(1) → Range("A1").value = "ssh"

Object.property = value.

add titles to cells.

→ 'add user values to cells

Range("B1").value = Environ("UserName") ⇒ Dynamic -

---

(2)

```
sub createAndLabelNewsheet()
    'Create a new worksheet
    'object.method
    Worksheets.Add
    'add titles to cells
    'object.property = value
Range("A1").value = "created by"
Range("A2").value = "created on"
Range("A3").value = "Version"
```

```
'add user values to cells
Range("B1").value = Environ("UserName")
Range("B2").value = Date
Range("B3").value = 1
'format titles
Range("A1:A3").Font.Color = vbBlue
Range("A1:A3").Interior.Color = rgbLightCyan
End Sub
```

→ 8 colors
→ ctrl+space = intelligence
: more than 8 colors

(3)

The Types of Errors in VBA

- Syntax Errors
- Compile Errors
- Runtime Errors

Debugging a subroutine

- Stepping Through Code
- Setting Breakpoints
- The Debug Toolbar

From Macos → Tools Menu → options → To disable errors pop-up and editor format and etc....

→ Miss-spell leads to compile time errors and etc...

Rang ("A1"). value = "created by"

↓

Range

→ Debug → Compile VBA project.

---

option Explicit

Range("B2"): value = Dat

↓

Date ⇒ But it compiles.

To avoid this typing use option Explicit at top of sub.

Tools → option → editor format →

→ To become automatic filled in

☑ Require variable Declaration.

VBA editor.

Trying to refer a cell which doesn't Exist

↓

Runtime Error.

or trying to spell diff language Ex:- color → colour

Ramp ge

↓

f9 to add Break point.

---

methods for Running VBA code (4)

#Assigning keyboard shortcuts

II Drawing Simple Buttons

III Using Drawing Objects

IV Designing Ribbon Tabs and Toolbars.

I:- Alt + F8 (Macros)

Develop → Macros → options → Type letter shortcut

↓

If already there it overrides

II:- Developer → Insert → → Form controls

→ Activex controls

select Button from form control and click on sheet and select Macro to attach.

press ctrl and move button for smoothness

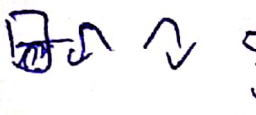
- Editing the Button:-

Right click → select your option.

III Insert → Shapes

→ Right click → Assign Macro → ok.

The Quick Access Tool Bar :-



→ Quick Access Tool bar

click → More commands.

→ For this workBook (select on right side)
Basic (For Basic VBA-ism)

→ on leftside select the Macros →
select & Add → Modify → select different icon
instead of default → ~~not~~ (~~con~~) It display
on Quick Access Tool Bar.

IV Customising Ribbon Tabs :-

Right click on any Ribbon → (Customize the Ribbon →)
→ New Tab → select & Rename → choose
Macros → & select Macro & select group in Tab
→ Add → Rename (To change icon & Name).

~~creating custom Toolbars~~ :-

selecting cells(Range, cells, ActiveCell, End, Offset) (5)

Selecting cells by Absolute position

a) selecting single cells using Range or cells
b) Referring to the Active Cell
c) selecting Multiple cells
d) Referring to the Selection
e) Using Range Names

Selecting cells Relatively

f) Finding the End of a list
g) Moving UP, Down, Left and Right
h) selecting From the Top to Bottom of a list
i) selecting Entire Regions and Entire Columns.

```
sub seledsingleCellsByPosition()
    worksheets("sheet1").Activate ---> a
    Range("A13").select
    Activecell.value = 1 ---> b
'using Row/column number
    cells(13,2).select ---> No intelligence
    Activecell.value = "the Lorax"
```

'A shorthand way to select cells
→ no intelligence

[c13].select

ActiveCell.value = #3/2/2012#
=> Days in US format
or
#2 Mar 2012#

End sub.

Selecting cells on other worksheets

worksheets("sheet1").activate

selecting cells in other workBooks.

workbooks("Book2").Activate

Changing cells without selecting Them

Sub changeCellValueswithoutSelecting()

Range("A14").value = 12

Range("B14").value = "week 7 of Ralph"

Range("C14").value = #2 nov 2012#
or
11/2/2012.

