'copying from Excel into word :-

sub ---:

End with

=> Early Binding.

End with

Range("A2", Range("A2").End(xlDown).End(xlToRight)).

wdApp.Selection.Paste => many format in copy intelligence.

'saving a word Document :-

sub ---:

End with

Range("A2", Range("A2").End(xlDown).

End(xlToRight)).Copy

.Selection.Paste

.ActiveDocument.SaveAs2 Environ("Userprofile")&

"\Desktop\movie Repost.docx"

End with

End sub.

'Closing Documents and quitting word :-

```
sub---:
.ActiveDocument.SaveAs2 ----
.ActiveDocument.close
.Quit
End with
set wdApp = Nothing => Good practise
                       Though it diallocates at
End sub.                End sub.
```

```
sub---:
with wdApp
    '.visible=True //word is completly invisible.
    '.Activate
End sub
```

' Generating Unique Filenames :-

View -> Immediate window.

```
?Now
20/03/2014 11:20:44
```

: } Illigal characters.

```
? format(Now, "yyyy-mm-dd hh-mm-ss")
  2014-03-20    11-25-33
```

```
sub ----
  Dim wdApp ---
  Dim saveName As string
  ---
  ---
  .selection.paste
  saveName = Environ("userprofile") & "\Desktop\
      Movie Report " & _
    Format(Now, "yyyy-mm-dd hh-mm-ss") & _
                        ".docx"
  .ActiveDocument.saveAs2  saveName
  .ActiveDocument.close
  .quit
end with
set wdApp = nothing
end sub.
```

=> in 2007 & earlier would fail. this
=> If 2003 & earlier then doc.

'writing version - Independent codes -

view -> Object Browser -> word -> wdParagraphAlignment
-> wdAlignParagraphCenter = 1 =values.
Left = 0

F8 and (or) point it to get to know 1 & 0.

---

option explicit
sub CreateBasicWordReport()
Dim wdApp As object => late Binding.
Dim SaveName As string
Set wdApp = CreateObject("word.Application")
with wdApp
'. Visible = True
'. Activate
. Documents.Add
with . selection

.paragraphFormat.Alignment = !
                              ⇒ Underlying Values.
.paragraphFormat.Alignment = 0

End sub.

'Testing which Version is Running :-

sub--- Late Binding

with wdApp
.visible = True
'debug.print .Version ⇒ Version of word.
view → Immediate window
.Activate

End sub.

---

option Explicit

sub CreateBasicWordReport()

Dim - - - //Late Binding.

Dim - - -

Dim FileExt As string

- - -

- - -

with wdApp

.Visible = True

.Activate

.Document.Add

- - -

- - -

.selection.paste

If .Version = "11.0" Then => 2003 => If .Version <= 11 Then => Implicit conversion < 2003 and - - - (Best)

FileExt = ".doc"

Else

FileExt = ".docx"

End If

```
SaveName = Environ("Userprofile") & "\Desktop\
    Movie Report " & Format (Now ,
        "yyyy-mm-dd  hh-mm-ss") & FileExt
                                => 2007
If .Version <= 12 Then
    .ActiveDocument.SaveAS SaveName
Else
    .ActiveDocument.SaveAS2 SaveName
End If.
End sub.
```

'Creating a word Template :-

MS word :-

TOP Movies of 2012.

This week's Movie data

=insert → Links → Bookmark → Table location → Add.
| => Hidden Bookmark is here.

ctrl + enter => New page / page Break.

This weeks Movie chart

Insert → Links → Bookmarks → chartlocation ⇒ Add

⇒ Hidden Bookmark.

⇒ Save As → Browse → Word Template (*.dotx)

Movie Repost Template.dotx

VB editor in Excel :-

'creating Documents from Templates :-

Early Binding

```
sub createBasicWordReport()
 Dim wdApp As word.Application
Dim saveName As string
Dim FileExt As string
set wdApp=createobject("word.Application")
  with wdApp
        'visible =true
        'Activate
```

```
·Documents.Add "C:\Users\Andrew.Gould\_
  Documents\Custom Office Templates\_
                Movie Report Template.dotx"

'Going to a Bookmark :-
Range("A8",Range("A8").End(xlDown).
                        End(xlToRight)).Copy
'.Selection.GoTo wdGoToBookmark => For Early Binding
                                   F8 and find
'.Selection.GoTo -1,,,                value.
                                   (or) objectBrowser
.Selection.GoTo What:=-1, Name:="TableLocation"

.Selection.Paste
=> No need copy Full chart, but
ChartI.ChartArea.Copy          chart Area.
.Selection.GoTo What:=-1, Name:="ChartLocation"
.Selection.Paste
//In Realtime, many will be their so we
                will use loops
```

#f .version ---

End sub

creating and Updating linked word Tables:- 31

'Linking Excel Data in word Documents:-

- creating a New word Document
- pasting linked Excel Data
- Changing the link Source
- creating and Updating multiple linked Tables
- Changing the link Source using the save events.

