- Data science: Pandas, NumPy, SciPy, and Matplotlib
- AI: TensorFlow, PyTorch, Keras, and Scikit-learn
- Natural Language Processing (NLP) : Natural Language Toolkit (NLTK).

Conditional statements:

- If X > Y:
  Body
  elif X<Y:
  Body
  else:
  body

Loops:

Operators:

- x + = 1 → x = x + 1
- x - = 1 → x = x – 1
- X/Y return answer in float
- X//Y returns answer in int
- X%Y returns modulo
- A**B is B in power of A : A^B
- in append operation initial variable wj=hich is used for accumulation operation
- number is defined A=0
- list Is defined as A=[], A[0]=first element, A[-1]=Last element
- string is defined as A="" or A='' and ' Apple\'s
- List, Dictionary are a inbuilt class in Python
- a,b = b,a can be used to exchange the values
- return none does not returns any thing

Data Types:

- Int
- Float
- Str use " " or ' ' or "" "" or """ """
  - " and ' are equal in python but start and end quotes must be same
  - S="""STR""" used for multi line STR with enters \n inside """ """
  - Start with 0 for 1st character in string
  - String are not mutable
  - STR[NUM] to access individual character
  - STR[-NUM] is also possible
  - STR[-1] represent last character of the string
    - X='HELLO' → X[-1]='O' and X[-4]='H'
  - STR1.count("a") returns number of times "a" is in the STR1
  - STR1.index("a") returns first location of "a" appears the screen
  - \' → ' \" → " \\ → \ \n → new line
  - STR.isdigit() returns true if all elements in string are numbers from 0 to 9 , '123' , '98'

- Bool : True or False
- And, or, not
- And is evaluated 1st before Or
- () then expression then * or / then + or -
- \n new line
- \t tab
- \\ to display \ in output
- (r"AB\CD") display AB\CD as it is without considering escape charactors
- **Tuple** is an array of elements of variable data type
  T=('a',1,'ABC',1.5) then T[0]='a'
- Tuple1=Tuple2+Tuple3
- Tuple(0:3) then it created new tuple with 0,1,2 elements. Last 3 is not considered
- Same tuple can be represented by two different names
- Once tuple is created we can not change value (immutable)
- T=(1,('A','B'),2.5,(12,'AB')) Nesting of tuple T[0]=1 and T[1]=('A','B') and T[1][1]='B' and T[3][1][1]='A'
- Tuple.index(2.5) returns index of 2.5→2
- List is collection of different data class
- List has sorted(),sum(), max(),min(),len()
- C_list=sorted(C_tuple) stores shorted version of C_tuple in C_list according to alphabetical
- C_tuple.sort() sort the sequence and store in same variable C_tuple.
- ABC.reverse() returns ABC in reverse order
- B=A assign A and B both ;able to any tuple or list
- B=A[:] it creates new list or tuple and assign label B to it
- Tuple () and list [] and dictionary {}
- List are mutable so we can change data inside of list but sting are not mutable
- A[1:3]=[] removes 2,d and 3rd element so [1,2,3,4]→[1,4]
- del A[2:4] is built in command to delete some elements
- LST.extend : add new member as an elements, increase length by number added
- LST.append : add new member as a tuple, increase length by 1
- LIST.insert(1,10) insert 10 between 0th and 1st position, it will not delete any number, 1 number is shifted on 2nd position
- LIST.remove(10) removes elements with value 10
- LIST.append(X) add X element at last
- LIST.pop() removes last element X and return X
- A=LIST.pop() removes last element in the list and assign to variable A
- Index=LIST.index(X) returns the Index (location) of X in the list
- X in LIST returns True or False
- Del(A[0]) used to delete elements from the list
- A=list(STRIG1.split(",")) used to identify specifier between two elements and created a list of ['A','B','C','D'] by default SPLIT symbol is space
- "-".join(['A','B','C','D']) joints all elements of the string by – operator so it returns
- Dictionaries use {} brackets, and instead of numeric index 1,2 it has String index like A,B,C,AB
- DICT={'FIRST':1,'SECOND':2,'THIRD':3} creates dictionaries and element 1 can be excessed by DICT['FIRST']
- FIRST, SECOND are known as KEY and 1,2,3 are known as value.