End sub.

sub changecellvaluewithoutselecting()

Workbooks("Book3").Worksheets("sheet2").Range("A14")
.Value=12

workbooks("Book3").worksheets("sheet2").Range("B14").
Value="ssh"

workbooks("Book3").worksheets("sheet2").Range("C14").
value=#11/2/2012#

End sub.

Selecting multiple cells :-

sub selectmultiplecells()
→ Activecell doesnot work in this case.
(Range("A1:C1").select
selection.interior.color = rgbDarkBlue.

End sub

Remotely changing multiple cells:-

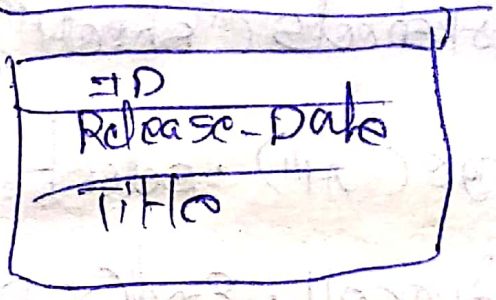
```
sub selectMultiplecells()
  Range("A1:C11").select
selection.Interior.color=rgb DarkBlue
'Remotely changing multiple cells
Range("A1:C11").Font.Color=rgbwhite

'Referring to multiple cells in shorthand
[A1:C1].Font.size = 14
'An Alternative syntax for Range
Range("A2","C2").Interior.Color=rgblightBlue
'Combining Range with cells
Range(cells(2,1),cells(2,3)).Font.Color
                              =rgb DarkBlue
End sub.
```

Creating Range Names in Excel :-

→ select Range of cells & give Name in NameBox.

→ select Range of cells → Formulae → create from selection → Find in (Name Box. (ctrl+shift+F3)



→ To Delete the Name Range, we can use the Name Manager → select → Delete.

Referring to Range Names in VBA:-

```
sub ReferToRangeNames()
Range("ID").Font.Italic=True.
[Title].font.color=rgbDarkBlue.
End sub.
```

Referring to cells Relatively:-

⇒ ctrl+↓ // Jump down to last populate cell.

Finding the end of a list

```
sub AddFilmToEndofList()
 worksheets("sheet1").Activate
 Range("A1").select
 Activecell.End(xlDown).select
 Activecell.offset(1,0).select
'Using End and offset Together.
 Range("A1").End(xlDown).offset(1,0).select
'changing cell values using Offset:-
 ActiveCell.value = Activecell.offset(-1,0).value + 1
 Activecell.offset(0,1).value = "Lincoln"
 Activecell.offset(0,2).value = #9 nov 2012#
End sub
```

11/9/2012

'selecting a list from Top To Bottom :-

ctrl + shift + ↓ : To select cells all down to end.

~~select~~
sub selectVariableCol()

Range("A3", Range("A1").End(xlDown)).select

selection.Font.Italic = True

'How Blank cells Affect the End property.

Range("B3", Range("B2").End(xlDown)).Font.
Color = rgbDarkBlue.

'selecting a Block of cells using end.

ctrl + shift + ↓

ctrl + shift + →

Range("A3", Range("A1").End(xlDown).
End(xlToRight)).select

selection.Interior.color = rgbAliceBlue

End sub.

'selecting an Entire Region of cells

=>select any where in range and ctrl+A.

'copying and pasting cells.

```
sub copyFilmList()
worksheets("sheet1").Activate
Range("A1").CurrentRegion.copy
worksheets("sheet2").Activate
Range("A1").pastespecial
End sub
Range("A1").pasteSpecial xlPasteColumnWidths
End sub.
```

'copying cells to a Destination

```
sub copyFilmList()
worksheets("sheet1").Activate
Range("A1").CurrentRegion.copy worksheets("sheet2")
                              .Range("A1")
```

```
'changing column widths.
Worksheets("sheet4").Activate
'Range("B:B") / Range("B:C") x
'Columns("B:C").width = 20   x
'Columns("B:C").AutoFit
Range("B:C").EntireColumn.AutoFit
End Sub
```

(6)

Worksheets, charts and sheets :-

⇒ Working with sheets in VBA
- Referring to and moving Between sheets
- selecting Single and multiple sheets
- Sheet Names, Code Names and Index Numbers

⇒ Manipulating sheets :-
- Inserting and Deleting sheets.
- Copying and Moving sheets
- Renaming sheets
- Hiding and Unhiding Sheets.