'creating Some Sample Data:-

column1	column2	column3
33	37	73
97	21	32
32	71	85
21	24	71

RandBetween

Ctrl + L → To apply randbetween at a time to selected range

F9 → Refresh cell

F12 → Save As

'choosing an initial location for the file :-

F12 → Save As → Link workbook . xlsm .

'writing code to create a word Document :-

Module

option Explicit

Sub createlinkedTable()

'Referencing the word object library

Tools → References → ☑ microsoft word 15.0 object library → OK

Early Binding.

'Declaring a word Application Variable :-

Dim wdApp AS New word.Application

'Creating a New word Document :-

wdApp. Visible = True

wdApp. Documents. Add

sheet1.Range("A1:C5").copy

'creating a linked word Table :-

wdApp.selection.pasteExcelTable True, False, False

RTF = Rich Text Format
False means uses Excel RTF

'checking the Documents are linked :-

'If we made some changes in Excel, word also changes Automatically.

'Adding code to save the Document :-

wdApp.ActiveDocument.SaveAs2 "C:\Test\linked Doc.docx"

wdApp.Quit

'viewing links in the word Document :-

'If we made some changes in Excel, word also changes Automatically.

File → Info → Edit Links to Files → Dialog Box (Links) will Appear.

End sub.

```
'writing code to open the word Document :-

sub UpdateWordLinks()

  Dim wdApp As New word.Application
  Dim wdDoc As word.Document
  Dim wdField As word.Field
  set wdDoc = wdApp.Documents.open("C:\Test\
                                    Linked Doc.docx")

'Referring to a Linked Field :-
                    The Document we are opening has only
                    one item in the fields collection.
set wdField = wdDoc.Fields(1)

'changing the linked file Source :-

wdField.LinkFormat.SourceFullName = Thisworkbook.
                                    FullName

'Saving the Document and Quitting word :-

wdDoc.Save
wdApp.Quit
End sub.
```

'moving the Excel File and Updating the Links -

'Note :- If we change Excel file location, The link will work.

'multiple Links in one Document :-

'creating more Sample Data :-

column1	column2	column3	} in	sheet1, sheet2 sheet3
—	—	—		
—	—	—		
—	—	—		
—	—	—		

'copying Existing code :- (module 2)

'creating multiple linked Tables :-

```
option explicit
sub CreateMultiLinkedTable()
Dim wdApp As New word.Application
Dim ws As worksheet
wd
```

```
'wdApp.Visible = True
wdApp.Documents.Add

For Each ws In Worksheets
    ws.Range("A1:C5").Copy
    wdApp.Selection.PasteExcelTable True, False, False
    wdApp.Selection.TypeParagraph
Next ws

wdApp.ActiveDocument.SaveAs2 "C:\Test\
                              multiLinkedDoc.docx"
wdApp.Quit
End Sub
```

'Viewing the links in word :-

File -> Info -> Edit links to Files.

'Looping over Fields in a Document :-

Issue with For Each loop

↓

=infinite loop In some cases.

so, using For Next loop.

```
sub UpdateMultiWordLinks()
Dim wdApp AS New word.Application
Dim wdDoc AS word.Document
Dim i As Integer

set wdDoc= wdApp.Documents.open("C:\Test
                Multi Linked Doc.docx")

For i=1 To wdDoc.Fields.Count
  wdDoc.Fields(i).linkFormat.SourceFullName=
                ThisWorkbook.FullName
Next i
wdDoc.Save
wdApp.Quit
End sub
```

'moving the excel file and updating links:-

'Now if we change the excel file location, the links will work.

' checking the links

File -> INFO -> edit links to files.

