# What is Python?

python is general purpose, high level, interpreted language with easy syntax and dynamically initialised. the development of python started out as a hobby for its creator GUIDO VAN ROSSUM in 1991. because he wanted the language which is beautiful to look and easy to learn. So, he used indentation inplace of curly braces to describe the piece of code. it become popular when the machine learning and D.S and A.I cames into the picture.

print ("hello world")  
print ("hello again") ## shift+enter to excute  
a= 5  
b= 10  
print (a+b)  
print (5/2)  
  
if:

hello world  
hello again  
15  
2.5

# Difference between python 2 and python 3

print "hello world" ## print ("hello world")

5/ 2 = 2 ## 5/2 = 2.5

raw\_input ## input

# 1 Scripting   
# 2- Programming   
# 3 - GUI  
# 4 - A.I  
# 5 - Data science  
# 6- Machine learning

import random

random.randint (1,100)

69

import time

time.sleep (5)  
print ("hello world")

hello world

# Why python is Popular ?

1- Easy

2- Free (open-source)

3- application

4- libraries

# Feature

1-Simplicity

2- Open source

3- Portability

4- Embedded

5- Interpreter

6- Huge libraries

# Companies uses python

1- Google

2- Netflix

3- Bit torrent

4- NASA

# Carrer opportunities

1- Game development

2- WEb development

3- AI/Data science

4- big data

5- Smart device

6- Automation task

# Python basics

1 - Varible,datatype, Operators

2- Arrays

3- Flow control

4- Methods

5- File handling

6- Exception handling

## Variables

#### 1. Variables are containers for storing data values.

#### 2. Unlike other programming languages, Python has no command for declaring a variable.

#### 3. A variable is created the moment you first assign a value to it.

#### 4. ( \_ ) , ( a-z ) , (A-Z)

#### 5. variable name cannot be start with numbers.

#### 6. Special character ( @!#$%^&\*)

a= "world"  
print(a)

world

a1bcd = 10  
print (a1bcd)

10

uttkarsh = "delhi"  
print(uttkarsh)

delhi

C = "class"  
print (C)

class

Xyz = 100  
print (Xyz)

100

## Comments

### Single line comment  
  
a = "hello" ### this is to greet   
print(a)

hello

### Multiline Comment  
  
print("hello ")  
  
"""this is   
multi line comment  
here it is to show."""  
  
'''this is also   
multi line   
comment'''  
  
print("Ankit")

hello   
Ankit

## Keyword ( Reserved\_words )

import keyword

print(keyword.kwlist)

['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']

import = a  
print (a)

File "<ipython-input-19-58ec20b7c9fa>", line 1  
 import = a  
 ^  
SyntaxError: invalid syntax

# 1. Integer - 2,5,4,-3,-6,0,9  
# 2. Float - 2.545,3.14,-8.889  
# 3. String - "hello",'hii',"98875"  
# 4. boolean - True, False  
# 5. List - [4,6,4,3.143,"hello"]  
# 6. Tuple - (4,6,4,3.143,"hello")  
# 7. Set - {"hello",4, 3}  
# 8. Dictionary- {2:"two",5:"hello"}

a = 3  
b = 3.143  
c = "hello"

print(a)  
print(type(a))

3  
<class 'int'>

# String

String can be sliced

string is combination of character

s = 'this is something important string.'  
print(s[0])  
print(s[0:6:2])  
print(len(s))

t  
ti   
35

### 3. List

###### a. list is an ordered sequence.

###### b. we can store duplicate values.

###### c. it is mutable.

###### d. we can store same value many times.

###### e. different datatype values can be stores in list.

###### f. list items seperate by , and enclosed within [ ]

l1 = [3,5,7,2,5,7,"hello",99,"hii"] ### indexing always starts from 0(zero)

print(l1)  
print(type(l1))

[3, 5, 7, 2, 5, 7, 'hello', 99, 'hii']  
<class 'list'>

l1[6] ## Slicing to be done only in squaree brackets

'hello'

# l1[start:stop:steps]  
# l1[include:exclude:jumps]  
l1[0:6:2] ## positive slicing

[3, 7, 5]

l1[-1]

'hii'

l1[-3::-1] ## Negative slicing

['hello', 7, 5, 2, 7, 5, 3]

l1[0] = 100  
print(l1)

[100, 5, 7, 2, 5, 7, 'hello', 99, 'hii']

print(len(l1))

9

## Tuple

###### a. It is Ordered Sequence

###### b. It is immutable (unchangable)

###### c. tuples are faster then list.

###### d. tuple items are seperated by , and enclosed within ( )

###### e. we can store duplicate values also.

t1 = (2,5,4,3,2,5,3,"hello","python")

print(t1,type(t1))

(2, 5, 4, 3, 2, 5, 3, 'hello', 'python') <class 'tuple'>

# t1[0] = 100  
# print(t1)

## Set

###### a. sets are unordered.

###### b. set values are seperated by , and enclosed within { }

###### c. we can do Union, Intersection

###### d. they dont have duplicates. ( Important )

s1 = {2,3,1,5,3,2,4,3,"hello","hello","pyhton"}

print(s1,type(s1))

