# #-------------------------return function -----------------------------

def fun1(x,y):

    sum=x+y

    return sum

    #print(sum)

w=fun1(2,3)

print(w)

# #------------------------------non return type-----------------------------

def fun2(x,y):

    sum=x+y

    #return sum

    print(sum)

w=fun1(2,3)

print(w)

print("-----------------------------------default value----------------------)

def fun3(x="Bangladeh"):

    print("I am from "+ x)

fun3("India")

fun3("pakistan")

fun3() # default value show korte empty funtion

print("--------------------------passing a list in funtion--------“)

def fun4(phone):

    for x in phone:

        print(x)

s=("xomi","apple","i-phone")

fun4(s)

# #--------------------passing a range in funtion------------

def fun5():

    for x in range(1, 50):

        print(x)

fun5()

# #--------------------use inf if-elif condition in funtion------

def fun6(x,y):

    if x>y:

        print("x is bigger then y:", x)

    else:

        print("y is bigger then x:", y)

fun6(5,9)

print("-------------------------find even number using if-elif contion in funtion--")

def fun7(x):

    if x%2==0:

        print("even number:", x)

    else:

        print("odd number:", x)

fun7(90)

print("-----------------------------------find odd-even number using for loop and if-elif contion in funtion--")

def fun8():

    for x in range(1, 5):

        if x%2==0:

            print("all even number:", x)

fun8()

print("---------------------------find odd-even number by taking input and if-else contion in funtion--")

def fun9(x):

    if x%2==0:

        print("input number is even:", x)

    else:

        print("input number is odd:", x)

given=int(input("enter any number:"))

fun9(given)

# #---------------------------simple input data in function-----------

def fun10(x):

    print(x)

data=str(input("input name:"))

fun10(data)

print("------------------------------------combine input date with fixed data in funtion--")

def fun11(name):

    print(" my name is "+ name +" hossain")

enter=str(input("input your first name:",))

fun11(enter)

print("------------------------------------combine input digit with fixed digit in funtion--")

def fun12(math):

    print("sum is: ",5\*math)

    print("sum is: ",6\*math)

enter=int(input("input a number: "))

fun12(enter)

# #---------------using a build-in math formula----------

import math

print(math.tan(45))

def fun13(x,y):

    sum=x+y

    return sum

w=fun13(2,3)

print(w)

# #----------take multiple value from key board input and sum in function---

def fun14(x,y,z):

    sum=x+y+z

    print("sum of three number is: ",sum)

a=int(input("enter first number:"))

b=int(input("enter second number:"))

c=int(input("enter third number:"))

fun14(a,b,c)

# #-----------------------------use \*args and \*\*kwargs in funtion-------

def fun15(\*args):

    print(args)

fun15(1,2,3,5,8,9)

def fun15(\*args):

    print(args,type(args))# it funtion also so type of args, ans:tuple

fun15(1,2,3,5,8,9)

def fun15(capten,\*args):#in this funtion mixed parameter and \*args

    print("capten:",capten)

    print("other:",args)

fun15("mahmud","lamyaa","fariya","nowin","niva")

#this funtion output show as keyword formate, othat parameter+argument 2ti show korbe

def fun16(\*\*kwargs):

    print(kwargs)

fun16(a=2,b=4,c=6)

def fun16(\*\*kwargs):

    print(kwargs,type(kwargs))# this funtion aslo so type of kwargs,ans:dics

fun16(a=2,b=4,c=6)

def fun17(\*\*kwargs):

   # sum=\*\*kwargs

    for x in range(1,5):

        print("sum:",x)

fun17()

def fun18(\*\*kwargs):

    for x in kwargs:

        print(x)

fun18(a=1, b=2, c=3,d=4)

def fun19(\*\*kwargs):

    for args in kwargs:

        print("{0} : {1}".format(args, kwargs[args]))

fun19(a=1, b=2, c=3)

# #------------------------------------------simple range funtion------

def fun20():

    ww=list(range(2,100,10))

    print(ww)

fun20()

# #-----------------------------------mathemetical squre funtion--------

def fun21(x,y):

    squre=x\*x+2\*x\*y+y\*y

    print(squre)

fun21(5,5)

# #-----------------------------------genaral and formated style ------

def fun22(num):

    print("hello "+num)

    #Or

    print("wellcome {num}".format(num=num))#formated style

fun22("mahmud")

fun22("hossain")

# #---------------------------- positional agrument style in function-------

def fun23(name,age,roll):

    print(name,age,roll, sep=' ! ') #use separetor fromate sep=' wise ymble '

fun23("mahmud",23,2) #positional argument

fun23("hossain",25,3)

# #---------------------------- keyword agrument style in function----

def fun24(name,age,roll):

    print(name,age,roll, sep=' ! ')

fun24(name="mahmud",age=23,roll=2)#keyword argument

fun24(name="hossain",age=25,roll=5)

print("-------------------------------------------- mixed positional and keyword argument in funtion")

def fun25(name,age,roll):

    print(name,age,roll, sep='  : ')

fun25("mahmud",age=25,roll=5)#work , cause first keyword then positional argument

#fun25(name="mahmud",age=25,5) # not work ,cause first positional then keyword argument

print("-------------------------------------------- default value of a parameter argument in funtion")

#mone rakte hobe jodi default argumnent funtion a ,given value deya hoy tobe default value bad jabe and given value show korbe

def fun26(name,age,country="Bangladesh"):#Bangladesh is default value

    print(name,age,country)

fun26("mahmud",23)#here taken country is default argument value

fun26("Hossain",23,"japan")# here country is given argument value

print("----------------------------------------------------------------- Return type value funtion")

def fun27(x):

   return x+x

print(fun27(2), fun27(3), sep=' :  ')

def fun28(first\_name,last\_name):

    return first\_name+"\_"+last\_name

name=fun28("mahmud","hossain")

print(name)

print("-----------------------------optional argumnent base return funtion")