- KEY are not mutable but values are
- We can add new KEY and value to dictionary D as D['KEY']=NEW VALUE
- A in DICT can be used to identify A key in DICT. Returns true or false
- DICT.keys() returns only list of KEYS of DICT
- DICT.values() returns only values of DICT
- X,Y=DICT.items() stores X=keys and Y=Values
- SET is collection like list and tuples. But difference is they are unordered , and unique elements, not reputation of same element
- ST={'AB','BC','CD','EF','GH'} defines set ST
- ST1={'AB','BC','CD','EF','GH','AB','AB'} is equal to ST as SET removes duplicate items
- SET1=set(LIST1) converts LIST1 to set data type and store in SET1
- SET1.add('IJ') add new element to SET1
- SET1.remove('IJ') delete element 'IJ' from the set SET1
- 'AB' in SET1 returns TRUE if 'AB' is in SET1 else false
- SET1 & SET2 returns common elements from both set
- SET1.union(SET2) returns the combination of all elements from both set SET1 and SET2
- SET1.issubset(SET2) returns TRUE if all elements of SET1 are in SET2
- **Loop: for**
  - For VAR1 in XYZ:
  - If XYZ is integer or float it gives an error, it must be string, list or tuple
    Here XYZ is sequence like 1:4 or ['A','B','C','D'] or "ABCD" or range(X:Y) or range(Z)
    Write inside for commands after TAB
  - x=list(range(5)) creates list of 0 to 4 and store in x
  - x=list(range(0,5)) creates list of 0 to 4 and store in x
- **enumerate** for index and element identification
  - for i, names in enumerate(myList):
    - print(i, names)
- **Loop: while**
  - While X < Y:
    - body
- **List comprehensions**
  - [n**2 for n in range(10)] = [0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
  - [planet for planet in planets if len(planet) < 6] = ['Venus', 'Earth', 'Mars']
  - [planet.upper() + '!' for planet in planets if len(planet) < 6] = ['VENUS!', 'EARTH!', 'MARS!']
  - Return the number of negative numbers in the given list = len([num for num in nums if num < 0]) = sum([num < 0 for num in nums])
- **None:**
  - Similar to NULL in python
  - None means nothing
  - do not use "==" & "!=" but use "is" & "is not" to check if something is None

Commands:

- min(1,2,3) returns minimum value
- max(1,2,3) returns maximum value
- abs(5) and abs(-5) returns 5 which is absolute value
- X=input("Enter Value") ask user to enter DATA and store in X as a sting.

- Print(A,B,C,sep='-',end='')
    - A-B-C (without enter)
- Print("STR1",NUM1,"STR2",NUM2) will print SRT1 NUM1 STR2 NUM2 on O/P
- print('sum of {} and {} is {}; product: {}.'.format(x, y, x+y, x*y)) # replace {} by variable values
- print('sum of {:.1f} and {:.2f} is {:.3f}; product: {}.'.format(x, y, x+y, x*y)) #
- Type(data) returns data type of data int, float or string
- Int(NUM) returns integer part of NUM
- Float(NUM) returns NUM as float for NUM.0
- Str(NUM) returns number as a string
- Type(True) has bool data type True=1
- Bool(1) is True and bool(0) id False
- A:B:C → From A to B with step size of C
    - A,A+c,A+2C,A+3C……B
- Len(STRING1), Len(list) returns the length of the string/List
- Sum(ARRAY) returns the sum of all the elements if an array
- + is used to concatenate two string for 'ABCD'='AB'+'CD'
- 'HI'+'HELLOW' used to concatenate the two string
- NUM*STR1 repeates STR1 string NUM of times
    - 2*'Hi ' = "HI Hi"
- String Methods
    - STR1.upper() converts to all uppercase character
    - STR1.replace("OLD","NEW") replace OLD string with NEW string
    - STR1.find("STR2") return the starting location of STR2 in STR1
    - STR1.index("STR2") return the starting location of STR2 in STR1
    - STR1.strip() remove white space in begging or end
    - STR1.split(), convert sentence in list of words, default Space is used to differentiate
    - STR1.split('separator') = STR1.split('-') = split with reference to –
    - '/'.join([month, day, year]) = 'jointer'.join(LIST) = month/dau/year
    - STR = STR1 + STR2 = joint two strings
    - "{} XYZ {} ".format(A, B) → A XYZ B → {:.2},
    - "{0} X {0} Y {1} Z {1}".format(A, B) → A X A Y B Z B
- Square(NUM) returns NUM*NUM
- Print () arguments must be string only
- If NUM=5, To display "value is 5": Print("value is"+ str(NUM))
- for _ in range(2):
  print('Hi')
  Prints Hi two times
- for I,name in enumerate(LIST_of_NAMES){
  print (name," ", i)}
  prints all names with its index start form 0.
- A in apple : in operator used to identify left string is in right string or not return true or false
- A not in apple : not in operator used to identify left string is not inside right string return true or false
- If A="HELLO" and B="HELLO" then If(A is B) returns TRUE
- A=[1,2,3,4] , B=[1,2,3,4]
- print(A is B) returns FALSE bcz id(A) and id(B) are different
- print(A==B) returns TRUE because both have same elements at each place