{1, 2, 3, 4, 5, 'hello', 'pyhton'} <class 'set'>

# s1[0]  
len(s1)

7

## Dictionary

###### a. Dictionary are unordered collection of key-value pairs.

###### b. key is always unique but values can be same.

###### c. dictionary are enclosed within { }

###### d. key values are written as key : value

###### e. keys and values can be of any type ( int, float , string, list , tuple)

d1 = {"madhur":"chocolate","piyush":"dal-chawal","sayeed":1000,"list":[5,4,3,45,3,5,6,625]}

print(d1,len(d1),type(d1))

{'madhur': 'chocolate', 'piyush': 'dal-chawal', 'sayeed': 1000, 'list': [5, 4, 3, 45, 3, 5, 6, 625]} 4 <class 'dict'>

# Type Conversion

to converse the types

## int,str,floot,bool,list,tuple,.set ,dict

a = 1.9  
b = int(a)  
print(b)

1

a = 9  
b = float(a)  
print(b)

9.0

a = "hello" ## alphabets cannot be convert into int or float  
s = "83"  
b = float(s)  
b = int(b)  
print(b)

83

l1 = [3,5,7,2,5,7,"hello",99,"hii"]  
print(l1)

[3, 5, 7, 2, 5, 7, 'hello', 99, 'hii']

t = tuple(l1)  
print(t)

(3, 5, 7, 2, 5, 7, 'hello', 99, 'hii')

s = set(l1)  
print(s)

{2, 3, 99, 'hello', 5, 7, 'hii'}

## User Input

a = input() ### Default input is String  
print(a,type(a))

Sayeed  
Sayeed <class 'str'>

a = input("Enter Name number: ") ### Default input is String  
print(a,type(a))

Enter Name number: uttarsh  
uttarsh <class 'str'>

# user input 2 number and add  
a = int( input("Enter 1st Number: ") )  
b = int( input("Enter 2nd Number: ") )  
print(a+b)

Enter 1st Number: 6  
Enter 2nd Number: 3  
9

## Print Statement

a = (10)  
b = (20)  
c = (30)  
d = (40)  
print (a,b,c,d)

10 20 30 40

# print(obj , sep = " " , end = "\n" ) ## \n - new line , \t - tab(four spaces)  
print(a,b,a,b,b,a, sep=" " ,end="\n")  
print(a,b)

10 20 10 20 20 10  
10 20

print (a,b, sep= ' kjdnv,n ')

10 kjdnv,n 20

print(a,b , sep ='\n')

10  
20

print (a,b, sep = " and " , end = " and ")  
print (c,d)

10 and 20 and 30 40

# two value integer ,float ,string   
a = 23  
b = 10  
c = 3.4312  
d = 7.3525  
e = "hello"  
f = "4324"  
print (a+b)

33

### , format of print   
print ("the value of a " ,a)

the value of a 23

### + method of print   
print (e + f)

hello4324

print ("dbc"+"snc")

dbcsnc

print ("the value of a is " + str(a) + " the value of b is " + str(c))

the value of a is 23 the value of b is 3.4312

### f " ", f ' ' , method of print   
print (f'the value of a {a} and the srikant of b is {b}')

the value of a 23 and the srikant of b is 10

a = 10  
b = 16  
print("the sum of two number is",a+b,"here is string")

the sum of two number is 26 here is string 10

## Print Functions ( % and .format )

############ % Function ( %d - integer , %f - Float , %s - String )  
a = 5  
b = 3.14325  
c = "Piyush is here"  
print("The number a is %d and b is %f and c is %s "%(b,a,c))

The number a is 3 and b is 5.000000 and c is Piyush is here

#### .format()  
a = 5  
b = 3.14325  
c = "Piyush is here"  
print("the number a is {} and b is {} and c is {}".format(a,b,c))

the number a is 5 and b is 3.14325 and c is Piyush is here

#### indexing always starts from 0  
print("the number a is {0} and b is {1} and c is {2}".format(a,b,c))

the number a is 5 and b is 3.14325 and c is Piyush is here

print("the number a is {2} and b is {1} and c is {0}".format(a,b,c))

the number a is Piyush is here and b is 3.14325 and c is 5

print("the number a is {1} and b is {1} and c is {1} {2} {0}".format(a,b,c))

the number a is 3.14325 and b is 3.14325 and c is 3.14325 Piyush is here 5

#### F string  
print( f"This a value is {b}")

This a value is 3.14325

## Opreators

1. arithmetic operators
2. logical operator
3. comparison operator
4. bitwise operator
5. assignment operator
6. identity operator
7. membership oeprator

## Arithemetic Operator  
# + 'add', - 'subtract' , \* 'multiply' , \*\* 'exponential' , / 'division', // 'floor division', % 'modulus'  
a = 5  
b = 2  
print(a+b)  
print(a-b)  
print(a\*b)  
print(a\*\*b) ### 5 ^ 2  
print(a/b)  
print(a//b) ### convert into int  
print(a%b) ### remainder

7  
3  
10  
25  
2.5  
2  
1

### logical operator  
### and , or , not  
  
  
# T AND T = T  
# T AND F = F ### BOTH CONDITION SHOULD BE TRUE  
# F AND T = F  
# F AND F = F   
  
# T OR T = T  
# T OR F = T ### ANY ONE IS TRUE THEN TRUE  
# F OR T = T  
# F OR F = F   
  
# a = True  
### not(a) ### opposite  
##### output False  
  
a = False  
b = False  
  
print(not(a) or b)  
# print(not(a))