#ekhane parameter argument gulo optional kora hoyse ' ' empty symble diya, jodi ekta na take tobe onnoti hobe

def fun29(first\_name,last\_name='',midd\_name=''):#2nd and 3rd name optional argument

    complete\_name=first\_name

    if midd\_name:

        complete\_name=first\_name + midd\_name

    complete\_name=first\_name + midd\_name + last\_name

    return complete\_name

print(fun29("mahmud ", "hossain","niva "))# give all name

print(fun29("mahmud ", "hossain"))# give 1st and 2nd name

print(fun29("mahmud "," ", "lamyaa"))# give 1st and 3rd name

print(fun29("mahmud"))#give only 1st name

# #--------------------------- Lamda funtion-------------

#lamda funtion syntex: lamda argument/parameter:condition

frist\_example=(lambda x,y:x\*x+2\*x\*y+y\*y)(5,5)

print(frist\_example)

#             Or

print((lambda x,y:x\*x+2\*x\*y+y\*y)(5,5))

#    10/12/2019

second\_example=(lambda x:x\*x\*x)(3)

print(second\_example)

def hat(x,y):

    if x>y:

        print("x is big:", x-y)

    elif x==y:

        print("i am equal",x,"=" ,y)

    else:

        print("y is big:", y-x)

o=int(input("enter x:",))

p=int(input("enter y:",))

hat(o,p)

def mah(x,y):

    if x>y:

        return ("x is bigg:",x-y)

    elif x==y:

        return ("x and is equal:",x==y)

    else:

        return ("y is bigg:",y-x)

a=int(input("enter x:"))

b=int(input("enter y:"))

print(mah(a,b))

def fun():

    for q in range(1,100):

        print(q)

fun()

#print(a)

def name(first\_name,second\_name='',mid\_name=''):

    compalte\_name=first\_name

    if mid\_name:

        complate\_name=first\_name+mid\_name

    complate\_name=first\_name+mid\_name+second\_name

    return complate\_name

print(name("mahmud","hossain","nannu"))

def myfnc(x):

    print("inside myfnc", x)

    x = 10

    print("inside myfnc", x)

x = 20

myfnc(x)

print(x)

def make(\*args):

    sum=0

    for x in  args:

        sum +=x

    return sum

print(make(2,3,4))

def fun(a,\*args,\*\*kwargs):

    print(a)

    print(args)

    print(kwargs)

fun(2,3,4,5, x=1,y=2,z=3)

def fun(a,b,c,\*args,\*\*kwargs):

    print(a,b,c)

    print(args)

    print(kwargs)

fun(1,2,3,4,5,x=2)

def fun(cool):

    return cool+"love"

update=fun

update=("i love you ")

print(update)

# # import function

from math import pi, sqrt

print(pi)

print(sqrt(25))

import random

print(random.randint (1,5))

x=10

def fun():

    x=20

    print(x)

fun()

print(x)

#

# x=10

def fun():

    global x

    x=20

    print(x)

fun()

print(x)

def fun(x,y):

    sum=x+y

    return sum

hi=all(input("enter value of x:"))

my=int(input("enter value of y:"))

print(fun(hi,my))

def fun(\*args):

    print(args[4])

fun(1,2,3,4,5)

# #lambda function-----------

x=lambda lol:lol+5

print(x(5))

print((lambda x,y,z: x+y+z)(1,1,1))

def fun(n):

  return  lambda a : a \* n

x = fun(2)

print(x(2))

def fun(new):

    new=([1,1,1])

    print(new)

new = [2,2,2]

fun(new)

# #-----import function------------------------

import mymodule

mymodule.greeting("Jonathan")

mymodule.fun(2,2)

x=mymodule.fun1(5,5)

print(x)

a = mymodule.person1["age"]

print(a)

import mymodule as mx

a = mx.person1["age"]

print(a)

import platform

x = platform.system()

print(x)

x = dir(platform)

print(x)

from mymodule import \*

print (fun1)

# #function to another function------------

def fun():

    print("lololololo")

fun()

hi=fun

print(hi)

hi()

def fun():

    print("hello my dear friend\_1")

    def fun1():

        print("hello my dear friend\_2")

    fun1()

    print("hello my dear friend\_3")

fun()

def fun1():

    def fun2():

        print("hello mis dod")

    return fun2

hi=fun1()

print(hi)

hi()

def fun1():

   return fun1

#hi=fun1()

print("fun1")

fun1()

def hello():

    print("Hello World!")

def hi(func):

    print("Hi!")

    func()

hi(hello)

#26/12/2019

def fun(\*\*kw):

    print(kw)

fun(x=2,y=3,z=4)

def fun(\*kw):

    print(kw)

fun(2,3,4)

def fun(x,y):

    sum=x+y

    return x

print(fun(5,3))

print((lambda x,y:x+y)(25,25))

def fun():

    for x in range(5,10):

        print(x)

fun()

def fun():

    for x in range(1,20):

        if x%5==0:

            print("even number",x)

        else:

            print("odd number",x)

fun()

def fun(x):

    if x%2==0:

        print("even",x)

    else:

        print("odd",x)

go=int(input("enter a number x:"))

fun(go)

def fun(s):

    if s-2==0:

        print("right")

        print('\a') #for wrong/bip spund

    else:

        print("wrong")

go=int(input("enter number:"))

fun(go)

##\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*work with import module\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import modul1 #take module page or ekta niddisto module page er sokol funtion import kora hoyse, essa...>

print(modul1.fun1(3,4)) #moto j kono funtion run kora jabe, a jonne syntax:

                                                                #import module\_page\_name

                                                                #print(module\_page.funtion\_name(argument))

import modul1, modul2, modul3 # add multiple module page at a time

print(modul1.fun1(1,2))

print(modul2.fun2(1,2))

print(modul3.fun3(1,2))

      #OR

import modul1

import modul2

import modul3

print(modul1.fun1(1,2))

print(modul2.fun2(1,2))

print(modul3.fun3(1,2))

#27/12/2019

from modul4 import fun4 #use of from import format ,syntax : from modul page name import function name

print(fun4(2, 3))

def fun(x):

    while x<=10:

        print("value:",x)

        x=x+1

lol=int(input("enter a value:"))

print(fun(lol))

"

#27/12/2019

# in this program, we show that how to import all model funtion by using \* symbol

from modul5 import \*

print(fun1(2,2))

print(fun2(1,1))

# in this program,we show that,how to change funtion name to a new function name

from modul5 import fun2 as fun

print(fun(3,2))

