a5").Rows.AutoFit
End sub
```

Activate select

(85)

```
sub activate_select()
Range("a2").select
Range("b2").select
Range("a2:a5").select

Range("b3").Activate
Range("b2").Activate
Range("a2:a5").Activate
End sub
```

Columns Hide & Unhide

(86)

```
sub hide_Unhide_Columns()
Range("a:a").Columns.Hidden = True/False
Range("b:d").Columns.Hidden = True/False

End sub
```

Rows Hide & Unhide (27)

```
sub hide_unhide_rows()
Range("1:1").Rows.Hidden=True/False
Range("1:9").Rows.Hidden=True/False
End sub
```

.sheets Introduction (28)

```
sub sheet_referencing()
sheets(1).Range("a1:a10") = "ssh"
sheets("ssh").Range("a1:a10")="ssh"
End sub
```

Add sheets (29)

```
sub add_sheets()
sheets.Add
worksheets.Add
sheets.Add after:=sheets("ssh")
sheets.Add before:=sheets("ssh")
End sub
```

Add sheet with Names (30)

```
sub Add_sheets_with_Names()
sheets.Add
sheets.Add.Name="ssh"
End sub
```

Rename sheets (31)

```
sub Rename_sheets()
sheets(1).Name = "ssh"
sheets("siva").Name="ssh"
End sub
```

Get sheet Names (32)

```
Sub Get_sheet_Name()
msgBox (sheets(1).Name)
msgBox (sheets(2).Name)
End sub
```

copy sheets

(33)

```
sub copy_paste_sheet()
sheets("ssh").copy after:=sheets("siva")
sheets("ssh").copy before:=sheets("siva")
End sub
```

move sheets

(34)

```
sub move_sheets()
sheets("ssh").move after:=sheets("siva")
sheets("ssh").move before:=sheets("siva")
End sub
```

change sheet Tab color

(35)

```
sub Tab_color()
sheets("Details").Tab.color= vbBlack
sheets("Details").Tab.colorIndex=11
sheets("Details").Tab.color= False
End sub
```

Hide and Unhide sheets (36)

```
sub Hide_unhide_sheets()
sheets("ssh").visible = False/True
End sub
```

sheet protection (37)

```
sub protect_Unprotect_sheet()
sheets("ssh").protect password:=123
sheets("ssh").Unprotect password:=123
End sub
```

Activate sheet (38)

```
sub Activate_sheet()
sheets("ssh").Activate
sheets("sshsiva").select
End sub
```

create workbook

(39)

```
sub workbook_create()
workbooks.Add
workbooks.Add.saveAs Filename:="E:\siva.xlsx"
End sub.
```

Get workbook Name

(40)

```
sub Get_workbook_Name()
msgBox (Activeworkbook.Name)
msgBox (Thisworkbook.Name)
workbooks("Book1.xlsx").Activate
msgBox (Activeworkbook.Name)
msgBox (Thisworkbook.Name)
End sub
```

save & close workbook

(11)

```
sub workbook_save_close()
workbooks("Book1.xlsx").sheets(1).Range("a1:a10")
                                    = "Excel"
workbooks("Book1.xlsx").save
workbooks("Book1.xlsx").close
End sub
```

open and close workbook

(12)

```
sub workbook_open_close()
workbooks.open Filename:="E:\Demobook.xlsx"
workbooks("Demobook.xlsx").sheets(1).Range("a1:a
                                    = "Excel"
workbooks("Demobook.xlsx").save
workbooks("Demobook.xlsx").close
End sub
```

Delete workbook

(43)

```
sub Delete_workbook()
Kill ("E:\a.xlsx")
Kill ("E:\b.xlsx")
Kill ("E:\c.xlsx")
Kill ("E:\Book1.xlsx")
Kill ("E:\Demobook.xlsx")
End sub
```

create Folder

(44)

```
sub create_Folder()
MkDir ("E:\Folder1")
mkDir ("E:\Folder2")
mkDir ("E:\Folder3")
mkDir ("E:\Folder4")
end sub
```

Variable Usage


```
sub variables_usage()
Range("a1").value = "ssh"
Range("au").value= "ssh"
Range("aq").value = "ssh"
Range("as").value = "ssh"

var1 = "ssh"

Range("c1").value= var1
Range("c5").value=var1 & var1
Range("c7").value= var1
Range("c9").value= var1
End sub
```

writing multiple times

To avoid multiple lengthy words use variables.

① If any changes, we can change only one place. otherwise each time every place we have to change.

comment

```
sub variables_usage()
Dim var1
'code without variable
Range("a1").value= "ssh"
Range("a4").value= "ssh"
Range("a9").value= "ssh"
Range("a8").value= "ssh"

Rem code with variable
var1= "ssh"
Range("c1").value=var1
Range("c5").value= var1 & var1
Range("c9").value=var1
Range("c7").value=var1
End sub
```

To comment multiple lines

view - Toolbars -> customize -> commands -> edit -> comment Block / uncomment Block

Double click & drag and bring into any menubar.

For loop Exl

```
1) sub ForLoop1()
   Dim x as Integer
   For x=1 To 10
     MsgBox 25/MsgBox x
   Next ⟶ To end forloop
   End sub
```

=> If we want to learn the value of x, we can can add to watch
=> Debug → Add watch → Expression x → ok
=> x values 1 to 10

For loop Exl 2

```
2) sub For_loop1()
   Dim x as Integer
   For x=1 To 10 Step2
     MsgBox x
   Next
   End sub
```

→ Increment value by 2
Default is step1
↓
step3 ---

For loop Ex 2

```
sub For_loop2()
Dim x As integer
 For x=1 To 10
  cells(x,1).value = 10   i.e; A1 to A10
 Next
End sub
```