- b=a assign same ID code for a and b so if e change any one variable then 2nd one is also changed default
- b=a[:] assign all elements of a in b with different id so change in one variable does not reflect change in other
- round(num,X) will retuen a rounded version of num with X digits after decimal point
    - round(5.1234) → 5
    - round(5.1234,2) → 5.12
-

Don't's:

- STR[0]='A' is not possible

Basics:

- # used for comments
- All datatypes mean object
- Object has type,
- Methods means method for interaction between objects

Special libraries:

- Basics
    - Import math
    - from math import log, pi
    - from math import * → Import all variables
- Turtle
    - MARKER is an instance of turtle class

```
import turtle # import turtle library
Wn=turtle.Screen() # open screen
Wn.bgcolor('lightgreen') # change screen color to light green
MARKER=turtle.Turtle() # assign MARKER as object for turtle class
MARKER.color('blue') # change turtle color to blue
MARKER.pensixe(10)
MARKER.shape('turtle') # arrow, blank, circle, classic, square, triangle, turtle
MARKER.up() # lift the pen so it wont draw line
MARKER.down() # down the pen so it can draw line
MARKER.spped(10)#to control the speed
for _ in range(4)
    MARKER.stamp() # print shape of object
    MARKER.forward(150) # move MARKER object forward by 150 pixel
    MARKER.left(90) # turn MARKER object in left (ACW) direction by 90 degree
Wn.exitonclick()
```

- Random
    - Random.randam() returns random number
    - Random.randrange(1,7) returns 1 to 6 int number
    - From random import randrange
        - Then random.randrange can be used as randrange
- Image
    - import image
    - p = image.Pixel(45, 76, 200) # assign RGB 45,76,200 respectively for pixel p

- print(p.getRed())# p.getRed() returns red component of p
- p.setRed(66)#p.setRed is used to assign new value to red component