True

## Comparision

###### > , < , >= , <= , == , !=

a = 5  
b = 6  
print(a=b)

---------------------------------------------------------------------------  
TypeError Traceback (most recent call last)  
<ipython-input-8-991fb8693e14> in <module>  
 1 a = 5  
 2 b = 6  
----> 3 print(a=b)  
  
TypeError: 'a' is an invalid keyword argument for print()

a = 5  
b = 10  
print(a!=b and a<b)

True

## Bitwise Operator

###### these operator have comparison of binary number.

###### & , | , ~ , ^ , >> , <<

a = 6 ### ### 0 = False , 1 = True  
b = 3 ###   
print(a & b)

2

a = 10 # 0000 1010 ### 0 = False , 1 = True  
b = 4 # 0000 0100  
 # 0000 1110 "or"  
print(a | b)

14

## Assignment operator

#### = , + = , - = , \* = ,\*\* = , / = , // = , % =

a = 8  
print(a)  
a += 2  
print(a)

8  
10

## Identity Operator

###### is , is\_not

a = 10  
b = 10  
print(a is b)  
print(a is not b)

True  
False

## Membership Operator

##### in , not in

a = "this is piyush"  
  
b = 'piyus'  
  
print(b in a)

True

## If - Else Statements

# 1. if  
# 2. if - else  
# 3. if - elif - else  
# 4. Nested if

# if (test statement True then run) :  
# body  
  
a = int(input("Enter number: "))  
if a<=0:  
 print("Positive")

# if (test statement True then run) :  
# body  
# else :  
# print("else")  
# bahar aajao  
  
a = int(input("Enter number: "))  
  
if a>=0 :  
 print("Positive")  
else:  
 print("Negative")

Enter number: 37  
Positive

# user input and check weather number is even or odd   
# take two sides from user input and check weather it is recantangle or square  
# two user input numbers and tell which one is greater

# if (test statement True then run) :  
# body  
# elif (test statement True then run) :  
# body  
# else :  
# print("else")  
a = -10  
if a>0:  
 print("positive")  
elif a<0:  
 print("Negative")  
else:  
 print("zero")  
# user input positive negative zero  
# three number user and check which among them is greatest 5 8 10 , 10 is greatest .format

Negative

# # nested if  
  
# if :  
# body  
# if :  
# if:  
# if:  
# body  
# elif:  
   
# else:  
# body  
# else:  
# body  
# else:  
# body  
  
# greater then 0 then check even ya odd

## For Loop

for i in range(2,10): # Range 0 by default and ends at given number-1  
 print(i)

2  
3  
4  
5  
6  
7  
8  
9

### Range (start(include) 0 , end(exclude) , step(skips) 1)  
  
for i in range(20,5,-2):  
 print(i)

20  
18  
16  
14  
12  
10  
8  
6

a = "Python is Advanced"  
print(len(a))  
# for i in range(0,len(a),2):  
# print(a[i])  
for i in a:  
 print(i)

18  
P  
y  
t  
h  
o  
n  
   
i  
s  
   
A  
d  
v  
a  
n  
c  
e  
d

l1 = [3,6,10,5]  
ans = 4   
for i in l1:  
 ans = ans \* i  
print("The sum of all numbers in a list is",ans)  
  
# sum of all numbers in the list b = 0   
# multiplication of all numbers in the list b = 1

The sum of all numbers in a list is 3600

## Pattern printing

there is a typical structure to print any pattern ,,, i.e the number of rows and colums in the pattern .

outer loop tells us the number the number of rows used and inner loop tells us the column used to print pattern

# \*  
# \*\*  
# \*\*\*  
# \*\*\*\*  
  
for row in range(4):  
 for col in range(row+1):  
 print("\*" ,end = "")  
 print()

\*  
\*\*  
\*\*\*  
\*\*\*\*

# \*\*\*\*  
# \*\*\*  
# \*\*  
# \*  
  
for row in range(5):  
 for col in range(5-row):  
 print("\*",end="")  
 print()

\*\*\*\*\*  
\*\*\*\*  
\*\*\*  
\*\*  
\*

# \*  
# \*\*  
# \*\*\*  
# \*\*\*\*  
for row in range(4):  
 for col in range(4-row):  
 print(" ",end="")  
 for col in range(row+1):  
 print("\*",end="")  
 print()

\*  
 \*\*  
 \*\*\*  
 \*\*\*\*

# 1  
# 12  
# 123  
# 1234  
for row in range(1,5):  
 for col in range(1,row+1):  
 print(row,end="")  
 print()  
  
# 1  
# 22  
# 333  
# 4444  
  
# 1  
# 21  
# 321  
# 4321  
  
# \*  
# \*\*  
# \*\*\*  
# \*\*\*\*  
# \*\*\*  
# \*\*  
# \*  
  
  
# \*  
# \* \*  
# \* \* \*  
# \* \* \* \*  
# \* \* \* \* \*   
  
# A  
# BC  
# DEF  
# GHIJ

1  
22  
333  
4444

x = 65  
print(chr(x)) ## chr command for getting character of a number

A

## While loop

while loop is used for two purpose .for checking the condition and run the loop (if else used for cheching the confition and for loop is used for the run loop) if else and for loop both

# while test\_statement:  
# body  
  
a = 0  
while a<10:  
 print(a)  
 a = a + 1  
print("the end")