'Using workbook events to update links:-

Thisworkbook -> Double click

General : workbook

Declarations : AfterSave -> event

' The After Save event :-

```
option Explicit
Private OriginalFileName As String
private sub workbook_AfterSave(ByVal success AS Boolean)
If originalFileName <> ThisWorkBook.FullName then
   ' UpdateMultiwordlinks
   call updateMultiwordlinks
End If
End sub.
```

" with path also.

```
'The Before Save Event
General: workbook
Declarations: BeforeSave
private sub workbook_BeforeSave (Byval SaveAsUI
                    As Boolean, Cancel As Boolean)
  OriginalFileName = Thisworkbook.FullName
End sub.
```

'Testing the save events:-
Ex: Save As (F12)

```
'Asking the User for Confirmation:-
private sub workbook_AfterSave (Byval success
                    As Boolean)
  Dim ButtonClicked As VbmsgBoxResult
  If OriginalFileName <> Thisworkbook.FullName
                    Then
    ButtonClicked = msgBox("Update links?",
                    vbquestion + VbYesNO)
```

```
If ButtonClicked = vbYes Then UpdateMultiWordLinks
End If
End Sub
```

Creating Powerpoint presentation :- (32)

'Controlling Powerpoint with Excel VBA
- Referencing the powerpoint object library
- Creating a New Instance of powerpoint
- Creating presentations and slides
- Copying Tables and charts into powerpoint
- Adding and Formatting Textboxes
- Testing with Version of PowerPoint is Running
- Making code Version-Independent

'creating Powerpoint presentations :-

module → Usingpowerpoint

option Explicit

sub CreateNewPresentation()

'Referencing powerpoint's object library :-

Tools → References → ☑ Microsoft powerpoint 15.0 object library → ok.

~~Reference~~

'Referring to PowerPoint using Variables :-

'Dim PPApp As Application

Tools → Reference → Order.

Dim PPApp As powerpoint.Application

'Creating a New Instance of powerpoint

~~ppA~~

set ppApp = New powerpoint.Application

PPApp.Visible = True → (or) MsoTrue → Microsoft Generic

ppApp.Activate

End sub.

'Using Auto-Instancing Variables :-

```
sub CreateNewPresentationAutoInstancing()
'Early Binding.          //Intelligence.
Dim ppApp As New Powerpoint.Application
ppApp.Visible = True
ppApp.Activate
End sub.
```

Downside.

// Every time it encounters ① ppApp, it checks for New Instance

② ⇒ creates new instances

```
If ppApp Is Nothing Then
  MsgBox => This never Happens.
End If
```

'Using the CreateObject Function :-

```
sub createNewPresentationwithCreateObject()
Dim ppApp As Object (No need microsoft powerpoint)
set ppApp = CreateObject("powerpoint.Application")
ppApp.Visible = True
ppApp.Activate
End sub.
```

Dim ppApp As Powerpoint.Application (Need microsoft powerpoint)

Object

Late Binding // No=Intelligence.

```
'Creating a New presentation %-
sub createNewPresentation()
Dim ppApp As powerpoint.Application.
'Dim pppres AS Powerpoint.presentation
set ppApp = New powerpoint.Application
ppApp.Visible = True
ppApp.Activate
'set pppres = ppApp.presentations.Add
ppApp.presentations.Add
End sub.
```

'creating a Title slide :-

```
sub ---
Dim
Dim
Dim ppslide As PowerPoint.slide

set
---
---
set ---
'set ppslide = ppPres.slides.Addslide(1, '')
```

⇒ not their in 2003

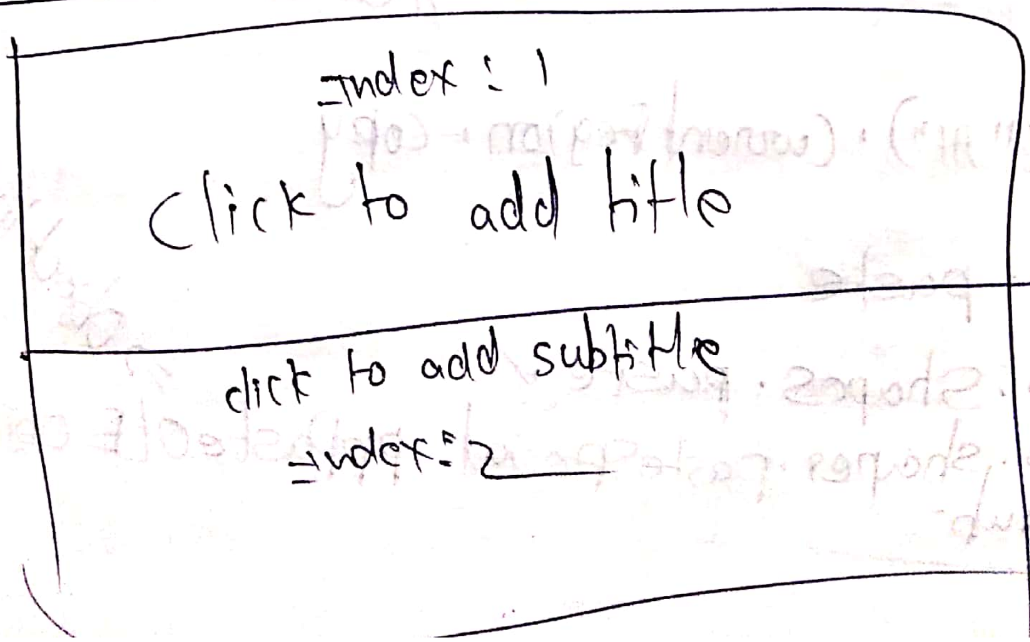
⇒ No. of slide.