- Image
  - import image
  - img = image.Image("luther.jpg") # read image and store in img
  - print(img.getWidth()) # img.getWidth() returns number of column
  - print(img.getHeight()) # img.getHight() returns number of rows
  - p = img.getPixel(45, 55) #omg.getpixel(c,r) values read [r,g,b] values and #store in p
- Image
  - import image #import image library
  - img = image.Image("luther.jpg") #read an image
  - win = image.ImageWin(img.getWidth(), img.getHeight()) #open a window to draw an image
  - img.draw(win) # draw or display image
  - img.setDelay(1,15) # setDelay(0) turns off animation #set delay of process
  - for row in range(img.getHeight()): #row loop
  - for col in range(img.getWidth()): #column loop
  - p = img.getPixel(col, row) # read pixel of img
  - newred = 255 - p.getRed() # change red color
  - newgreen = 255 - p.getGreen() #green
  - newblue = 255 - p.getBlue() #blue
  - newpixel = image.Pixel(newred, newgreen, newblue) #assign new values to newpixel
  - img.setPixel(col, row, newpixel) #change the color of displayed image
  - img.draw(win) # show image
  - win.exitonclick() # close image window when we click on it
- **function**
  - def FUNCTION_NAME(ARGUMENTS=”Default_Values”) :
    “””Help Instructions”””
    DO something
    Global VARIABLE1
    Return RET
  - without return also function returns None
  - Python must have one line as body in fuction, if we don’t have we can use pass
    - Def NO_OP()
      - Pass
  - If FUNCTION_NAME() is used the default ARGUMENTS is ”Default_Values”
  - If FUNCTION_NAME(“ABC”) is used the ARGUMENTS is ”ABC”
  - X,Y=function(Arguments) store two returns values in X and Y
  - Do something=pass used do nothing
  - VARIABLE1 type is global and use value which is defined outside function

- *ARRAY returns number of elements in array used as function arguments, it is equivalent to len(array), instead of array, tuple, list can be used
- Variable defines outside functions are known as global variable
- Variable defines inside functions are known as local variable
- Scope of variable: If global variable x is used inside function, its value can be changed inside function but after execution of function global variable remains same as old value not assigned inside function.
- Global X defines the scope of the variable X as global variable

- Class
  - **Class** CLASS_NAME**(Object):**
    Def _init_(Self, Radious=1,color='black') ## constructor to initiable variable
    Self.radious=radious;
    Self.color=color ;
  - Class contain data/attributes/values and methods/functions/process
  - Constructor is there in class only not in function
  - Radious=1 and color='black' defines default values of an arguments
  - List1.sort() is method
  - List1 is an object/instance of type list (type(List1)→ List)
  - Class Circle(object): # How to define class
    - Def _init_ (self, radius=1, colour='black')
      - Self.radius=radius
      - Self.colour=colour
    - Def add_radius(self,r) # method to change attributes
      - Self.radious= Self.radious+r
      - Return(self.radius)
  - C1=Circle (5,'pink) # used to create object C1 of class Circle
  - C1.radius() is used to address radius value of object C1 of type Circle class
  - C1.radius()=10 can be used to modify value of radius of object C1
  - C1.add_radius(3) # will add 3 in current radius value
  - Dir(C1) list the methods and attributes of object C1
- Files
  - FILE1=open("M:\Python\Coursera\PYforAIandDS\Files\Marks.csv")
    FILE1=open("M:\Python\Coursera\PYforAIandDS\Files\Marks.csv",'r')
    open open a file in reading mode and assign object FILE1
  - FILE2=open("M:\Python\Coursera\PYforAIandDS\Files\Marks.csv","w") open a file in Writing mode and assign object FILE2
  - FILE3=open("M:\Python\Coursera\PYforAIandDS\Files\Marks.csv","a") open a file in appending mode and assign object FILE3
  - FILE1.name returned name of file
  - FILE1.mode returns weather it is in reading/writing/append mode
  - FILE1.close() close file
  - To close file automatically
    - **With** open("NAME.txt","r") as FILE1: #open a file and label it FILE1