0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
the end

a = 0  
   
 while a<10:  
 print(a)  
 while True:  
print("hello")

File "<ipython-input-6-e153548aefa7>", line 3  
 while a<10:  
 ^  
IndentationError: unexpected indent

user = int(input("enter number"))  
i = 1  
while False:  
 print (i)  
   
 i+=1  
print("the end")

# 1 - 100 numbers addition   
a = 1  
ans = 0  
  
while a <= 100:  
 ans = ans+a  
 a = a + 1  
print(ans)

# start  
# test\_statement  
# increment

## break and continue and Pass

# Break

It is used to break the loop

for i in range (10):  
 if i ==5:  
 break  
   
   
 else:  
 print(i)  
   
print ("outside the loop")

l1 = [2,55,6,3,"poison",6,7,3,6,6]  
for i in l1:  
 if i =="poison":  
 break  
 else:  
 print (i)  
print ("outside the loop")

# Continue

In continue when the value find then skip this value and continue

l1 = [2,55,6,3,"poison",6,7,3,6,6]  
for i in l1:  
 if i == "poison":  
 continue  
   
 else:  
 print (i)  
print ("outside the loop")

# Pass

if we want to any code it checking but do not run . and we don't want to error in input

num = 10  
if num<20:  
 print ("less then 20")  
else:  
 print ("greater then 20")

less then 20

### if we don't want to print   
num = 10  
if num<20:  
   
else:  
 print ("greater then 20")

File "<ipython-input-9-d9cd35a56cf6>", line 5  
 else:  
 ^  
IndentationError: expected an indented block

num = 10  
if num<20:  
 pass  
   
else:  
 print ("greater then 20")

# Datatype function

we learn about many datatype

-- Number,string,list ,tuple,set,dictionary

number don't have function (integer,float,complex,bool)

### String Function

1. lower
2. Upper
3. capitalize
4. count
5. startswith
6. endswith
7. find
8. index
9. replace
10. join
11. split

a = "this is tHe Example of StRinG fuNcTioNs"

print(a[0:10:2])

ti st

print(a) #### String functions does not effect the original string

this is tHe Example of StRinG fuNcTioNs

a.lower()

'this is the example of string functions'

a.upper()

'THIS IS THE EXAMPLE OF STRING FUNCTIONS'

a.capitalize()

'This is the example of string functions'

a.count("s")

3

a.startswith("T")

False

a.endswith("ioNs")

True

a.find("tHe",1)

8

a.find ("la")

-1

a.index("t",9)

24

a.index("j")

---------------------------------------------------------------------------  
NameError Traceback (most recent call last)  
<ipython-input-1-38eb640d9bb3> in <module>  
----> 1 a.index("j")  
  
NameError: name 'a' is not defined

a.replace("this","There",1)

'There is tHe Example of StRinG fuNcTioNs'

a.split("s") ### it devide o the basis of space by default

['thi', ' i', ' tHe Example of StRinG fuNcTioN', '']

l1 = ['thi', ' i', ' tHe Example of StRinG fuNcTioN', '']  
"sss".join(l1)

'thisss isss tHe Example of StRinG fuNcTioNsss'

## List Function

1. append
2. extend
3. copy
4. clear
5. pop
6. remove
7. insert
8. index
9. reverse
10. sort
11. count

l1 = [2,4,3,2,5,2,56,'hello world',63,6,6,'python']  
print(l1) #### list functions change original list

[2, 4, 3, 2, 5, 2, 56, 'hello world', 63, 6, 6, 'python']

## append - it is used to add one item in a list  
l1.append(999)

l1

[2, 4, 3, 2, 5, 2, 56, 'hello world', 63, 6, 6, 'python', 999, 999]

## Extend - it is used to add more then one item in a list  
l1.extend(["python",10])

l2=l1 ### this is not make a copy only l2 only point to l1

l2

[2, 4, 3, 2, 5, 2, 56, 'hello world', 63, 6, 6, 'python', 999, 999]

l2 = l1.copy() #### its make a copy of list

l2.clear() ### clear all the list

l1.pop(4) ### it gives some value from list(remove )it given 'index-1'it means from last

'world'

a = l1.pop()

a

999

l1.remove(2) ### shift+tab   
 ### here we give value for remove and it can remove first value

l1

[4, 3, 'hello', 2, 5, 2, 56, 'hello world', 63, 6, 6, 'python', 999]

l1.insert(2,"hello") ### In append we can add anything but it add in the last but In insert we can  
 ### add anthing in any position (index) and object

l1.index(999,0,7) ### this is used to find the index position

6

l1.reverse() ### It reverse all the list

l1.count("hello") ### it count all the index but it can't find single charcter and it gives 0 when  
 ### when index is not available

1

l1.sort() ### It put all

l1

['HELLO', 'HEllo', 'Hello', 'hello', 'india', 'world']

l1 = [9,55,66,15,54,454,66]

l1.sort()

l1

[9, 15, 54, 55, 66, 66, 454]

## Tuple

this is immutable (can't change)

1. count
2. index

t1 = (2,42,5,2,5,25,56,47,25)  
print(t1,type(t1),len(t1))

(2, 42, 5, 2, 5, 25, 56, 47, 25) <class 'tuple'> 9

t1[0]=44

---------------------------------------------------------------------------  
TypeError Traceback (most recent call last)  
<ipython-input-3-458f8f97aa8b> in <module>  
----> 1 t1[0]=44  
  