Activating a worksheet :-

```
sub MoveToNamedWorksheet()

 Worksheets("sheet2").Activate
End sub.
'Activating a chart sheet
sub MoveToChart()
  charts("chart1").Activate
 End sub.
```

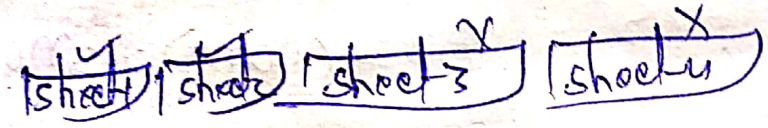
```
'Using the Generic sheets Collection
 sub MoveToAnySheet()
   sheets("sheet3").Activate
   Sheets("chart1").Activate

End sub.
```

```
'Selecting Rather than Activating
 sub MoveToNamedWorksheet()
    Worksheets("sheet2").select
End sub
```

'select is used to select more than one sheet at a time, but Activate can't.

```
sub selectMultiplesheets()
Worksheets("sheet1").select
Worksheets("sheet2").select False
'Excel will be in Group mode
End sub
```



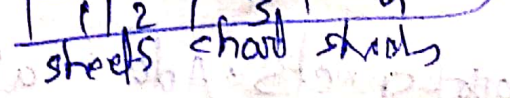
---

problems with Using sheet Names :-

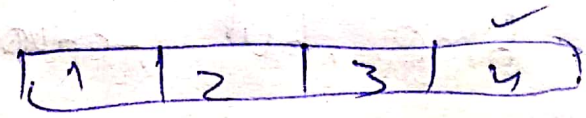
=> If some body changes sheet Name, we will get Runtime error.

```
'Using worksheet Index Numbers
sub methodsForReferringtoSheets()
Worksheets("sheet1").select
worksheets(2).select
End sub
```

=> 2 = sheet2



'Worksheets Index vs. sheets Index.



Sheets(4).select

End sub

Using sheet code Names :-

~~End sub~~

sub UsingCodeNames()

sheet2.Activate

wsMovies.Activate

End sub

code Name property

(Name) : sheet2

Name ; ssh

(Name) ; wsMovies.

Inserting worksheets :-

sub AddNewSheets()

Worksheets.Add

'choosing where to insert worksheets

Worksheets.Add Worksheets("sheet3") //left Before

worksheets.Add , worksheets("sheet3") // Right After

worksheets.Add after:=worksheets("sheet3")

End sub

'Inserting sheets at the start or End.

```
sub AddNewsheets()
worksheets.Add before:=sheets(1)        => index Number
                                        => start
worksheets.Add after:=sheets(sheets.count)  => End
End sub
```

'Inserting multiple sheets :-

```
sub AddNewsheets()
Worksheets.Add after:=sheets(sheets.count),
                                    count:=3
End sub
```

'Inserting charts :-

```
sub AddNewcharts()
'charts.Add After:=charts("Chart1")
'sheets.add type:=xlsheetType.xlchart
End sub
```

Deleting sheets :-

```
sub Deletespecificsheets()
Worksheets("sheets").Delete
End sub
```

Turning off Warning Messages

```
sub delet_specif(sheets()
Application.DisplayAlerts = False
Worksheets("sheet1").Delete
Application.DisplayAlerts = True
End sub
```

Deleting sheets Based on position :-

```
sub Deletespecifirsheets()
Application.DisplayAlerts = False
sheets(sheets.count).Delete
Application.DisplayAlerts = True
End sub
```

Deleting all sheets of one type :-

```
sub deleteAllCharts()

charts.delete (or) worksheets.Delete

End sub
```

Copying sheets :-

```
sub copysheets()
  wsMovies.copy after:=worksheets("Anything")
End sub
```

Copying sheets to a New Workbook :-

```
sub copysheets()
  wsMovies.copy
End sub
```

=> Excel creates a Brand new workbook.