#28/12/2019

#use defoult liabrary 'random'

from random import choice

list=[1,2,3,4,5,6,7,8,9]

for x in range(0,7):

    print("serial:",choice(list))

# use a pakage(multiple module or module liabrary)

#first creat a 'folder' pakage as wish

#cerate module/liabrary in this folder/pakage

#create \_\_init\_\_.py file in this folder

#call from pakage name import module name

from pack1 import modul1

from pack1 import modul2

print(modul1.fun1(1,1))

print(modul2.fun2(5,2))

#use multiple module from a pakage use \* symbol

from pack1 import \*

print(modul1.fun1(100,100))

# call all module from a pakage but use one module

from pack1.modul2 import \*

print(fun2(10,11))

#in this program ,we can show type of any varyable

x=1

print(x,type(x))

# in this program 'divmode' we can show vagfol and vagses at a time

s=10

r=2

result=divmod(s,r)

print(result)

#29/12/2019

#some dafoult funtion

x="mahmud"

print(x.title())

print(x.upper())

print(x.lower())

print(x.title().lower().upper())

#find any element in pre\_define line

s="i am mahmud hossain, i love my God ALLAH"

print(s.find('ALLAH'))

#replace pre\_define line by new element

x="i like Nowsin"

print(x.replace("Nowsin","Lamyaa"))

#replace multiple same word by new word

hi=" hello miss lamyaa and i like lamyaa"

print(hi.replace('lamyaa','niva'))

#use sep'!' function

x='raj'

y='dha'

z='khu'

print(x+'!'+y+'!'+z)

#OR

print(x,y,z,sep='!')

#strig interpulation / formatting

person="{name} is {age} year old"

print(person.format(name='mahmud',age=25))

#OR

person='%s is %d year old'

print(person %('mahmud',25))

#here, %s=string represent and %d=integer represent

#use break in while statement

x=1

while x<=100:

    print("read:",x)

    x=x+1

    if x==50:

        break

#use continue statement

x=0

while x<=20:

    x=x+1

    if x%2==0:

        continue

        #print(x)

    print(x)

#using for lop itarate a range and after show total sum of the range value

sum=0

for num in range(1,11):

    print(num)

    sum=num+1

print('total is:',sum)

#iterate a list element and give total element number at the end

x=[1,2,3,4,5,6,7,8,9]

sum=0

for s in x:

    print(s)

    sum=s+1

print("total:",sum)

#condition if different type of list element

sum=0

hi=[1,2,3,'mah',4,5.5,6,7,8,9]

for x in hi:

    print(x)

    if type(x)==int:

        sum=x+1

print("total:",sum)

#use format in different funtion

def fun(x):

    print("hello {x}".format(x=x))

fun("lamyaa")

fun("nowsin")

#OF

def fn(x):

    print("hello",x)

fn("lamyaa")

fn("nowsin")

#30/12/2019

# use of try,except,finally

#this program for non type error handeling

def fun(x,y):

    try:

        sum=x/y

    except ZeroDivisionError:

        print("given number y is zero:")

        return None

    except TypeError:

        print("given number y is string:")

        return None

    print("result:",sum)

fun(2,2)

fun(2,0)

fun("mahmud",2)

#this program is applicable for unknown type error handeling

def fun(x,y):

    try:

        div=x/y

    except:

        print("unknown error")

        return None

    print("divition is:",div)

fun("dd",2)

fun(2,2)

#why error happend/ error gotar karon janar jonne

def fun(x,y):

    try:

        div=x/y

    except Exception as e:

        print("cause of error hapend:", e)

        return None

    print("divition is:",div)

fun("dd",2)

fun(2,0)

#use of finally block

#fanally block run both condition error and non error condition, it mainly third condition which much show

def fun(x,y):

    try:

        div=x/y

        print("result:",div)

    except:

        print("unknown errpr:")

    finally:

        print("i am finally ,so run must be")

fun(2,2)

fun("mah",2)

fun(2,0)

#use pass in try,except

def fun(x,y):

    try:

        div=x/y

        pass

    except:

        pass

    finally:

        print("use pass")

fun("r",3)

#use try, exception, except formula

def fun(x,y):

    try:

        division=x/y

        print("good input:",division)

    except Exception as lol:

        print("error by:",lol)

        return None

    except:

        print("unknown error:")

        return None

(

fun(2,2)

fun("d",3)

#multiple exception handeling

try:

    file=open('file\_name')

    print("file is open")

except FileNotFoundError:

    print("file is open")

except PermissionError:

    print("permit not granted")

#OR

try:

    file=open("file\_name")

    print("file is open")

except (FileNotFoundError, PermissionError):

    print("file not found")

def fun(x,y):

    try:

        sum=x/y

        print("result:",sum)

    except Exception as e:

        print("unknown:",e)

fun(2,2)

fun(2,0)

fun("cool",2)

#create a file

with open('pactics/page1.txt','w') as fobj:

    coll=fobj.write("helloggggggggggggggggggggggggggggggggggggggggggggggggggggggggggggg")

    print(coll) #if use print, it count of totall world write

#open a file

with open('pactics/page1.txt','r') as fobj:

    cool=fobj.read()

    print(cool)

#using try, except find a file

try:

    with open('pactics/page1.txt','r') as r:

        lol=r.read()

        print(lol)

except Exception as e:

    print("do not found",e)

with open('pactics/page2.txt','w') as w:

    lol=w.write('ALLLAH is load')

    print(lol)

try:

    with open('pactics/page2.txt','r') as r:

        l=r.read()

        print(l)

except:

    print('do not found ')

#using try , Exceptipn

with open('pactics/page3.txt','w') as w:

    lol=w.write('coollllllllllllllllllllllllllllllllllllllllllllll')

try:

    with open('pactics/page33.txt','r') as r:

        lo=r.read()

        print(lo)

except Exception as e:

    print(e)

def fun():

    with open('pactics/page4.txt','w') as w:

        lol=w.write('assalamu alikum')

        print(lol)

    try:

        with open('pactics/page5.txt','r') as r:

            sol=r.read()

            print(sol)

    except Exception as e:

        print('error by:',e)

fun()