TypeError: 'tuple' object does not support item assignment

t1[2]

5

# Count

It count the value

t1.count(2)

2

# Index

this tells us the index position

t1.index(25,6)

8

## Set

they don't have duplicate ,set are unordered

1. union
2. intersection
3. difference
4. add
5. remove
6. Discard
7. issubset
8. issuperset
9. copy
10. clear
11. pop
12. update

s1 = {2,5,25,7,45,24,4,5,6,5,4,3,4}  
s2 = {"hello",2,3,6,"world"}  
print(s1,s2)

{2, 3, 4, 5, 6, 7, 45, 24, 25} {2, 3, 6, 'world', 'hello'}

s1.union(s2) ### It combine all the set

{2, 24, 25, 3, 4, 45, 5, 6, 7, 'hello', 'world'}

s1.intersection(s2) ## find the common in both

{2, 3, 6}

s2.difference(s1) ### It subtract from S1 all the index of S2

{'hello', 'world'}

s2.add("python") ### it is used to add anything in set .. In this position is not fix , and only one   
 ###v only one thing add

s2.remove(100) ### It use to remove any element   
 ### IT gives error when item is not available

---------------------------------------------------------------------------  
KeyError Traceback (most recent call last)  
<ipython-input-76-d5342f579284> in <module>  
----> 1 s2.remove(100)  
  
KeyError: 100

s2

{2, 3, 6, 'hello', 'python', 'world'}

s2

{3, 6, 'hello', 'python', 'world'}

# s2.discard(3) ### It also used for remove any element but it doesn't give error if item is not  
s2.discard(100) ### available

s2

{6, 'hello', 'python', 'world'}

s3 = s2.copy() ### it's used to make copy

s3.clear() ### this used to clear the set

s2.pop() ### its used to make rmove anything

'world'

s2.update({2,325,2,6,4,6,436}) ### its use to adding more then one element

s2

{2, 3, 325, 4, 436, 6, 'hello', 'world'}

s1 = {2,5,25,7,45,24,4,5,6,5,4,3,4}  
s2 = {5,7,45,25}  
s3 = {2,4,6,5}  
print(s1,s2,s3)

{2, 3, 4, 5, 6, 7, 45, 24, 25} {25, 45, 5, 7} {2, 4, 5, 6}

s2.issubset(s1) ### it check s2 is small part of s1 or not

True

s2.issuperset(s3 ### it means in s1 all the element of s2 and s3

False

## Dictionary

unordered collection

1. clear
2. copy
3. keys
4. values
5. pop
6. popitem
7. update
8. get

d1 = {"madhur":"chocolate","piyush":"dal-chawal","sayeed":1000,"list":[5,4,3,45,3,5,6,625]}  
print(d1,type(d1),len(d1))

{'madhur': 'chocolate', 'piyush': 'dal-chawal', 'sayeed': 1000, 'list': [5, 4, 3, 45, 3, 5, 6, 625]} <class 'dict'> 4

d1.clear() ## its used to clear

d2 = d1.copy() ### its make a copy

d1.keys() ### its used to find the all keys

dict\_keys(['madhur', 'piyush', 'sayeed', 'list'])

d1.values() ## its used to find all value

dict\_values(['chocolate', 'dal-chawal', 1000, [5, 4, 3, 45, 3, 5, 6, 625]])

d1.pop('piyush') ## its used to remove specific keys. and return the corresponding value

'dal-chawal'

d1.popitem() ### it remove from last item and return to make tuple

('list', [5, 4, 3, 45, 3, 5, 6, 625])

d1.update({1:'one',"two":2.224}) ### its used to add more then one item

d1["shahjahan"] = "chicken"

d1.get("madhur") ### it is used the value of keys

'lollipop'

d1["madhur"]

'lollipop'

# Function

if we can make function of any code then we can use this code several times

first we can make function then we can call

### def func1 (Argument):  
## body  
 ## func1(argument)

def func1():  
 print ("hello world")

func1()

hello world

### Note - when we want to run "hello world " simply run "func1" function

def func1(a,b):  
 print ("hello world",a+b) ## non\_return type function

func1() ### "error" because it require both argumet

---------------------------------------------------------------------------  
TypeError Traceback (most recent call last)  
<ipython-input-8-d88c41ef3303> in <module>  
----> 1 func1()  
  
TypeError: func1() missing 2 required positional arguments: 'a' and 'b'

func1(10,5)

hello world 15

func1(10,8)

hello world 18

def func2(a,b):  
 return ("hello world",a-b)

a= func2(10,6)  
print (a)

('hello world', 4)

### Note - it gives something in return so we can store in a

def func2(a,b):  
 return ("hello world","hey"+a-b)

a= func2(10,6)  
print (a) ### "error" because we can use string with number so we can convert number into string

---------------------------------------------------------------------------  
TypeError Traceback (most recent call last)  
<ipython-input-14-114c7e58c9a0> in <module>  
----> 1 a= func2(10,6)  
 2 print (a)  
  
<ipython-input-13-79d17d62bb55> in func2(a, b)  
 1 def func2(a,b):  
----> 2 return ("hello world","hey"+a-b)  
  
TypeError: can only concatenate str (not "int") to str

def func2(a,b):  
 return ("hello world"+ "hey"+ str(a-b))

func2 (10,6)  
print (a)

('hello world', 4)

########### Function  
# function to calculate area of rectangle  
# function to check a number is even or not.  
  