we have to create custom layout

```
set ppslide = ppPres.slides.Add(1, ppLayoutTitle)

End sub.
```

'Adding Text To Textboxes :-

PowerPoint



sub --

set ppslide -- index

//Textframe
Textframe2 => 2007

ppslide.shapes(1).TextFrame.TextRange.Text = "Movie presentation"

ppslide.shapes(2).TextFrame.TextRange.Text = "By wise owl"

end sub.

'Copying an Excel Range into Powerpoint :-

sub --

ppslide.shapes(2) ---

set ppslide = ppPres.slides.Add(2, ppLayoutBlank) // 2nd slide

ppslide.select

//Excel Embed's

Range("A1").CurrentRegion.copy

~~ppslide.paste~~

'ppslide.Shapes.paste ✓

'ppslide.shapes.pasteSpecial ppPasteOLEObject => Object linking And Embedding

end sub

'Editing the size and position of shapes% -

sub - - -

ppslide.shapes.paste -> index

'ppslide.shapes(1).width = 600

ppslide.shapes(1).width = ppPres.pageSetup.slideWidth

ppslide.shapes(1).Left = 0

~~ppslide~~

End sub

sub - - -

Range ("A") . - - -

ppslide.shapes.paste.select

ppslide.shapes(1).width = ppPres.pageSetup.slideWidth

ppslide.shapes(1).Left = 0

'ppApp.ActiveWindow.Selection.ShapeRange.Align _
msoAlignMiddles, msoTrue

ppslide.shapes(1).Top = (ppPres.pageSetup.slideHeight/2) -
(ppslide.shapes(1).Height/2)

End sub.

'Adding a Custom Textbox :-

```
sub ---
Dim ---
Dim ---
Dim ---
Dim ppTextbox As powerpoint.shape
---
ppslide.shapes(1).Top= ---
set ppTextbox = ppslide.shapes.AddTextbox( _
                msoTextOrientationHorizontal,
                0,
                20,
                pppres.pagesetup.slidewidth,
                60)
'Formatting a Textbox :-
with ppTextbox.TextFrame
    .TextRange.Text="List of current Films"
    .TextRange.Paragraphformat.Alignment=ppAlignCenter
    .TextRange.Font.Size=26
    .TextRange.Font.Name="Calibri"
```

.VerticalAnchor = mso Anchor middle

End with

End sub.

'copying an Excel chart into powerpoint :-

sub ---

End with

set ppslide = PPPres.slides.Add (3, pplayoutBlank)

ppslide.select

chart1.chartArea.copy

ppslide.shapes.paste.select => To enable modify.

End sub.

'changing the chart size and position :-

PPApp.Activewindow.Selection.ShapeRange.Align
msoAlignMiddles, msoTrue

PPApp.Activewindow.Selection.ShapeRange.Align
msoAlignCenters, msoTrue

End sub.

sub - - -

ppslide.shapes.paste.select => ↑ wont work in 2016

ppslide.shapes.pastespecial ppPasteOLEObject

If ppslide.shapes(1).width > ppPres.pagesetup.slidewidth Then

ppslide.shapes(1).width = ppPres.pagesetup.slidewidth

End If

ppApp.ActiveWindow - - -

End sub.