#using emumerate funtipon ,read a text file by serial number

with open('pactics/page2.txt','r') as r:

    for i,line in enumerate (r):

        print(i+1,line)

#read a file in upper mode

with open('pactics/page2.txt','r') as r:

    lol=r.read()

    print(lol.upper())

#using append add new line without delete previus line

with open('pactics/page2.txt','a') as a:

    lol=a.write('mohammad is massenger of God')

    print(lol)

#bangla write/read er jonne encoding='UTF-8' use korte hobe

with open('pactics/page5.txt','w',encoding='UTF-8') as w:

    lol=w.write('কোরআনে')

    print(lol)

with open('pactics/page5.txt','r',encoding='UTF-8') as r:

    col=r.read()

    print(col)

#checking existing file in same directory

import os

if os.path.exists('pactics/page3.txt'):

    print('yes it have')

def fun():

    try:

        with open('pactics/love.txt','w') as w:

            lol=w.write("cool cool coll")

    except Exception as e:

        print('cause of error:',e)

    except:

        pass

    try:

        with open('pactics/love3.txt','r') as r:

            col=r.read()

            print(col)

    except:

        print('not found')

fun()

#defoult import math modeule add/show korte press kore rakte hobe, Ctrl+space bur

from math import pow

print(pow(2,22))

from math import cos

print(cos(0))

from math import  degrees

print(degrees(1))

from math import sqrt

print(sqrt(9))

# add all default math module by using \*

from math import \*

print(pow(2,2))

#use of continue

x=1

while x<=10:

    if x==5:

        continue

    print(x)

    x=x+1

#use of break

x=1

while x<=10:

    if x==5:

        break

    print(x)

    x=x+1

#sum of total number

#using while loop

sum=0

x=1

while x<=5:

    sum=sum+x

    x=x+1

print(sum)

#using range function

sum=0

for x in range(0,6):

    sum=sum+x

print(sum)

#user joto number type korbe sey porjonto sum korbe

sum=0

i=1

x=int(input("give a range number:"))

while i<=x:

    sum=sum+i

    i=i+1

print(sum)

for x in range(1,11):

    if x%2==0:

        sum=sum+x

        x=x+1

print(sum)

hight=float(input("enter hight:"))

base=float(input("enter base:"))

area=.5\*hight\*base

print(area)

#some math funtion

from math import \*

print(max(2,3,10,5,2))

print(sqrt(16))

print(min(1,10,2,0))

print(abs(-4))

def fun(\*kw):

    print("my name is ",kw)

fun("mahmud","hossain","nannu")

def fun(\*\*kw):

    print("he is a ",kw['one'])

fun(one="good",two="bad")

def fun(country="usa"):

    print("he come on from  "+country)

fun("bangladesh")

fun("choina")

fun()

def fun(x):

    for p in x:

        print('name is :',p)

g=['apple','cola','lol','tol']

fun(g)

dol=lambda x,y: x+y

print(dol(2,2))

print((lambda x,y:x-y)(5,2))

#To delete a file, you must import the OS module, and run its os.remove() function:

import os

os.remove('pactics/page3.txt')

#Check if File exist

import os

if os.path.exists("pactics/page2.txt"):

  print("yes this page is exist")

else:

  print("The file does not exist")

# Delete Folder

#To delete an entire folder, use the os.rmdir() method:

#Note: You can only remove empty folders.it not possible to remove full/non-empty folder

import os

os.rmdir("pactics/sos")

#Create a New File

#To create a new file in Python, use the open() method, with one of the following parameters:

#"x" - Create - will create a file, returns an error if the file exist

#"a" - Append - will create a file if the specified file does not exist

#"w" - Write - will create a file if the specified file does not exist

f= open('pactics/new.txt','x')

#OR

with open('pactics/new2.txt','x') as x:

    x.open()

#Read Only Parts of the File

#By default the read() method returns the whole text, but you can also specify

# how many characters you want to return:

#total page

with open('pactics/page2.txt','r') as r:

    lol=r.read()

    print(lol)

#fixed para/part

with open('pactics/page2.txt','r') as r:

    lol=r.read(9)

    print(lol)

#OR

f = open("pactics/page2.txt", "r")

print(f.read(5))

#Read Lines

#You can return one line by using the readline() method:

with open('pactics/page2.txt','r') as r:

    lol=r.readline()

    print(lol)

#if you read 5 number line,then you write/type readline() 3 time, show below

with open('pactics/page2.txt','r') as r:

    lol=r.readline()

    lol=r.readline()

    lol=r.readline()

    print(lol)

#By calling readline() two times, you can read the two first lines:

f = open("pactics/page2.txt", "r")

print(f.readline())

print(f.readline())

print(f.readline())

#By looping through the lines of the file, you can read the whole file, line by line:

with open('pactics/page.txt','r') as r:

    for x in r:

        print(x)

        #OR

f = open("pactics/page.txt", "r")

for x in f:

  print(x)

#Close Files, if you create program given below style

#It is a good practice to always close the file when you are done with it.

f = open("pactics/page.txt", "r")

print(f.readline())

f.close()

#string format method

#Syntax\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

#string.format(value1, value2...)

#Sometimes there are parts of a text that you do not control, maybe they come from a database,\_ \_

#or user input?

#To control such values, add placeholders (curly brackets {}) in the text,\_ \_

# and run the values through the format() method:

age=25

name='i am mahmud hossain, i {} year old'

print(name.format(age))

#multiple elements

x=2

y=3

sum=x+y

add='the {} and {} is sumession is: {}'

print(add.format(x,y,sum))

def fun(x,y):

    sum=x+y

    add='the two number,s sum is :{}'

    print(add.format(sum))

fun(2,3)

#taking kewboard input then ,formating

def fun(x,y):

    sum=x+y

    add='the two number sum is:{}'

    print(add.format(sum))

one=int(input('enter first number:'))

two=int(input('enter second number:'))

fun(one,two)

sum=0

for x in range(2,10):

        sum=sum+x

print('total:',sum)

sum=0

for x in range(0,6):

    sum=sum+x

print(sum)

txt = "We have {:=8} chickens."

print(txt.format(49))

txt = "The temperature is {:=8} degrees celsius."

print(txt.format(-5))

txt = "The binary version of {0} is {0:b}"

print(txt.format(2))

#Many Exceptions

#You can define as many exception blocks as you want..

def fun(x,y):

    try:

        sum=x+y

        print('summation:',sum)

    except NameError:

        print('i dont now cause of error')

    except:

        print('error:')

fun(2,'s')

#using else in try,except formula

try:

  print("Hello")

except:

  print("Something went wrong")

else:

  print("Nothing went wrong")

#error handaling by raise type

x = -1

if x < 0:

  raise Exception("Sorry, no numbers below zero")

x = "hello"

if not type(x) is int:

  raise TypeError("Only integers are allowed")

#03/01/2020

#Check if the string starts with "The" and ends with "Spain":

import re

str = "The rain in Spain"

x = re.search(r"\bS\w+", str)

print(x.group())

#..................................revesion part from w3 school........................