Copying sheets to an Open Workbook :-

```
sub copysheets()
wsMovies.copy Before:=workbooks("Book4").
                        sheets(1)
End sub
```

Moving sheets:-

```
sub Movingsheets()
  wsMovies.move after:=sheets(sheets.count)
End sub
```

Renaming sheets:-

```
sub RenameSingleSheet()          changes Movie list to ssh
  wsMovies.Name = "ssh"
  worksheets("movie list").select
End sub
```

it wont work

Hiding and showing sheets :-

```
sub HideMoviesheet()
wsMovies.visible = xlSheetHidden (we can unhide from sheet)
                   xlSheetVisible
                   xlSheetVeryHidden (we cannot unhide from sheet)
End sub
```

we can find in property window also

Working with Workbooks

=> Using workbooks in Excel VBA

- Referring to workbooks
- Opening and creating workbooks
- Saving workbooks

```
'Referring to workbooks by Name
sub ReferringToWorkBooksByName()
Workbooks("Book2.xlsx").Activate   => case sensitive

Workbooks("Top Movies 2022.xlsm").Activate
End sub
```

```
'Referring to workbooks by Number
sub RefringToWorkbooksByIndexNumber()
workbooks(2).Activate
workbooks(1).Activate
End sub
```

'Referring to the Activeworkbook :-

```
sub UsingActiveWorkbook()
  Workbooks("Book2.xlsx").Activate
  Activeworkbook.close          => save changes
  Activeworkbook.close true
end sub
```

'Refering to ThisWorkbook :-

```
sub UsingThisWorkbook()
  workbooks("Book2.xlsx").Activate
  Thisworkbook.close => Actually closes Top movies 2012 workbook because this code is their in Top movies 2012 workbook.
End sub
```

'Opening Existing workbooks :-

```
sub openingAnExistingworkbook()

workbooks.open "C:\Users\ssh\Desktop\
                                Book2.xlsx"
        ⇓
    opens & become
    Active workbook.

End sub
```

```
'Creating New workbooks :-

sub creatingANewWorkbook()

' Workbooks.Add

workbooks.Add "Top Movies 2012.xltm"
```

=> Doesn't.

```
End sub
```

```
'Saving workbooks

sub savingAWorkbookThatHasAlreadyBeenSaved()

workbooks("Book2.xlsx").save

End sub
```

```
sub savingAnUnsavedWorkbook()
  Workbooks.Add
  ActiveWorkbook.Save => last saved location.

End sub
```

```
sub specifyingWhereToSave()
  Workbooks.Add
  ActiveWorkbook.SaveAs "C:\Test workbook.xlsx"

end sub
```

```
'changing the FileType
sub changingTheFileType()  -> This will show errors because macro enabled.
  Workbooks.Add
'ActiveWorkbook.SaveAS "C:Test workbook.xlsm"  -> F1 for information

End sub.
```

```
ActiveWorkbook.SaveAs "C:\Test workbook.xlsm", xlOpenXMLWorkbookMacroEnabled

End sub
```

Using variables in VBA

- what are Variables?
- Using Non-Declared variables
- Explicitly-Declared variables
- Variable Data Types
- Error when Using variables
- changing the scope of Variables

```
'what are variables?
sub AddaNew Film()
worksheets("sheet1").Activate

Range("B2").End(xlDown).Offset(1,0).select

Activecell.value = "Gravity"

MsgBox "Gravity was added to the list"
```

↓↓ so we need variable.

```
End sub.
```

```
' creating a Non-Declared Variable.
```

```
sub AddANewFilm()
    NewFilmName = "American Hustle"
worksheets("sheet1").Activate
Range("B2").end(xlDown).offset(1,0).select
Activecell.value = NewFilmName
MsgBox "NewFilmName" & "was added to the list"
End sub
```

'storing Inputbox Results with Variables :-

```
sub AddANewFilm()
    NewFilmName = "American Hustle"
    NewFilmName = InputBox("Type in a new FilmName")
    worksheets("sheet1").Active
Range("B2").end(xlDown).offset(1,0).select
Activecell.value = NewFilmName
```

```
MsgBox NewFilmName & "was added to the list!!"
End Sub
```

'A problem with Non-Declared Variables:-

```
sub AddANewFilm()
NewFilmName = InputBox("Type in a new film name")
Worksheets("sheet1").Activate
Range("B2").end(xlDown).offset(1,0).select
ActiveCell.value = NewFlimName
MsgBox NewFilmName & "was added to the list-!!"
End sub
```

wrong spell but no error.