        - FILE_STUFF=FILE1.read() #read file and store in FILE_STUFF
        - Print(FILE_STUFF) #print file data
    - How to read one or multiple lines from the file
        - FILE_LINES=FILE1.readlines() # Read data in file in terms of lines so user can access individual line data by FILE_LINES[0] for 1st line and FILE_LINES[1] for 2nd line as on
        - FILE_LINES=FILE1.readlin**e**() # Read 1st line only during 1st call
        - FILE_LINES=FILE1.readlin**e**() # Read 2nd line only when called 2nd time
        - FILE_LINES=FILE1.readlin**e**() # Read 3rd line only when call 3rd time
        - FILE_LINES=FILE1.readlin**es**(x) # Read only first x character in 1st line
        - FILE_LINES=FILE1.readlin**es**(y) # Read only first x character in 2nd line
        - Print(FILE1.closed) #Return wather file is closed (TRUE) or not (FALSE)
        - Print(FILE_STUFF) #print file data even we have closed file bcz data is stored in
    - How to Write and copy files
        - FILE1.write("TEXT") # add one line TEXT in file which is handled by object FILE1 in write mode
        - FILE1.write("TEXT") # keep existing details and add one line TEXT in file which is handled by object FILE1 in append mode
        - Copy one file in 2nd
            - with open ("path\Marks3.txt","r") as FILE1:
            - with open ("path\Marks.txt","w") as FILE2:
            - for line in FILE1:
            - FILE2.write(line)
    - Exception handling
        - Try: #try to execute this lines
            - Main_File=open("Temp.txt",'r')
            - Main_File.Write("Hello")
        - Except IOError: # if IOError occur the print this part
            - Print('Unable to open the file')
        - Else: #If exception doesnot occur then print this part
            - print('the file is written successfully')
        - finally: # weather exception occur or not execute this part (always executed)
            - Main_File.close()
            - Print('file is now closed')
    - Additional modes
        - r+ : Reading and writing. Cannot truncate the file.
        - w+ : Writing and reading. Truncates the file.
        - a+ : Appending and Reading. Creates a new file, if none exists
        -
- PANDAS
    - Import pandas # import data analytics library

- Import pandas as PD # give all properties of panda to PD object
- HANDLE1=”FILENAME.csv” # assign one handle to file and stores path of file in handle HANDLE1
- F_DATA=PD.read_csv(HANDLE1) # read csv file FILENAME.csv and store data in F_DATA
- F_HD=F_DATA.head() # Load top 6 rows in F_HD to visualize what is in the file
- F_DATA=PD.read_excel(HANDLE1) # read EXCEL file
- In case of dictionary structure SONGS the command S_FRANE=PD.DataFranme(SONGS) load data frame of dictionary SONGS, Key assigned as Lable of each column and values of key assigned as individual row
- songs={"Album":['A','B','C','D','E'],"Released":[1,2,3,4,5],"Length":['1.2','2.3','3.4','4.5','5.6']} # Created dictionary with label and values
- songDF=PD.DataFrame(songs) # Assign songDF handle to data frames created by dictionary
- A=songDF['Released'].unique() # stores all unique element available in Released label values
- A.to_csv("NEW.csv")# Store file NEW.csv with data frames A
- DF1=songDF[songDF[“Realeased”]>5] returns all the rows which has Realeased value greather than 5
- songALBUM=songDF[["Album"]] #load only data frame of columns which las label Album
- songTWO=songDF[["Album",”Released”]] #load data frame of two columns which las label Album and Released
- songDF.**iloc**[0,0] and songDF.**ix**[0,0]# retuen the oth row and 0^{th} column element of data frame songDF
- songDF.**loc**[0,’Album’] and songDF.**ix**[0,’Album’]# retuen the 0th row element of Album labe column in data frame songDF
- songDF.ix[**1:2,0:2**] # returns sub matrix with defined range of data frame songDF here ix can be replaced by iloc
- songDF.**ix**[1:2,**”Album”:”Length”**] # returns sub matrix with defined range of rows and comumn from Album to Length in data frame songDF, here ix can be replaced by loc
- XML: extensible markup language van be used with pandas and xml library to read

- NUMPY (1D)
  - Library for scientific computing with less memory and fast
  - Import numpy as NP
  - from numpy import *→ then instead of NP.array we can directly use array
  - A=NP.array([0,1,2,3,4]) # Creates an array of integer number
  - B=NP.array([0.5,1.4,2.5,3.2,4.9]) # Creates an array of float number
  - In NUMPY all elements must of same type like integer, not mixed data type like list
  - print(type(A)) # print type of an array which is numpy.ndarray, it will not πreturn int or float
  - print(A.dtype) # Print data type which is int32, for B float64