#Python allows you to assign values to multiple variables in one line:

x, y, z = "Orange", "Banana", "Cherry"

print(x)

print(y)

print(z)

#And you can assign the same value to multiple variables in one line:

x=y=z='blue'

print(x,y,z)

   #OR

print(x)

print(y)

print(z)

#To combine both text and a variable, Python uses the + character:

x='hello'

print('python is good '+x+' python')

#You can also use the + character to add a variable to another variable:

x='mahmud '

y=' hossain'

print(x+y)

#For numbers, the + character works as a mathematical operator:

x=10

y=10

print(x+y)

#If you try to combine a string and a number, Python will give you an error:

#x='hello'

#y=10

#print(x+y)

#Global Variables

#Variables that are created outside of a function

#Global variables can be used by everyone, both inside of functions and outside.

#Create a variable outside of a function, and use it inside the function

x = "awesome"

def myfunc():

    print("Python is " + x)

myfunc()

#Local variable

#Normally, when you create a variable inside a function, that variable is local, ........

# and can only be used inside that function.

#If you create a variable with the same name inside a function, this variable will be local,.......

# and can only be used inside the function

#Create a variable inside a function, with the same name as the global variable

x='awosem'

def fun():

    x='very nice'

    print('python is '+x)

fun()

print('python is '+x)

#To create a global variable inside a function, you can use the global keyword.

def fun():

    x='hello1'

    global y

    y='hello2'

    print(x)

    print(y)

fun()

#print(x)

print(y)

#To change the value of a global variable inside a function,......................

#  refer to the variable by using the global keyword:

x = "awesome"

def myfunc():

  global x

  x = "fantastic"

myfunc()

print("Python is " + x)

#...........Built-in Data Types in python..........

#Text Type: str

#Numeric Types: int, float, complex

#Sequence Types:    list, tuple, range

#Mapping Type:  dict

#Set Types: set, frozenset

#Boolean Type:  bool

#Binary Types:  bytes, bytearray, memoryview

#You can get the data type of any object by using the type() function:

x='hello'

y=2

z=3.2

o=True

p=range(2,5)

q=(1,2,3,4,5)

r=[1,2,3,4,5]

s={1,2,3,4,5}

t =1j

u={"name" : "John", "age" : 36}

v=b"Hello"

print(type(x))

print(type(y))

print(type(z))

print(type(o))

print(type(p))

print(type(q))

print(type(r))

print(type(s))

print(type(t))

print(type(u))

print(type(v))

#If you want to specify the data type, you can use the following constructor functions:

#x = str("Hello World") str

#x = int(20)    int

#x = float(20.5)    float

#x = complex(1j)    complex

#x = list(("apple", "banana", "cherry"))    list

#x = tuple(("apple", "banana", "cherry"))   tuple

#x = range(6)   range

#x = dict(name="John", age=36)  dict

#x = set(("apple", "banana", "cherry")) set

#x = frozenset(("apple", "banana", "cherry"))   frozenset

#x = bool(5)    bool

#x = bytes(5)   bytes

#x = bytearray(5)   bytearray

#x = memoryview(bytes(5))   memoryview

#Float can also be scientific numbers with an "e" to indicate the power of 10.

x = 35e3

y = 12E4

z = -87.7e100

print(type(x))

print(type(y))

print(type(z))

#Complex numbers are written with a "j" as the imaginary part:

x = 3+5j

y = 5j

z = -5j

print(type(x))

print(type(y))

print(type(z))

#Random Number

#Python does not have a random() function to make a random number, .........

#but Python has a built-in module called random that can be used to make random numbers:

import random

print(random.randrange(2,10))

#Specify a Variable Type

#integer

x = int(1)   # x will be 1

y = int(2.8) # y will be 2

z = int("3") # z will be 3

print(type(x))

print(type(y))

print(type(z))

#float

x = float(1)     # x will be 1.0

y = float(2.8)   # y will be 2.8

z = float("3")   # z will be 3.0

w = float("4.2") # w will be 4.2

print(type(x))

print(type(y))

print(type(z))

#strings

x = str("s1") # x will be 's1'

y = str(2)    # y will be '2'

z = str(3.0)  # z will be '3.0'

print(type(x))

print(type(y))

print(type(z))

#You can assign a multiline string to a variable by using three quotes:

a = Lorem ipsum dolor sit amet,

consectetur adipiscing elit,

sed do eiusmod tempor incididunt

ut labore et dolore magna aliqua.

print(a)

#Strings are Arrays

x='hello'

print(x[1])

#Slicing

x='hello miss world'

print(x[2:7])

#Negative Indexing

x='hello Layuaa'

print(x[-6])

print(x[-5:-2])

#String Length

#The len() function returns the length of a string:

x='hello world'

print(len(x))

#..................String Methods.........................

#The strip() method removes any whitespace from the beginning or the end:

x='    hello lamyaa'

print(x.strip())

#The lower() method returns the string in lower case:

x='Hello LAMYAA'

print(x.lower())

#The upper() method returns the string in upper case:

x='hello lamyaa'

print(x.upper())

#The replace() method replaces a string with another string:

x='hello Lamyaa'

print(x.replace('L','N')) #only a latter replace

print(x.replace('Lamyaa','Nowsin')) #a holy word replace

print(x.replace('hello Lamyaa','how are you Nowsin')) #a holy Sentence replace

#The split() method splits the string into substrings if it finds instances of the separator:

a = "Hello, World!"

print(a.split(",")) # returns ['Hello', ' World!']