View → localswindow

shows the any value of variable.

'Non-Declared vs Explicitly Declared

Option Explicit → can't use any non-declared subroutines within this module.

```
sub AddANewFilm()
'Declaring a variable

Dim NewFilmName as string

NewFilmname = InputBox ("Type in a new film name")
worksheets("sheet1").Activate
Range("B2").End(xlDown).offset(1,0).select
Activecell.value = NewFilmName
msgBox NewFilmName & "was added to the list!"
end sub
```

'Forcing Explicit Declaration :- (option Explicit)

Tools -> options -> ☑ Require Variable Declaration.

'Variable Data Types :-

```
Option Explicit
'Declaring multiple variables
Sub NewFilmName() As String
Dim NewFilmName As String
Dim FilmDate As Date
Dim FilmLength As Integer
NewFilmName = InputBox("Type in a new film name")
FilmDate = InputBox("Type in the release date")
FilmLength = InputBox("Type in the length in minutes")
Worksheets("Sheet1").Activate
Range("B2").End(xlDown).Offset(1,0).Select
ActiveCell.Value = NewFilmName
ActiveCell.Offset(0,1).Value = FilmDate //18 Nov 2013
ActiveCell.Offset(0,2).Value = FilmLength
MsgBox NewFilmName
End Sub
```

The Type mismatch Error :- date :- 31 Feb 2013

'The Overflow Error :- 32,767 - Integer.

length (Integer)

32768 => overflow

Populating and Manipulating Variables.

```
sub AddANewFilm()
Dim FilmID As Integer
Dim NewFilmName As String
Dim FilmDate As Date
Dim FilmLength As Integer

NewFilmName = InputBox("Type in a new film Name")
FilmDate = InputBox("Type in the release date")
FilmLength = InputBox("Type in the length in minutes")
Worksheets("Sheet1").Activate
```

```
Range("B2").End(xlDown).Offset(1,0).Select
FilmId = ActiveCell.Offset(-1, -1).Value
FilmId = FilmId + 1
ActiveCell.Offset(0,-1).Value = FilmId
ActiveCell.Value = NewFilmName
ActiveCell.Offset(0,1).Value = FilmDate
ActiveCell.Offset(0,2).Value = FilmLength
End Sub
```

'Variable scope

```
sub getUserInput()
  Dim FilmId As Integer
  Dim NewFilmName As String
  Dim FilmDate As Date
  Dim FilmLength As Integer
```

```
NewFilmName = InputBox ("Type in a new film name")
FilmDate = InputBox ("Type in the release date")
FilmLength = InputBox ("Type in the length in minutes")
 call AddFilmToList /or/ AddFilmToList
End sub
```

```
sub AddFilmToList()
 worksheets("sheet1").Activate
Range("B2").End(xlDown).offset(1,0).select
FilmId = Activecell.offset(-1,-1).value
FilmId = FilmId +1
Activecell.offset(0,-1).value = FilmId
Activecell.value = NewFilmName
Activecell.offset(0,1).value = FilmDate
Activecell.offset(0,2).value = FilmLength
End sub
```

---

compile error
variable not defined (o/p)

Extending the scope of Variables:-

```
option Explicit

Dim FilmId As Integer
Dim NewFilmName As String
Dim FilmDate As Date
Dim FilmLength As Integer

sub GetUserInput()
NewFilmName=InputBox("Type in a new film name")
FilmDate=InputBox("Type in the release date")
FilmLength=InputBox("Type in the length in minutes")
call AddFilmToList
End sub
```

```
sub AddFilmToList()
 Worksheets("sheet1").Activate
 Range("B2").End(xlDown).offset(1,0).select
 FilmId = Activecell.offset(-1,-1).value
 FilmId = FilmId + 1
Activecell.offset(0,-1).value = FilmId
Activecell.value = NewFilmName
Activecell.offset(0,1).value = FilmDate
Activecell.offset(0,2).value = FilmLength
End sub
```

Dim for visibility in same module.

public for visibility in Any module.