########## Recursive Function  
# print multiplication table of 4 using recursion  
  
  
def table(n,i):  
 print(n\*i)  
 i+=1  
 if i <=10:  
 table(n,i)  
  
table(4,1)

# positive negative zero  
  
# age above 18 vote eligible  
  
# table of 4 using loop

## anonymous function (lambda)

-- Anonymous or nameless function -- 'lambda ' is not a name ,but its a keyword -- lambda function is created using operator

a = lambda x,y : x+y  
print(a(10,7))

17

# syntax  
# lambda arguments: statement  
  
a = lambda x : True if x%2==0 else False  
print(a(10))

True

a = lambda x : print("positive") if x>0 else print("negative")  
a(5)

positive

## map -- If we can make any function it apply on every element   
l1 = [2,5,2,45,3,6,4,2,6,46]  
f = list(map(lambda x : x\*2 , l1))  
print(f)  
  
# s = "14 11 9 5 15" print(addition ans)

[4, 10, 4, 90, 6, 12, 8, 4, 12, 92]

## filter - In this wecan filter true or flase value   
### -- filter take two thing 1- function 2- where to apply  
l1 = [2,5,2,45,3,6,4,2,6,46]  
f = list(filter(lambda x : x%2!=0 , l1))  
print(f)  
  
# l2 = [2,2,4,2,5,"hello","heyy",2,"python"]

[5, 45, 3]

### Global and Local variable

if we write a program and we give any value it means this value is used for all the program this is

global varible .. and we can make a function in program and if we can give any value to this function

this used to local variable

# Note

if we can give any value in local varible only it can run in local but when we can give

### Rules of Global keyword

1. when we create a variable inside a function,it is local by default.
2. when we define a variable outside of a function,it is global by default. you don't have to use global keyword.
3. we use global keyword to read and write a global variable inside a function.
4. Use of global keyword outsidea function has no effect.

d = 1 # global Variable  
  
def add():  
 print(d)  
  
add()

1

def add():  
  
  
 c = c+5  
 print(c)  
   
c = 1   
add()  
  
print("Main value of c ",c)

---------------------------------------------------------------------------  
UnboundLocalError Traceback (most recent call last)  
<ipython-input-5-5d3e9134bb20> in <module>  
 6   
 7 c = 1  
----> 8 add()  
 9   
 10 print("Main value of c ",c)  
  
<ipython-input-5-5d3e9134bb20> in add()  
 2   
 3   
----> 4 c = c+5  
 5 print(c)  
 6   
  
UnboundLocalError: local variable 'c' referenced before assignment

def local():  
 l = 5 # Local Keyword  
 print(l)  
   
local()

5

c = 5   
def func2():  
 c+=10  
 print ("c is ",c)  
print ("c outside is ",c)  
func2() ### error we can global varible in local but we can't change   
  
### if we want to make change then we can use "glbal keyword"

c outside is 5

---------------------------------------------------------------------------  
UnboundLocalError Traceback (most recent call last)  
<ipython-input-1-a117d2702056> in <module>  
 4 print ("c is ",c)  
 5 print ("c outside is ",c)  
----> 6 func2()  
  
<ipython-input-1-a117d2702056> in func2()  
 1 c = 5  
 2 def func2():  
----> 3 c+=10  
 4 print ("c is ",c)  
 5 print ("c outside is ",c)  
  
UnboundLocalError: local variable 'c' referenced before assignment

def func1():  
 x = 20  
   
 def func2():  
   
 x = 25  
   
 print("Before calling func2: ",x)  
 print("Calling func2 now")  
 func2()  
 print("after calling func2 : ",x)  
  
func1()  
  
print("main x value is ",x)

Before calling func2: 20  
Calling func2 now  
after calling func2 : 20  
main x value is 25

def func1():  
 s = "i like paris"  
 print(s)  
   
s = "i love india"  
func1()  
print(s)

def func1():  
 global s  
 print(s)  
 s = "here it is."  
 print(s)  
  
s = " now it is the time."  
func1()  
print(s)

def func1(x,y):  
 global a  
 a = 42  
 x,y = y,x  
 b = 12  
 b = 17  
 print(a,b,x,y)  
   
a, b, x, y = 1,15,3,4  
func1(21,7)  
print(a,b,x,y)

# File handling

r - read (default)

w - write - open a file for writing ,creates a new file if not exists then clear all content of file

x - excute - open file if exist and give error if not exist

a - append - open a file if exists and start writing from the end,otherwise new file create

t- test - (default)

b - binary

(+ ) - open file for updating (reading and writing both )

Note - when we open a file after that we have to save this

f = open("text2.txt","w")

f.close () ### to save file

f = open("text2.txt","w")

f.write("the first line is this.")  
f.write ("this is second line.")

20

f.close ()

## now we want to write in new line   
f = open ("test2.txt","w")  
f. write ("this is first line. \n this is second line.")

42

f.close()

# Append

f = open ("text2.txt","a")  
f.write ("\n this is third line .\n this is last line")

41

f.close ()

f = open ("test.txt", "w") ### create file

f. close ()

f = open ("test.txt","w")

f.write ("this is first line") ## we write something

18

f.close () ## to save the file

f = open ("test1.txt","w") ### it clear all data from file

f.close ()

f = open ("test1.txt","w")  
f. write ("this is first line. \n this is second line.")