#..................Check String....................

#To check if a certain phrase or character is present in a string, we can use the keywords in or not in

x='i like you lamyaa'

y='lamyaa' in x

print(y)

txt = "The rain in Spain stays mainly in the plain"

x = "ain" not in txt

print(x)

#String Concatenation

a = "Hello"

b = "World"

c = a + b

print(c)

#.............String Format..............

#first method

x='hello'

y='world'

print(x+' miss '+y)

#second method

#The format() method takes the passed arguments, formats them,...

# and places them in the string where the placeholders {} are:

name='lamyaa'

x='i love you {}'

print(x.format(name))

#The format() method takes unlimited number of arguments,...

#and are placed into the respective placeholders

quantity = 3

itemno = 567

price = 49.95

myorder = "I want {} pieces of item {} for {} dollars."

print(myorder.format(quantity, itemno, price))

#You can use index numbers {0} to be sure the arguments are placed in the correct placeholders:

quantity = 3

itemno = 567

price = 49.95

myorder = "I want to pay {2} dollars for {0} pieces of item {1}."

print(myorder.format(quantity, itemno, price))

#..............................Escape Character..........

#txt = "We are the so-called "Vikings" from the north." output error show

#print(txt)

#To fix this problem, use the escape character \":

txt = "We are the so-called \"Vikings\" from the north."

print(txt)

#Other escape characters used in Python:

#Code   Result  Try it

"""

#\' Single Quote

#\\ Backslash

#\n New Line

#\r Carriage Return

#\t Tab

#\b Backspace

#\f Form Feed

#\ooo   Octal value

#\xhh   Hex value

"""

#String Methods

#All string methods returns new values. They do not change the original string.

#Method Description

#capitalize()   Converts the first character to upper case

#casefold() Converts string into lower case

#center()   Returns a centered string

#count()    Returns the number of times a specified value occurs in a string

#encode()   Returns an encoded version of the string

#endswith() Returns true if the string ends with the specified value

#expandtabs()   Sets the tab size of the string

#find() Searches the string for a specified value and returns the position of where it was found

#format()   Formats specified values in a string

#format\_map()   Formats specified values in a string

#index()    Searches the string for a specified value and returns the position of where it was found

#isalnum()  Returns True if all characters in the string are alphanumeric

#isalpha()  Returns True if all characters in the string are in the alphabet

#isdecimal()    Returns True if all characters in the string are decimals

#isdigit()  Returns True if all characters in the string are digits

#isidentifier() Returns True if the string is an identifier

#islower()  Returns True if all characters in the string are lower case

#isnumeric()    Returns True if all characters in the string are numeric

#isprintable()  Returns True if all characters in the string are printable

#isspace()  Returns True if all characters in the string are whitespaces

#istitle()  Returns True if the string follows the rules of a title

#isupper()  Returns True if all characters in the string are upper case

#join() Joins the elements of an iterable to the end of the string

#ljust()    Returns a left justified version of the string

#lower()    Converts a string into lower case

#lstrip()   Returns a left trim version of the string

#maketrans()    Returns a translation table to be used in translations

#partition()    Returns a tuple where the string is parted into three parts

#replace()  Returns a string where a specified value is replaced with a specified value

#rfind()    Searches the string for a specified value and returns the last position of where it was found

#rindex()   Searches the string for a specified value and returns the last position of where it was found

#rjust()    Returns a right justified version of the string

#rpartition()   Returns a tuple where the string is parted into three parts

#rsplit()   Splits the string at the specified separator, and returns a list

#rstrip()   Returns a right trim version of the string

#split()    Splits the string at the specified separator, and returns a list

#splitlines()   Splits the string at line breaks and returns a list

#startswith()   Returns true if the string starts with the specified value

#strip()    Returns a trimmed version of the string

#swapcase() Swaps cases, lower case becomes upper case and vice versa

#title()    Converts the first character of each word to upper case

#translate()    Returns a translated string

#upper()    Converts a string into upper case

#zfill()    Fills the string with a specified number of 0 values at the beginning

#..........................Python Booleans.................................

#Booleans represent one of two values: True or False

print(3>1)

print(10<2)

#The bool() function allows you to evaluate any value, and give you True or False in return,

print(bool("Hello"))

print(bool(15))

#Some Values are False

#except empty values, such as (), [], {}, "", the number 0, and the value None. ...

# And of course the value False evaluates to False.

print(bool(False))

print(bool(None))

print(bool(0))

print(bool(""))

print(bool(()))

print(bool([]))

print(bool({}))

#Functions can Return a Boolean

#Python also has many built-in functions that returns a boolean value, like the isinstance() function,...

#  which can be used to determine if an object is of a certain data type:

x = 200

print(isinstance(x, int))

print(isinstance(x,float))

print(isinstance(x, str))

#Python Operators

#Operator   Name    Example Try it

#+  Addition    x + y

#-  Subtraction x - y

#\*  Multiplication  x \* y

#/  Division    x / y

#%  Modulus x % y

#\*\* Exponentiation  x \*\* y

#// Floor division  x // y

#Exponentiation

x=4

ex=x\*\*2

print(ex)

#Floor division

x=10

y=2

s=x//2

print(s)

#Operator   Example Same As

#=  x = 5   x = 5

#+= x += 3  x = x + 3

#-= x -= 3  x = x - 3

#\*= x \*= 3  x = x \* 3

#/= x /= 3  x = x / 3

#%= x %= 3  x = x % 3

#//=    x //= 3 x = x // 3

#\*\*=    x \*\*= 3 x = x \*\* 3

#&= x &= 3  x = x & 3

#|= x |= 3  x = x | 3

#^= x ^= 3  x = x ^ 3

#>>=    x >>= 3 x = x >> 3

#<<=    x <<= 3 x = x << 3

x=x%3

print(x)

x=x//3

print(x)

x=x\*\*5

print(x)

#Python Comparison Operators

#Operator   Name    Example

#== Equal   x == y

#!= Not equal   x != y

#>  Greater than    x > y

#<  Less than   x < y

#>= Greater than or equal to    x >= y

#<= Less than or equal to   x <= y

x=3

y=4

print(x==y)

print(x!=y)

print(x<y)

print(x>y)

print(x<=y)

print(x>=y)