'Retaining Values in Variables :-

public/module level variables can retain entire module / across modules.

Dim public

object variables

Using object variables in VBA

- Declaring object variables
- Using the SET keyword
- Creating and Referencing objects
- Using methods to set References

Recap of Basic variables-

```
Sub RecapOfBasicVariables()
Dim myStringVariable As String
Dim myIntegerVariable As Integer
Dim myDateVariable As Date

myStringVariable = "SSH!!"
myIntegerVariable = 143
myDateVariable = Now
```

```
MsgBox "string variable contains: " & MyStringVariable
MsgBox "Integer variable contains: " & MyIntegerVariable
MsgBox "Date variable contains: " & MyDateVariable
End sub
```

```
'Declaring and setting object variables :-
Sub StoreRangeofcells()
    Dim FilmNamecells As Range   'F8 skips this one as it loads at the beginning (object)
    Set FilmNamecells = Range("B3:B15")
    FilmNamecells.Font.Color = vbBlue
End sub.
```

'Advantages of using object variables :-

```
sub storeRangeofCells()
Dim FilmNamecells As Range
set FilmNameCells = Range("B3",
                    Range("B3").End(xlDown))
FilmNamecells.Font.color = rgbBlue
FilmNameCells.Font.Italic = True
End sub.
```

~~sub~~ we can go to diff worksheet also.

```
sub storeRangeofCells()
Dim FilmNameCells As Range
set FilmNameCells = Range("B3", Range("B3").
                    End(xlDown))
Sheets2.Activate
FilmNamecells.Font.color = rgbBlue
FilmNamecells.Font.Italic = True
End sub.
```

This will work on sheet 1. but ~~we~~ active is sheet 2.

Storing new objects in Variables :-

```
option Explicit
sub ReferencingAWorksheetInVariables()
Dim MyNewsheet As Worksheet
 Worksheets.Add
 set MyNewsheet = Activesheet
 sheet1.Activate
 Range("A1").currentRegion.copy
        ↓
      ctrl+A (ctrl+↓ & ctrl+→)
MyNewsheet.Activate
Activecell.pasteSpecial
End sub
```

Creating and Referencing objects :-

```
sub Referencing AWorksheet InAVariable()
Dim myNewsheet As Worksheet

set myNewshed = Worksheets.Add

sheet.Activate
Range("A1").CurrentRegion.copy
mynewsheet.Activate
Activecell.pastespecial
end sub:
```

```
sub otherEgs()
Dim myNewBook As workbook

set myNewBook = workbooks.add

Dim myNewchart As chart

set myNewchart = charts.Add
End sub.
```

'Finding and Referencing a Ranges

```
sub FindingARange()
Dim FilmToFind As string
Dim Filmcell As Range

FilmToFind = InputBox("Type in a Film Name")
set Filmcell =
 Range("B3", Range("B3").end(xlDown)).Find
                                (FilmToFind)
' - allows to type in multiple lines
If filmcell is nothing Then
msgBox FilmToFind & "was not found" else
MsgBox Filmcell.value & "was found in"
                & Filmcell.Address
End If
End sub.
```

Message Boxes

(10)

Displaying Messages on Screen

- The MsgBox Function
- Customising Message Boxes

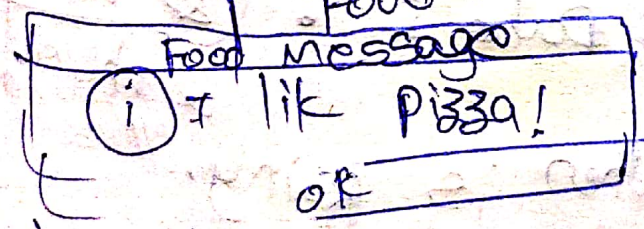
- Concatenating strings
- Using multiple lines
- Reading cell values into messages
- Asking questions with Message Boxes.

'Displaying a simple Message:-

```
Sub simpleMessage()
MsgBox "SSH"
End sub
```

'Customising a MessageBox :-

```
sub simpleMessage()
MsgBox "I like pizza!", vbInformation,
"Food message"
```




```
End Sub
```

'Using Named parameters :-

sub simpleMessage ()

→Name of parameter

msgBox prompt:= "I like pizza!",
Buttons:= vbInformation, Title:= "Food msg"

end sub.