42

f = open ("test1.txt","a")  
f.write ("\n this is third line .\n this is last line")

41

### now we can use read mode   
f = open ("test2.txt") ### read and text by default

f. read () ### if no argumeent pass then this will read whole file

'this is first line. \n this is second line.'

### if we want to read more this   
f.read () ### after last it can't fine anything our curser in last

''

### if we want read only 3 charcter  
f = open ("text2.txt")  
f.read (3)

'the'

f. read (6) ### it start after 3 charcter

's firs'

### if we want to find the position of curser   
f. tell ()

12

## if we want to change the position of curser   
f.seek(0)

0

f.tell()

0

### now we want to read only one line   
f. readline () ## it close whereit get \n now curser go in seond line

'this is first line. \n'

f. readline ()

' this is second line.'

f. readline ()

' first line is this.this is second line.\n'

f.readline()

' this is last line'

f.readline ()

''

## now we want to read all line and store in list

f. readlines()

['this is first line. \n',  
 ' this is second line.\n',  
 ' this is third line .\n',  
 ' this is last line']

### if we want to learn everyline from list then we use for loop  
for i in f.readlines():  
 print (i) ### no output because our curser in last position so we have to change  
 ###the curser postion

f.seek (0)

0

for i in f.readlines():  
 print (i)

the first line is this.this is second line.  
  
 this is third line .  
  
 this is last line  
  
 this is third line .  
  
 this is last line

# Directory

how to access any folder

### if we want to import OS   
import os

os.getcwd () ### tell current working directory

'C:\\Users\\Ankit'

os.chdir ("C:\\Users\\Ankit\\AppData") ### to chnage the location of a current working directory

os.getcwd()

'C:\\Users\\Ankit\\AppData'

os.listdir() ### to list all directories present inside it

['Local', 'LocalLow', 'Roaming']

### if we want to create directory (folder) in local   
os.mkdir("wow")

os.listdir()

['Local', 'LocalLow', 'Roaming', 'wow']

### if we want to rename of new folder   
os.rename ("wow","WoW")

os.listdir()

['ice-cream', 'Local', 'LocalLow', 'Roaming', 'WoW']

###how to remove empty folder  
os.rmdir ("WoW")

os.listdir()

['ice-cream', 'Local', 'LocalLow', 'Roaming']

### how to remove folder if anything in folder   
os.mkdir("class")

os.listdir()

['class', 'ice-cream', 'Local', 'LocalLow', 'Roaming']

###chnage directory   
os.chdir ("C:\\users\\Ankit\\AppData\\class")

os.getcwd()

'C:\\users\\Ankit\\AppData\\class'

### now we can create one more folder in 'class'  
os.mkdir ("room")

os.chdir ("C:\\users\\Ankit\\AppData") ### now we come on Appdata folder

os.getcwd ()

'C:\\users\\Ankit\\AppData'

os.listdir()

['class', 'ice-cream', 'Local', 'LocalLow', 'Roaming']

### now we can delete folder   
os.rmdir ("class") ### it remove only empty folder

---------------------------------------------------------------------------  
OSError Traceback (most recent call last)  
<ipython-input-29-6e76ab05d034> in <module>  
 1 ### now we can delete folder  
----> 2 os.rmdir ("class")  
  
OSError: [WinError 145] The directory is not empty: 'class'

import shutil ### shutil remove every folder

shutil.rmtree ("class") ## delete all folder

os.listdir ()

['ice-cream', 'Local', 'LocalLow', 'Roaming']

### how wecan copy all folder by using shutil   
shutil.copy ("C:\\users\\Ankit\\AppData\\ice-cream","C:\\users\\Ankit")

---------------------------------------------------------------------------  
NameError Traceback (most recent call last)  
<ipython-input-1-0beb25d4a2ec> in <module>  
 1 ### how wecan copy all folder by using shutil  
----> 2 shutil.copy ("C:\\users\\Ankit\\AppData\\ice-cream","C:\\users\\Ankit")  
  
NameError: name 'shutil' is not defined

shhutil.move ()

# Exception handling

if we want to do something but that is wrong

eg; if want to change a in integer

int ("a")

---------------------------------------------------------------------------  
ValueError Traceback (most recent call last)  
<ipython-input-2-a493060d763c> in <module>  
----> 1 int ("a")  
  
ValueError: invalid literal for int() with base 10: 'a'

## like this we have many types of error   
if = 99

File "<ipython-input-3-0aa79e915dc8>", line 2  
 if = 99  
 ^  
SyntaxError: invalid syntax

### for handle all the error we have exception handling

# In this we have three things

1- try -- -it means try to do this . if it is done other wise go in exception

2- exception -if there is an error it try to handle

3- finally - it means error comes or not but run this

### now we can learn how we can use file handling in exceptional handling  
  
### Note --- as we know we learn in file handling if we open so we have to close these files

### now we want to open the file but all times don't need to close   
 ### so we can use "with "keyword

with open ("text2.txt","rt")as f:  
 print (f.read())

the first line is this.this is second line.  
 this is third line .  
 this is last line  
 this is third line .  
 this is last line