- print(A.size) # print number of elements which is 5
- print(A.ndim) # print number of dimension which is 1
- print(A.shape) # print number of rows and column which is 1,5
- u=[1,0] and v=[0,1] then z=u+v add two vectors element by element and answer is [1,1]
- if z=[]:
  for n,m in zip(u,v):
    - z.append(n+m)

  here n is element in u and m is element in v, zip assign two variables n & M to given two vectors u & v, z. append add n+m element in z array
  of we perform z=u+v directly it returns [1, 2, 3, 2, 3, 4]=[u,v]
- With numpy library z=A+B or A-B or 2*A or A*B or A+2 is possible if A and B are numpy
- => C=NP.dot(A,B) #A^T*B # perform DOT product, Multiply elements then add them
- C=A.mean() or A.max() find mean and maximum value of an numpy array A
- NP.pi is π in mumpy library
- C=NP.sin(A) find the sine values of all elements
- C=NP.linspace(-5,5,num=11) # returns an array with 11 equally spaced elements starting from -5 and end with 5
- a=[[1,2,3],[4,5,6],[7,8,9]] # Creates an multi dimensional array
- A=np.array(a) # creates an array to Matrix
- print(A[1][2]) or print(A[1,2]) # access 2nd row and 3rd column value
- NP.arrange(start,end,stepsize) will create an arrany from Start to End numbers with mentioned stepsize

- Numpy(2D)
  - a=[[1,2],[3,4],[5,6]] is 2D List
  - A=NP.array(a)
  - A=
  - 1 2
  - 3 4
  - 5 6
  - A.ndim → 2
  - A.shape → 3X2
  - A.size→ 6
  - A.dtype → int8 /int16/float16 retuned data type
  - A.itemsize → returns size of bytes of each element
  - M=NP.dot(A,B) performs actual matrix multiplication between A and B
  - +,-,/,* performs element by element +,-,/,*
  - NP.zeros((r,c),dtypes=NP.int16) will create int type rXc matrix with all 0 elements
  - NP.ones((r,c),dtypes=NP.int8) will create int type rXc matrix with all 1 elements
  - NP.full((r,c),V) will create a rXc matrix with all ments as V

    - NP.random.rand(r,c) will create a rXc matrix with random numbers between 0 and 1
    - NP.reshape(ARRAY,(new_r,new_c)) will change given ARRAY to new_r X new_C domention array, if new_r or new_c = -1 then new ARRAY has new_c column or new_r rows

- MATPLOTLIB.PYPLOT library
    - Used to plot
    - import matplotlib.pyplot as PT
    - A=NP.linspace(-NP.pi,NP.pi,num=100)
    - B=NP.sin(A) # Canculate sin value for each element of A
    - PT.plot(A,B) # Plot sin
- Json
    - Import json
    - With open ("Filename.json",'r') as FileData:
        - Json_Obj=json.load(FileData)

API

- API : Application Program Interface
- User Program and any software can communicate with help of API
- In REST API
    - We use HTTP method to transfer data which contain JSON file
    - User program(client) and Server (Resource / Internet) can communicate with help of REST API
    - REST API: Representational State Transfer API provide advantage of cloud, processing, ML and AI algorithms, Storage
- API Key : is used to access API, with it server identify the client and authorise the client
- URL : Uniform resource locator (URL)
- HTTP : Hypertext Transfer Protocol : Protocol to transfer data between browser and servers
- JSON : JavaScript Object Notation
- Internet address = Scheme + Base address + Route
    - http:// + www.ibm.com + images/Birds.png
- HTTP file data transfer
    - Request method
        - GET : Retrieve data from server
        - POST: Submit data to server
        - PUT : Update data already on server
        - DELETE: Delete data from server
    - Status code in response:
        - 100 OK so far
        - 200 OK
        - 300 Multiple choices

        - 401 Unauthorised
        - 403 Forbidden
        - 404 Not found
        - 501 Not implemented
    - Import requests
        - Requests is a python Library that allows you to end HTTP/1.1 requests
        - R=Request.get(url) used to GET data from the URL
        - R.status_code : used to identify status like 100,200,300,401
        - R.request.header : used to see GET method header
        - R.request.body : used to GET method body, which is always none in GET
        - R.headers : used to get data of headers in received URL
        - R.encoding : UTF-8
        - R.text[0:100] display first 100 characters in body