# ISE110 Notes

| Setting | Steps |
| --- | --- |
| view the name collection | Formula > Manage Names |
| use relative reference   * relative reference uses +/- to indicate the cell change | Developer > User Relative Reference |

| Range | Command |
| --- | --- |
| select the whole table | ctrl + shift + 8  ctrl + shift + down + right |
| select the column | ctrl + shift + down (和直接选择不一样) |
| select cell/s | Range("B6(:C3)").Select  Range(Ragne("B6"), Range("C3")) |
| select rows/columns | Range("sales").Rows/Columns(3/"C").Select  Range("A1", Range("A1").End(xlDown)).Select |
| clear the format | Range("sales").ClearFormats |
| select a AE11 Cell | Cells(11,31) = Range("AE11")  Range("AE10").offset(1, 0) |
| get the address | Range().Cells().Address |
| Cells() function | Range(“A2:B11”).Cells(2,1)//A3  Range(“A2:A11”).Cells(2) //A3 |
| find the cell with value | Range(“A1:D1”).Find(“Patient”) |
| count rows/columns | Range().Rows.Count |
| set the color | Range().Font.Color = vbMagenta |

| Control flow | Command |
| --- | --- |
| If else | If ... Then  [Elseif ... Then]  [Else]  End If |
| Select | Select Case ...  Case 3  Case <= 3  Case 3 to 4  Case Else  End Select |
| For Loop | For i = 1 To 10  …  Exit For  Next i |
| For Each Loop | For Each cell In Selection  …  Next cell |
| Do Until Loop | Do  …  Loop Until Cells(Row, 1).Value = “” |
| Do While Loop | Do While …  Row = Row + 1  Loop |

* Ctrl + Break to interrupt the infinite loop

| Sheets | Command |
| --- | --- |
| copy a sheet | Sheets(name).Copy Before:=Sheets(number) |
| move a sheet | Sheets(name).Move After:=Sheets(number) |
| add a sheet to the end | Sheets.Add(After:=Worksheets(Worksheets.Count)) |
| delete sheet by name | Sheets(*name*).Delete |
| copy and insert rows in a sheet | Sheets(*name*).Rows(cell.row).EntireRow.Copy  Range("A1").Offset(row, 0).Insert |

| Workbook | Command |
| --- | --- |
| open a workbook | String thisPath = ActiveWorkbook.Path  Workbooks.Open filename:=thisPath & "\" & "name.xlsx"   * function (Workbook.Open Filename := ".xlsx") is better than sub (Workbook.Open(Filename:=".xslx")) |
| save the copy as | ActiveWorkbook.SaveCopyAs thisPath & "\" & "name.xlsm" |
| close and save changes | ActiveWorkbook.Close savechanges:=True |
| close the workbook | Workbooks(day & ".xlsx").Close |
| get the file name in the dir | Dir(ActiveWorkbook.Path & "\") |
| get the next file name | filename = Dir() |

| Text File | Command |
| --- | --- |
| open a text file | String thisPath = ActiveWorkbook.Path  Workbooks.Open filename:=thisPath & "\" & "name.xlsx"   * function (Workbook.Open Filename := ".xlsx") is better than sub (Workbook.Open(Filename:=".xslx")) |
| save the copy as | ActiveWorkbook.SaveCopyAs thisPath & "\" & "name.xlsm" |
| close and save changes | ActiveWorkbook.Close savechanges:=True |

| Array | Command |
| --- | --- |
| declare an array | Dim employee(100) as String //idx = 0...99  Dim employee(1 To 100) as String //idx = 1...100 |
| re-declare an array | ReDim employee(50) //lost data  ReDim Preserve employee(20) //keep data only resize |

| Function | Scope |
| --- | --- |
| public sub *sub\_name*()  sub *sub\_name*() | *sub\_name* can be called in all modules  *sub\_name* can be called in this module only |
| private/dim *var\_name*  public *var\_name* | *var\_name* is only used in current scope(procedure/module)  *var\_name* is accessible by subroutines in outside modules |
| call *procedure1(arg1)*  *function1(arg1)* | *procedure1* is called  *function1* is used(function can return values) |
| *procedure1(*ByRef *arg1)*  *procedure1(*ByVal *arg1)* | Default. Passing the address of *arg1*  Passing the copy of *arg1*. The original value cannot be changed |

| Form | Command |
| --- | --- |
| multi-column listbox | ReDim students(length, 3) As String  For i = 1 To length  students(i, 0) = Range("A1").Offset(i, 0)  students(i, 1) = Range("B1").Offset(i, 0)  students(i, 2) = Range("C1").Offset(i, 0)  Next i  Me.listBox.List = students |
| populate the list | ReDim employee(50) //lost data  ReDim Preserve employee(20) //keep data only resize |

| Others | Command |
| --- | --- |
| first/last name | first\_name = Left(*full\_name*, InStr(*full\_name*, " ") - 1)  last\_name = Right(*full\_name*, Len(*full\_name*) - InStr(*full\_name*, " ")) |
| no alert | Application.DisplayAlerts = False |
| average/min/max | Application.WorksheetFunction.Average(Range(*start*), Range(*end*))) |
| string format | string = Format(avg, “$##.00”) |
| random number | WorksheetFunction.RandBetween(Arg1:=10, Arg2:=15) //10-15  Int((1000 + 1 - 500) \* Rnd + 500) //integer between 500 to 1000 |
| weekday function | Weekday(*date*, *[return\_value]*) returns [number](https://www.techonthenet.com/excel/formulas/weekday.php) |
| multiply matrix | mmult(*matrice 1, matrice 2*) with ctrl + shift + enter |
| line break | & char(10) & |

TSP(Lab 16,

ISE Lab

form\_name.show

double click to add the control command

in initialize() sub:

*'Empty CityListBox*

CityListBox.Clear

*'Fill CityListBox*

With CityListBox

.AddItem "San Francisco"

.AddItem "Oakland"

.AddItem "Richmond"

End With

Me.CityListBox.RowSource = “customers” //customers is the range name

answer = MsgBox("Do you really want to quit?", vbYesNo, “Terminate form”)

if answer = vbYes then Unload Me

Dim cell as range

for each cell in range(“customers”)

call me.lstcustomer.addItem(cell.value)

next cell

[Read](https://www.excel-easy.com/vba/examples/read-data-from-text-file.html) and Write Files

Open MyFile for Input as #2

with Range

do until EOF(2)

Line Input #2, dataline.offset(i, 0) = dataline

i = i + 1

loop

end with

Close #1

n = len(myString) - len(Replace(myString, “$”, “”)) // n is the number of the dollar signs

solver:

use the function(absolute reference) to calculate the goal value (to maximize)

use the solver to select the goal value and constraints

on error Goto 0

on error Resume Next

on error Goto <label\_code>

Q: Specifying a range of numbers vba