'Applying a Template to a Presentation

sub - -

Apply Theme 2003
Apply Template => 2013
Apply Templat. => 2007

set ppPres = ppApp.Presentations.Add

'ppPres.ApplyTemplate

Templates	Themes	
2003 -> Templates -> PresentationDesigns		2003

2007 → Templates → 1033	2007 → Document Themes/2 → ✓	2007
"	"	2013

ppprocess.ApplyTemplate "c:\program files\office 2003\Templates\presentation Designs\shimmer.pot"

set ppslide = ----

End sub.

'Handling Different Template locations :-

view → immediate window

? ~~ppapp.version~~

? ppapp.version

15.0

? ppapp.path

c:\program ~~files~~\office 2013\office15

start steping F8 & check in immediate window.

'Detecting the Version of Powerpoint -

```
sub ----
Dim Templatepath As string
set pppres = ppApp.presentations.Add

'If ppApp.Version = "11.0" Then  -> Implicit conversion.
                                    2003 or Earlier
If ppApp.version <= 11 Then
  Templatepath = Left(ppApp.path, InstrRev(
                      ppApp.path, "\")) &_
    "Templates\presentation Designs)"

else
templatepath = Left (ppApp.path, InstrRev (ppApp.path,"\"))
    & "Document Themes" & Left (ppApp.version, 2) &
                                        "\"
End If
End sub
```

// 2003 & Earlier .pot

// 2007 & Above .thmx

```
sub---
Dim TemplatePath As String, TemplateExt As String
---.

If ppApp.Version <=11 Then
  TemplatePath = ----
  TemplateExt = ".pot"
Else
  TemplatePath = ----
  TemplateExt = ".thmx"
End If
---
End sub.
```

```
sub ---
---
Dim TemplateName As String
If ppApp.Version <=11 Then
---
end If
select case ppApp.Version
```

```
case "11.0"
    TemplateName = "Shimmer"
case "12.0","14.0"
    TemplateName = "Apex"
case "15.0"
    TemplateName = "Ion"
End select
On Error Resume Next => To continue without Template if error.
pppres.ApplyTemplate TemplatePath & TemplateName & "\\" & TemplateExt & "\\"
On Error GoTo 0
                stop Error Handling function
---
---
End sub.
```

'Saving and closing a presentation :-

```
sub---
'ppApp.ActiveWindow.---
'pppres.SaveAs Environ("UserProfile") &
"\Desktop\Movie pass.pptx"
End sub
```

2003 earlier ppt
2007 later pptx

```
sub ---
Dim FileExt As String
---
---
---
ppApp.ActiveWindow ---
If ppApp.Version <= 11 Then
  ---
FileExt = ".ppt"
Else
  ---
 FileExt = ".pptx"
End If
---
---
ppApp.Activewindow ---
pppres.SaveAs Environ("Userprofile") &
      "\Desktop\wise pres" & FileExt
 pppres.close
 ppApp.Quit
```

```
set ppApp = Nothing
End Sub.
```

' Converting the code to use Late Binding :-

```
option Explicit
sub CreateNewPresentation()
  Dim ppApp As object
  Dim ppPres As object
  Dim ppslide As object
'Dim ppTextBox As Shape
  Dim ppTextBox As object
Dim Templatepath As String, TemplateExt As String,
                    TemplateName As String
Dim FileExt As String

set ppApp = CreateObject("powerpoint.Application")
```

View → Object Browser → ppLayoutTitle = 1

(or)

Start stepping(f8) & see value

```
On Error GoTo 0
set ppslide = pppres.slides.Add(1,1)
set ppslide = pppres.slides.Add(2,12)
ppslide.shapes.pasteSpecial 10
With ...
.TextRange.paragraphFormat.Alignment = 2
End With
set ppslide = pppres.slides.Add(3,12)
End Sub.
```

'creating Unique File Names :-

sub - - - -
- - - -
- - - -

View → Immediate window :-

?Now
21/03/2014 13:33:24 ! } Illigal format

? format (Now, "yyyy-mm-dd hh-mm-ss")
2014-03-21 13-34-04

pp App. Activewindow - - -

pppres. Sheets environ ("User profile") &
"\Desktop\movie pres" &
format (Now, "yyyy mm dd hh mm ss") &
file ext

pppres. close
ppApp. Quit
set ppApp = Nothing
End sub.

Creating outlook Emails :-

'Controlling Outlook Using Excel VBA :-

- Referencing the outlook object Library
- Opening outlook and creating a New Email
- Setting Basic Email Properties
- setting the Format of the Email
- Adding Text to the Body of the Email
- Including a signature
- Adding Attachments
- making code Version - Independent
- Writing Complex Body Text
- Building a HTML Email
- Using the word Editor to Generate an RTF Email
- Choosing a specific signature file.