### now we learn how we can use try and exceptions in file

try:  
 f = open ("test44.txt",'"w"')  
 f.write ("hello again")  
 f.read ()  
finally:  
 f.close()

---------------------------------------------------------------------------  
ValueError Traceback (most recent call last)  
<ipython-input-8-3212691e620b> in <module>  
 1 try:  
----> 2 f = open ("test44.txt",'"w"')  
 3 f.write ("hello again")  
 4 f.read ()  
 5 finally:  
  
ValueError: invalid mode: '"w"'

try:  
 i= int (input("Enter:"))  
finally:  
 print ("the end") ### error becuase we run only int

Enter:abcd jaysurya  
the end

---------------------------------------------------------------------------  
ValueError Traceback (most recent call last)  
<ipython-input-2-044835ed5bad> in <module>  
 1 try:  
----> 2 i= int (input("Enter:"))  
 3 finally:  
 4 print ("the end") ### error becuase we run only int  
  
ValueError: invalid literal for int() with base 10: 'abcd jaysurya'

### now we use exception

l1 = ["b",0,2]  
for i in l1:  
 try:  
 ans = 1/int (i)  
 print (ans)  
 except Exception as e:  
 print (e)

invalid literal for int() with base 10: 'b'  
division by zero  
0.5

### now we use try,except ,and finally

l1 = ["b",0,2]  
for i in l1:  
 try:  
 ans= 1/int (i)  
 print (ans)  
 except Exception as e:  
 print (e)  
 finally:  
 print ("code complete")

invalid literal for int() with base 10: 'b'  
code complete  
division by zero  
code complete  
0.5  
code complete

# Class and Object

python is an object oriented programming language

almost everything in python is an object ,with its properties and method

A class is like an object constructor ,or a "blue print" for creating objects

class student:  
 pass  
a = student ()  
a.name = "jaysurya"

print (a.name)

jaysurya

class student:  
 pass  
a= student ()  
b= student ()  
a.name = "Jaysurya"  
a.age = 26  
b.name = "Srikanth"

print (a.name)

Jaysurya

class student:  
 no\_of\_leave = 10  
 def details(self): ### class inheritance  
 print ("the Name is {} and std is {} and sec is {} and leave are {}".format (self.name,self.std,self.sec,self.no\_of\_leave))  
   
a = student ()  
b = student ()  
a.name = "rajan"  
a.std = 10  
a.sec =1  
  
  
  
  
### b.name = "rahul"  
# b.std = 11  
# b.sec = 2  
# b.no\_of\_leave = 20

a.details()

the Name is rajan and std is 10 and sec is 1 and leave are 10

class student :  
 def \_init\_(self,name,std,sec,leaves=10): ### constructor  
   
 self.name1 = name   
 self.std1 = std  
 self.sec1 = sec  
 self.leaves1 = leaves  
 def details(self):  
 print("The name is {} and std is {} and sec {} and leaves are {}".format(self.name,self.std1,self.sec1,self.leaves1))

a = student("Aahul",12,"A",20)

---------------------------------------------------------------------------  
TypeError Traceback (most recent call last)  
<ipython-input-39-8f2e210854c4> in <module>  
----> 1 a = student("Aahul",12,"A",20)  
  
TypeError: student() takes no arguments

# Python Inheritance

Inheritance allow us to define a class that inherits all the methods and properties from another class

Parent class - Is the class being inherited from, also called base class

Clild class - Is the class that inherits from another class ,also called derived class

Single Inherits - simply from parent class inherits any properties to in child class

class Parent:  
 def \_\_init\_\_(self):  
 self.hair = "Black"  
 self.eye = "blue"  
   
 def result(self):  
 print("the hair color is {}".format(self.hair))  
   
class child(Parent):  
 def \_\_init\_\_(self):  
 self.hair = "Brown"  
 Parent.\_\_init\_\_(self)  
   
   
 def details(self):  
 print("the hair color is {} and eye color is {}".format(self.hair,self.eye))

a = child()

a.details()

the hair color is Black and eye color is blue

# Multiple Inheritance :

class mom:  
 def \_\_init\_\_(self):  
 self.hair = "Brown"  
 self.eye = "Blue"  
   
 def result1(self):  
 print("the hair color is {} and eye color is {}".format(self.hair,self.eye))  
   
class dad:  
 def \_\_init\_\_(self):  
 self.hair = "Black"  
   
 def result2(self):  
 print("the hair color is {} and eye color is {}".format(self.hair,self.eye))  
   
class child(dad,mom):  
 def \_\_init\_\_(self):  
 self.eye="Green"  
 dad.\_\_init\_\_(self)

obj = child ()

obj.result2()

the hair color is Black and eye color is Green

# Multilevel inheritance

class grandf:  
 hair\_color = "Brown"  
 height = 165  
   
 def details(self):  
 print("hi i have {} {}".format(self.hair\_color,self.height))  
  
class father(grandf):  
 hair\_color = "Black"  
 weight = 55  
 def display(self):  
 print("hello {} {}".format(self.hair\_color,self.height))  
  
class child(father):  
 pass  
  
# obj = child()  
# print(obj.hair\_color)  
# obj.details()

Abstraction :– Hiding the real implementation from the user and telling only how to use it. Ex:- Game

Encapsulation: - Combination of both code and data is called encapsulation.

Real life Ex- capsule