```
sub For_loop2()
Dim x As integer
For x=1 To 10
cells(x,1).value = x
```

```
For x=1 To 10 Step 2
cells(x,1).value= x
```

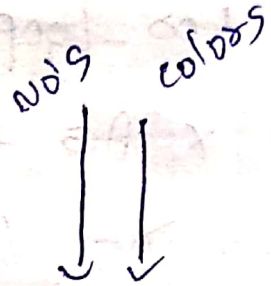
```
For x=1 To 10 step3
 cells(x,2).value = x
```

```
next
End sub
```

For loop Ex-3

(49)

```
sub For_loop3()
Dim x as Integer
  For x=1 To 56
  cells(x,1).value=x
  cells(x,1).Interior.colorIndex=x
  cells(x,2).Interior.colorIndex=x
```



For loop Ex-4

(50)

```
sub loop_Testu()
Dim x As Integer
For x=20 To 1 step -1
  cells(x,1).value = x
Next x
End sub
```

For loop Ex5

(51)

```
sub loop_Tests()
Dim x as integer
For x=1 To 10
  cells(x,x).value = x
Next x
End sub
```

For loop sheetName

(52)

```
sub sheets_Name()
For x=1 To Thisworkbook.sheets.count
msgBox Thisworkbook.sheets(x).Name
Next
End sub
```

For Each Next Loop

(53)

```
Sub For_Each_Next_1()
Dim sht as worksheet
'run on the collection of objects.
```

```
For Each sht In ThisWorkbook.Sheets
  MsgBox sht.Name
Next
End sub
```

Do while → Do until true

(54)

A

12
23
45
67
87
23
22
12

want to add 10 to each of the cell.
And I want to add only to non-Empty cells.

```
sub loop_do_while()
Dim i As Integer
  i = 1
Do while cells(i, 2).Value <> ""

cells(i, 2).value = (cells(i, 1).value + 10

i = i + 1
loop
End sub
```

Do until → Do until false (55)

Keep on execution of task until the condition is true.

```
sub loop_do_until()
Dim i As Integer
i=1
Do until i>4
cells(i,1).value=20
i=i+1
loop
end sub
```

Types of Errors (56)

Basically 3-types of Errors

1) Syntax Errors → syntax error, spelling mistakes ....

2) Compilation Errors
   ↓
   multiple lines & we miss some keywords ....

3) Runtime Errors
   ↓
   we cannot figure out Easily.

msgbox a 555555 → syntax error

⇒ How to disable dialog box → compile error Exp- - - - -

A) Tools → options → Editor → ☐ Autosyntax check → ok → dialog box

⇒ we will not get popup window but the color highlighted in red color.

⇒ keep msgbox dialog good practice.

⇒ sub abc()
For i=0 To 10
MsgBox i
End sub → no next error → compile error.

⇒ sub abc()
msgbox 6+8 but I want 6*8
End sub → runtime error → wrong results → logical mistakes.

Error Handling

sub using_resume_next() o/p:- 10

msgBox 10
msgBox 10/0 => Runtime error '11' Divide by Zero
msgBox 20

end sub

instead of stoping here, try to handling that error.

---

sub using_resume_next()

On Error Resume Next → keyword

msgBox 10
msgBox 10/0
msgBox 20

o/p:- 10
20

End sub.

If we want to notify the user that the error was in particular location.

```
sub using_resume_next()
  On Error GoTo abc  → label
    msgBox 10
    msgBox 10/0      o/p:-  10
    msgBox 20              Error message is xyz
  Done:
    Exit sub
  abc:
    msgBox "Error message is xyz"
End sub
```

Debugging

step by step execution in detail.
see & Analyze.

```
Sub abc()
Dim a, b, i  As Integer
b = 0
a = 0
For  i = 1  To 10
    a = a + i
    b = b + i
Next
End sub
```

step into (F8)

Toolbar → ☐ => local window.

Right click and Add to watch.

Break point.

locals

observe the values here

suppose I don't want to keep track on all the 3-variables & I am interest in only one variable.

we can use the method known as watch.

Toolbar → □ → watch window.

watch

b value will be track

=> At some point, if we want to keep on track among 1000 lines of code & we want track on one particular line use Breakpoint option.

i) ● → keep on left line (shotcut method)

2) Debug → Toggle Breakpoint (F9)

Immediate window

=> suppose sometimes, if we want to do some task immediatly, a very faster way, and if we are not ready to create a sub-procedure, or neither an event or a function and also you are not ready to create a userform, that time we can use immediate window. It will do your task, gives Answers, gives data which is in sheet itself.

View -> immediate window.

=> ?5>3
True
=> ?.5>50
False

} shift + ↑ + delete => To clear immediate window.

=> sheet1.Name = "ssh"
sheet2.Name = "siva"
} To change sheetName

=> To find find sheetName
? sheet1.Name
ssh

=> If we want to count how many sheets are there
? sheets.count
3

⇒ what is the value in ~~shell~~ cell No A4 -

? Range ("a4").value
April

⇒ Range ("a4").value = "ssh"
↓↓
a4 cell assign the value as ssh.

Number To Text Function (60)

↓
An readimate function
we have to create in macros.

ctrl+v = paste

seber site for code.

number	Text
200.9	= Tutorialspoint (A2) / just type No = 10, 200, ...
89.68	Two Hundred Dollars and Ninety cents
23.23	
100	
900	
3406	
100	

If statement

(61)

```
sub If_test1()
If Range("a2").value >=35 Then
    Range("c2").value = "pass"
End sub
```

If Else Statement

(62)

```
sub IF_Test2()
If Range("a2").value >=35 Then
  Range("c2").value = "pass"
Else
   Range("c2").value = "Fail"
End If
End sub
```