#Python Logical Operators

#Operator   Description Example

#and    Returns True if both statements are true    x < 5 and  x < 10

#or Returns True if one of the statements is true   x < 5 or x < 4

#not    Reverse the result, returns False if the result is true not(x < 5 and x < 10)

print(x<3 and x<9)

#Python Identity Operators

#Operator   Description Example

#is     Returns true if both variables are the same object  x is y

#is not Returns true if both variables are not the same object  x is not y

x='hello lamyaa'

y='hello lamyaa'

z='cool my baby'

print(x is y)

print(x is not z)

#Python Membership Operators

#Operator   Description Example

#in     Returns True if a sequence with the specified value is present in the object    x in y

#not in Returns True if a sequence with the specified value is not present in the object    x not in y

x='hello lamyaa'

print('hello' in x)

print('cool' not in x)

#Python Bitwise Operators

#Bitwise operators are used to compare (binary) numbers:

#Operator   Name    Description

#&  AND Sets each bit to 1 if both bits are 1

#|  OR  Sets each bit to 1 if one of two bits is 1

# ^ XOR Sets each bit to 1 if only one of two bits is 1

#~  NOT Inverts all the bits

#<< Zero fill left shift,Shift left by pushing zeros in from the right and let the leftmost bits fall off

#>> Signed right shift,Shift right by pushing copies of the leftmost bit in from the left, and let the rightmost bits fall off

#........................python list.......................................

x=['apple','banana','mango']

print(x)

#You access the list items by referring to the index number:

x=['abb','acc','add','ass']

print(x[2])

#Negative indexing means beginning from the end, -1 refers to the last item,

x=['abb','acc','add','ass']

print(x[-1])

#You can specify a range of indexes by specifying where to start and where to end the range.

x=[1,2,3,4,5,6,7,8,9]

print(x[2:5])

#only start range

print(x[2:])

#only ending range

print(x[:7])

#To change the value of a specific item, refer to the index number:

x=[1,2,3,4,5,'lol',6,7,8,9]

x[5]='lamyaa'

print(x)

#You can loop through the list items by using a for loop:

x=['apple','kola','peyara','jam']

for s in x:

    print(s)

#To determine if a specified item is present in a list use the in keyword:

x=['apple','kola','peyara','jam']

if 'jam' in x:

    print('yes have')

else:

    print('not have')

#To determine how many items a list has, use the len() function:

x=['apple','kola','peyara','jam']

print(len(x))

#Add Items

#To add an item to the end of the list, use the append() method:

x=['apple','kola','peyara','jam']

x.append('lamyaa')

print(x)

#To add an item at the specified index, use the insert() method:

x=['apple','kola','peyara','jam']

x.insert(1,'nowsin')

print(x)

#Remove Item

#The remove() method removes the specified item:

x=['apple','kola','peyara','jam']

x.remove('jam')

print(x)

#The pop() method removes the specified index, (or the last item if index is not specified):

x=['apple','kola','peyara','jam']

x.pop()

#OR

x.pop(1)

print(x)

#The del keyword removes the specified index:

x=['apple','kola','peyara','jam']

del x[3]

print(x)

#The del keyword can also delete the list completely:

#x=['apple','kola','peyara','jam']

#del x

#print(x)#

#The clear() method empties the list:

x=['apple','kola','peyara','jam']

x.clear()

print(x)

#Copy a List

#Make a copy of a list with the copy() method:

x=['apple','kola','peyara','jam']

y=x.copy()

print(y)

#Another way to make a copy is to use the built-in method list().

x=['apple','kola','peyara','jam']

y=list(x)

print(y)

#Join Two Lists

#One of the easiest ways are by using the + operator.

list1 = ["a", "b" , "c"]

list2 = [1, 2, 3]

list3 = list1 + list2

print(list3)

#Append list2 into list1:

list1 = ["a", "b" , "c"]

list2 = [1, 2, 3]

for x in list2:

  list1.append(x)

print(list1)

#Or you can use the extend() method, which purpose is to add elements from one list to another list:

list1=[1,2,3,4,5]

list2=['ab','bc','cd','de','ef']

list1.extend(list2)

print(list1)

# #....................python List Methods and discription...................

#Method        Description

#append()   Adds an element at the end of the list

#clear()    Removes all the elements from the list

#copy() Returns a copy of the list

#count()    Returns the number of elements with the specified value

#extend()   Add the elements of a list (or any iterable), to the end of the current list

#index()    Returns the index of the first element with the specified value

#insert()   Adds an element at the specified position

#pop()  Removes the element at the specified position

#remove()   Removes the item with the specified value

#reverse()  Reverses the order of the list

#sort() Sorts the list

# #...........................Python Tuples....................................

#A tuple is a collection which is ordered and unchangeable. tuples are written with round brackets.

x=('a','b','c','d')

print(x)

#Access Tuple Items

#You can access tuple items by referring to the index number, inside square brackets:

x=('a','b','c','d')

print(x[2])

#Negative Indexing

#Negative indexing means beginning from the end, -1 refers to the last item,

x=('a','b','c','d')

print(x[-1])

#Range of Indexes

x=(1,2,3,4,5,6,7,8,9)

print(x[2:6])

#Range of Negative Indexes

x=(1,2,3,4,5,6,7,8,9)

print(x[-6:-2])

#Change Tuple Values

#Once a tuple is created, you cannot change its values. Tuples are unchangeable,...........

# or immutable as it also is called.

#But there is a workaround. You can convert the tuple into a list, ..........

#change the list, and convert the list back into a tuple.

x=('a','b','c','d','e')

y=list(x)

y[2]='abcde'

x=tuple(y)

print(x)

#loop intuple

x=('ab','bc','cd','de','ef')

for s in x:

    print(x)

#To determine if a specified item is present in a tuple use the in keyword:

x=('ab','bc','cd','de','ef')

if 'ab' in x:

    print('yes have')

#To determine how many items a tuple has, use the len() method:

x=('ab','bc','cd','de','ef')

print(len(x))

#Add Items

#Once a tuple is created, you cannot add items to it. Tuples are unchangeable.