'creating outlook emails :-

Modules -> UsingOutlook

option Explicit

sub sendBasicEmail()

'Referencing the outlook Object library

Tools -> References -> ☑ Microsoft outlook 15.0 object library.

'Using a Variable to Reference Outlook :-

'Early Binding

'Dim olApp As Application => Tools -> Reference of proofing

Dim olApp As Outlook.Application

'creating a New Instance of outlook :-

set olApp = New outlook.Application

'F8 (step in) &

'windows Task manager (ctrl+shift+Esc) ->

processes -> OUTLOOK.EXE. -> After End sub

this will close Automatically.

End sub

'Using Auto-Instancing Variables :-

```
sub sendBasicEmailAutoInstancing()
Dim olApp As New Outlook.Application
```

olApp ---
⇓ Here it creates instance.

Downsides :-

① Every time it encounters olApp, it will check for instance.
⇒ This never happens, because it creates.

② If olApp IS Nothing Then
End If

End sub.

' Using the CreateObject Function :-

```
sub sendBasicEmailCreateObject()
 Dim olApp As Object //Late Binding
set olApp = CreateObject("Outlook.Application")
```
→ olApp. ⇒ No Intelligence.
```
End sub.
```

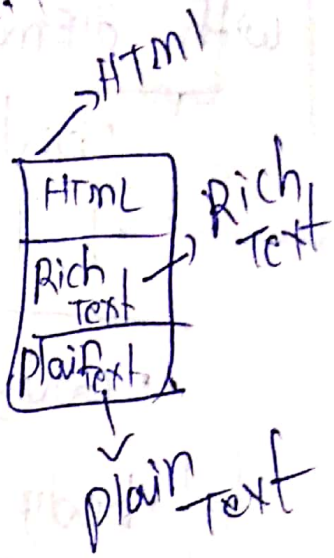
' Creating a New Email :-

```
option Explicit
sub sendBasicEmail()
    Dim olApp As outlook.Application
    Dim olApp A
    Dim olEmail AS Outlook.MailItem

set olApp = New outlook.Application
set olEmail = olApp.CreateItem(olMailItem)
'Displaying the Email
'olEmail.Display

'setting Basic Email properties :-
  with olEmail
      .Display
'     .TO = "sgk140294@gmail.com; siva.g.k140294@gmail.com"
      .TO = "sgk140294u@gmail.com"
'     .Bcc =
      .subject = "movie Repost"
```

=> or we can keep names of the mailing list

' choosing the correct Email format :-

' outlook options → mail → compose Messages in this format: => 3 formats Default: HTML format



' Format Text Tab → Aa HTML
Aa plain Text
Aa Rich Text

.BodyFormat = olFormatPlain

End with
End Sub

' If we change BodyFormat, we will loose signature.

So, move BodyFormat to Top

sub - -

with olEmail => This is in TOP! So, we don't loose Signature.
.BodyFormat = olFormatPlain => olFormatHTML olFormatRichText
.Display
.To = "sgkiu0294@gmail.com"
.subject = "movie Report"

End with
end sub

Sub---

With olEmail

.BodyFormat = olFormatPlain → we will loose signature

'.Body = "Dear Someone"

.BodyFormat = olFormatHTML

'.Body = "Dear Someone"

'.HTMLBody = "Dear Someone"

'.HTMLBody = "<H1>Dear Someone</H1>"

.BodyFormat = olFormatRichText

'.RTFBody = ~~Hello~~ "Dear Someone"

↓

Runtime error

accepts only specific formats.

End With

End Sub

Sub----

```
with olEmail
    .BodyFormat = olFormatPlain
'   .body = "Dear someone"
'Including the Email Signature :-
'   .body = "Dear someone" & vbcrlf & .body
```

vbcrlf => visual basics carriage return line feed
(return = new line, line feed = new line)

or vbnewline

The signature not yet Generated. so it Not display

So, keep .Display infront of .body.

```
End with
End sub
```

```
sub ...
with olEmail
    .BodyFormat = olFormatPlain
    .Display
    .body = "Dear someone" & vbcrlf & .body
    .To ...
End with
End sub
```