'Concatenating strings :-

sub DateMessage()

msgBox "The date is " & date & ". The weather is Rainy."

end sub.

'Adding Extra lines to Messages :-

sub DatesMessage()

msgBox "The date is " & Date & "." & vbcrlf & vbNewline & " The weather is Rainy."

end sub.

Visual Basic → return. line carriage feed.

'Displaying values from cells :-

```
sub MovieMessage()
Range("B7").select
msgBox Activecell.value & " was released on " &
        Activecell.offset(0,1).value
end sub
```

'Asking questions with a Message Box :-

```
sub simpleMessage()
Dim Buttonclicked AS vb msgBoxResult
msgBox prompt:= "I like pizza!", Buttons:=
        vbInformation, Title:= "Food Message"

'msgBox "Do you like pizza?", vbquestion,
        "Food question"

end sub.
```

```
MsgBox "Do you like pizza?", vbQuestion + vbYesNo,
                "Food question"

Sub 'storing the Result of a MsgBox
'Dim ButtonClicked As vbMsgBoxResult

ButtonClicked = MsgBox ("Do you like Pizza?",
        vbQuestion + vbYesNo, "Food question"

If ButtonClicked = vbYes Then
  MsgBox "Yes, pizzas are great!",
                vbExclamation
else
  MsgBox "Why not? pizza's are great!",
                vbCritical
End If
end sub
```

Input Boxes

Getting User Input

- Displaying an InputBox
- Customising an InputBox
- Capturing the Result
- Cancelling from an InputBox
- Returning Different Data Types

'Displaying an Input Box :-

```
sub whatIsYourName()
InputBox "please type in your Name"
End sub
```

'Customising an InputBox :-

```
sub whatIsYourName()
InputBox "please type in your Name", "Personal Details", "enter name here...."
End sub
```

Capturing the Result of an InputBox :-

```
Sub whatIsYourName()
Range("A1").value =
  InputBox("please type in your name", "personal
                                        Details")
End sub.
```

'Using Variables with InputBoxes :-

```
sub whatIsYourName()
  Dim YourName As string
YourName=InputBox("please type in your name",
                  "personal Details")
MsgBox "Hello" & YourName
End sub.
```

⇒ Return value of InputBox is string.

views → locals window (step in F8).

'Cancelling from an InputBox :-

```
sub whatIsYourName()
  Dim YourName As string
```

```
YourName = InputBox("please type in your Name",
                    "personal Details")
if YourName = "" then
  MsgBox "you didn't enter a Name",
                    vbExclamation
else
MsgBox "Hello" & YourName
End if
End sub.
```

Input Boxes and Data Types :-

VBA has feature called Implicit Datatype conversion.

```
sub CreateAFilm()
  Dim FilmName As String
  Dim FilmDate As Date
  Dim FilmLength As Integer
```

```
FilmName = InputBox("Type in a Film Name")
FilmDate = InputBox("Type in the release date")
FilmLength = InputBox("Type in the length")
Range("B2").end(xlDown).offset(1,0).select
Activecell.value = FilmName
Activecell.offset(0,1).value = FilmDate
Activecell.offset(0,2).value = FilmLength
end sub
```

8 Nov 2018 ✓ ⇒ 08/11/2018 ✓

Problems with Different Data Types :-

⇒ If I click cancel for the FilmDate, runtime errors, Datatype mismatch.

Explicitly converting data types :-

```
sub createAFilm()
Dim FilmName As string
Dim strFilmDate As string
Dim datFilmDate As ~~integer~~ Date
Dim FilmLength As integer
FilmName = InputBox("Type in a filmName")
strFilmDate = InputBox("Type in the release date")
                ⇓
            return type is string
if strFilmDate = "" Then
 MsgBox "You didn't enter a valid date"
 Exit sub
End if
datFilmDate = cDate(strFilmDate)
FilmLength = InputBox("Type in the length")
Range("B2").End(xlDown).offset(1,0).select
ActiveCell.value = FilmName
ActiveCell.offset(0,1).value = datFilmDate
ActiveCell.offset(0,2).value = FilmLength
End sub
```