Display's signature

'Including the signature in an HTML Email :-

```
Sub ---
--- ---
With olEmail
    .BodyFormat = olFormatHTML
    .Display
    .HTMLBody = "Dear Someone" & "<br>" & .Body  X
    .HTMLBody = "Dear Someone" & "<br>" & .HTMLBody  ✓
    ---
    ---
End Sub
```

'Adding Attachments to Emails :-

```
Sub ---
With olEmail
    ---
    .HTMLBody
    .Attachments.Add Environ("Userprofile") &
        "\Desktop\movie Repost.docx"
    .To =
    .Subject =
    .Send
End With
End Sub
```

```
'making the code Version-independent &-
option Explicit
sub sendBasicEmailLateBinding()
 Dim olApp As object
 Dim olEmail As object
set olApp=createobject("outlook.Application")
set olEmail= olApp.CreateItem(0)
' F8->(step-in) & keep mouse over it to
Tools
          find value
            (or)
 views-> object Browser for values.
with olEmail
    .BodyFormat = 2
    .Display
    .HTMLBody = "Dear someone" & "<bs>" &
                                    HTMLBody
.Attachments.Add environ("userprofile") &
          "\Desktop\movie Report.docx"
```

```
    .To = "someone@somewhere.com"
    .Subject = "Movie Report"
    .Send
  End With
End Sub
```

```
'Building Complex Body Text 8-
Module -> UsingOutlookPt2
  Option Explicit
  Sub SendComplexEmail()
    Dim olApp As Outlook.Application
    Dim olEmail As Outlook.MailItem

  Set olApp = New Outlook.Application
  Set olEmail = olApp.CreateItem(olMailItem)

  With olEmail
      .BodyFormat = olFormatPlain
      .Display
```

```
.body = "Dear Someone" & vbNewLine & .body
.To = "sgkluo24y@gmail.com"
.subject = "movie Repost"
End with
End sub
```

```
Function GetMovieData() As String
  Dim FilmColumn As Range, FilmRow As Range,
      x As Range, c As Range
  Dim sts As String

  sheet1.Activate
  set FilmColumn = Range("A2", Range("A1").End(xlDown))
For Each x In FilmColumn
  set FilmRow = Range(x, x.End(xlToRight))
  For Each c In FilmRow
    sts = sts & c
    'sts = sts & c.value & vbTab
```

=> It keep Tab character at end also

so, use ~~IF~~.

```
If c.column < r.End(xlToRight).column Then
    str = str & vbTab
End If
  Next c
If  r.Row < Range("A1").End(xlDown).Row Then
   str = str & vbNewLine
End If
 Next r
 GetMovieData = str
End Function
```

```
Option Explicit
Sub SendComplexEmail()
  ---
  ---
With  olEmail
    .BodyFormat = ---
    .Display
```

```
.body = "Dear someone" & vbnewline & vbnewline &
                    GetMovieData & .body

.To = ""
End With
End Sub
```

'Building complex HTML Body Text :-

```
Module -> Usingparts
option Explicit
sub SendComplexEmailHTML()
  ---
  ---
with olEmail
    .BodyFormat = olFormatHTML
    ---
    .HTMLBody = "Dear someone<br><br>" &
                GetMovieDataHTML & .HTMLBody
    ---
    ---
 End With
End Sub
```

```
Function GetMovieDataHTML() As String
=====
Sheet1.Activate
Set FilmColumn = ----
Sts = "<table>"
For Each s In FilmColumn
  Sts = Sts & "<ts>"
  Set FilmRow = Range(s, s.End(xlToRight))
  For Each c In FilmRow
    Sts = Sts & "<td>" & c.Value & "</td>"
  Next c
Sts= Sts & "</ts>"
Next s
Sts= Sts & "</table>"
GetMovieDataHTML = Sts
End Function.
```

'Formatting in HTML Emails :-

```
sub sendComplexEmailHTML()
  ---
'.HTMLBody = "<p>Dear someone</p> ---
                    => for single Double cote's
'HTMLBody = "<p style=""color:blue;font-family:calibri"">
    Dear some</p><br><br>" &
      GetMovieDataHTML & .HTMLBody
---
---
End sub.
```

```
Function GetMovieDataHTML() As string
  ---
  ---
  ---
End sub
```

```
Function GetHeadHTML() As string
    Dim str As string
    str = "<head><style>" & vbnewline

    str= str & "p{color:blue; font-family:calibri;
                font-size:20px;}" & vbNewline

    str= str & "table{color:blue;font-family:calibri;
                font-size:20px;boder:1px solid blue;}"
                & vbnewline
```

```
sts = sts & "</style></head><body>

GetHeadHTML = sts

End Function
```

Views → Immediate window

```
? GetHeadHTML
  <head> ---- <body>
```

```
Sub SendComplexEmailHTML()
  ---
With olEmail
   .BodyFormat = ---
   .Display
   .HTMLBody = getheadhtml & "<p>Dear someone</p>
    <br><br>" & GetMovieDataHTML & .HTMLBody
                                & "</body>"
' Debug.Print .HTMLBody
   .To
  --
End Sub.
```

' Viewing the HTML of an Email :-

'Using the RTFBody property :-

```
module -> UsingOutlookProperty
option Explicit
sub sendComplexEmailRTF()
    ===
    ===
    ===
with olEmail
    .BodyFormat = olFormatRichText
    .Display
'   .RTFBody = "Dear Someone"  => Runtime Error
'   Debug.print .RTFBody  -> Array of Bytes.
                             convert into a
                             sensible string.
    Debug.print strConv(.RTFBody, vbUnicode)
    ==
End with
End sub
```

```
sub ---
    ---
Dim str() As Byte
    ---
```

'Display

sts() = stsConv("Dear someone", vbFromUnicode) → But lost signature.

'RTFBody = sts()

'Debug.print stsConv(.RTFBody, vbUnicode)

End sub.

' Using the WordEditor to Create Emails :-

Option Explicit

sub sendEmailUsingWordEditor()

Dim olApp As outlook.Application

Dim olEmail As outlook.MailItem

Dim olInsp As outlook.Inspector ⇒ like view & edit email

Gives access to wordEditor

wordEditor returns returns a reference to Microsoft word document.

Dim wdDoc As word.Document

Tools → References → Microsoft word 15.0 object library

```
set olApp = New Outlook.Application
set olEmail = olApp.CreateItem(olMailItem)

with olEmail
    .BodyFormat = olFormatRichText
    .Display
    .To = "sgk140294@gmail.com"
    .subject = "Movie report"

    set olInsp = .GetInspector
    set wdDoc = olInsp.WordEditor
End with
End sub.
```

it also works with → olFormatHTML
→ olFormatPlain

⇒ A reference to word Document (returns)

we have access to all the methods & properties from MS word.

```
wdDoc.Range.InsertBefore "Dear Someone" & vbNewLine & vbNewLine
```

sub - - -

wdDoc.Range.InsertBefore - - - -

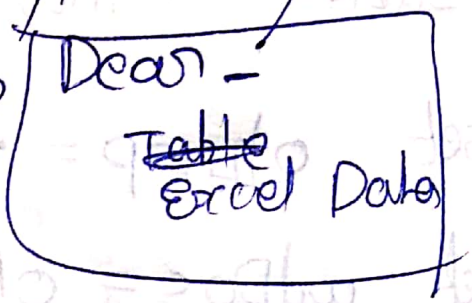
Sheet1.Activate

Range("A1").CurrentRegion.Copy

'wdDoc.Range.Paste

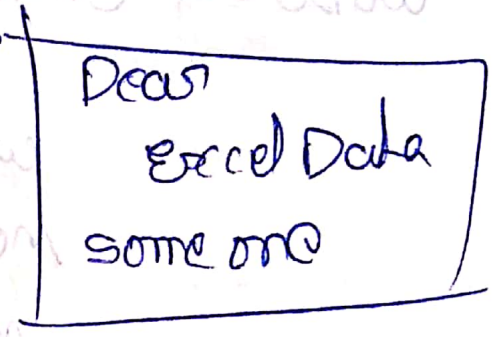
End with
End sub.

'wdDoc.Range(5).Paste → character position from starting. 5th character here



'wdDoc.Range(5,5).Paste

start ↑ 5 ↑ end



End with
End sub

---

sub - -

Dim strGreeting As String

```
strGreeting = "Dear someone" & vbNewLine
= = =
wdDoc.Range.InsertBefore strGreeting
Sheet1.Activate
Range("A1").CurrentRegion.Copy
wdDoc.Range (Len(strGreeting),
             Len(strGreeting)).Paste
End With
End Sub
```

---

choosing a specific signature :-

~~Alt~~ outlook → Message → Signature → | → personal
| → work
signatures.

→ No methods to switch signatures

→ so, ~~use~~ open up contains the signature text.