#x = ("apple", "banana", "cherry")

#x[3] = "orange" # This will raise an error

#print(x)

#Create Tuple With One Item

#To create a tuple with only one item, you have add a comma after the item,...........

#unless Python will not recognize the variable as a tuple.

x=('abc') #not tuple

y=('abc',) #tuple

print(type(x))

print(type(y))

#Remove Items

#Note: You cannot remove items in a tuple.

#Tuples are unchangeable, so you cannot remove items from it, but you can delete the tuple completely:

#thistuple = ("apple", "banana", "cherry")

#del thistuple

#print(thistuple) #this will raise an error because the tuple no longer exists

#Join Two Tuples

#To join two or more tuples you can use the + operator:

tup1=('a','b','c','d','e')

tup2=(1,2,3,4,5)

tup3=tup1+tup2

print(tup3)

# #.............................Python Sets................................

#A set is a collection which is unordered and unindexed. In Python sets are written with curly brackets.

x={'a',1,'mahmud','www'}

print(x)

#Loop through the set

x={'a',1,'mahmud','www'}

for s in x:

    print(s)

#Check parameter in set,by in

x={'a',1,'mahmud','www'}

lol='www' in x

print(lol)

 #OR

if 'mahmud' in x:

    print('yes have')

#Change Items

#Once a set is created, you cannot change its items, but using list conversion method you can do it

x={'a',1,'mahmud','www'}

y=list(x)

y[3]='lamyaa'

x=set(y)

print(x)

#add a new item, you can add new items.

x={'a',1,'mahmud','www'}

x.add('nowsin')

print(x)

#Add multiple items to a set, using the update() method:

x={'a',1,'mahmud','www'}

x.update(['nowsin','lamyaa','ara'])

print(x)

#To determine how many items a set has, use the len() method.

x={'a',1,'mahmud','www'}

print(len(x))

#Remove Item

#To remove an item in a set, use the remove(), or the discard() method.

x={'a',1,'mahmud','www'}

x.remove('mahmud')

print(x)

#Note: If the item to remove does not exist, remove() will raise an error.

#x={'a',1,'mahmud','www'}

#x.remove('lamyaa')

#print(x)

#discard() method

x={'a',1,'mahmud','www'}

x.discard('mahmud')

print(x)

#Note: If the item to remove does not exist, discard() will NOT raise an error.

x={'a',1,'mahmud','www'}

x.discard('lamyaa')

print(x)

#pop method

#Remove the last item by using the pop() method:

#Note: Sets are unordered, so when using the pop() method, you will not know which item that gets removed.

x={"apple", "banana", "cherry"}

y=x.pop()

print(y)

print(x)

#The clear() method empties the set:

x={"apple", "banana", "cherry"}

x.clear()

print(x)

#The del keyword will delete the set completely:

#x={"apple", "banana", "cherry"}

#del x

#print(x)

#Join Two Sets

#You can use the union() method that returns a new set containing all items from both sets,......

#or the update() method that inserts all the items from one set into another:

#The union() method returns a new set with all items from both sets:

x={1,2,3,4,5}

y={'a','b','c','d','e'}

z=x.union(y)

print(z)

#The update() method inserts the items in set2 into set1:

set1={1,2,3,4,5}

set2={'a','b','c','d','e'}

set1.update(set2)

print(set1)

#Note: Both union() and update() will exclude any duplicate items

# #.....................some Set Methods......................

#Method Description

#add()  Adds an element to the set

#clear()    Removes all the elements from the set

#copy() Returns a copy of the set

#difference()   Returns a set containing the difference between two or more sets

#difference\_update()    Removes the items in this set that are also included in another, specified set

#discard()  Remove the specified item

#intersection() Returns a set, that is the intersection of two other sets

#intersection\_update()  Removes the items in this set that are not present in other, specified set(s)

#isdisjoint()   Returns whether two sets have a intersection or not

#issubset() Returns whether another set contains this set or not

#issuperset()   Returns whether this set contains another set or not

#pop()  Removes an element from the set

#remove()   Removes the specified element

#symmetric\_difference() Returns a set with the symmetric differences of two sets

#symmetric\_difference\_update()  inserts the symmetric differences from this set and another

#union()    Return a set containing the union of sets

#update()

# #....................Python Dictionaries...................

#In Python dictionaries are written with curly brackets, and they have keys and values.

x={

    'name':'mahmud',

    'roll':21,

    'distics': 'Rajshahi',

    'to': 'dhaka'

}

print(x)

#Accessing Items

#You can access the items of a dictionary by referring to its key name, inside square brackets:

x={

    'name':'mahmud',

    'roll':21,

    'distics': 'Rajshahi',

    'to': 'dhaka'

}

s=x['name']

print(s)

#There is also a method called get() that will give you the same result:

s={

    'name':'Lamyaa',

    'city':'pabna',

    'relation':'once Love',

    'finally':'not accepted'

}

print(s)

x=s.get('name')

y=s['relation']

print(x)

print(y)

#Change Values

#You can change the value of a specific item by referring to its key name:

s={

    'name':'Lamyaa',

    'roll':2,

    'city':'pabna'

}

print(s)

s['name']='Nowsin'

print(s)

#Loop Through a Dictionary

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

for x in s:

    print(x)

#Print all values in the dictionary, one by one:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

for x in s:

    print(s[x])

#You can also use the values() function to return values of a dictionary:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

for x in s.values():

    print(x)

#Loop through both keys and values, by using the items() function:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

for x in s.items():

    print(x)

#Check if Key Exists

#To determine if a specified key is present in a dictionary use the in keyword:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

if 'name' in s:

    print('yes have')

else:

    print('not have')

#Dictionary Length

#To determine how many items (key-value pairs) a dictionary has, use the len() method.

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(len(s))

#Adding Items

#Adding an item to the dictionary is done by using a new index key and assigning a value to it:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

s['relation']='love'

print(s)

#Removing Items

#The pop() method removes the item with the specified key name:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

s.pop('name')

print(s)

#The popitem() method removes the last inserted item

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

s.popitem()

print(s)

#The del keyword removes the item with the specified key name:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

del s['city']

print(s)

#The del keyword can also delete the dictionary completely:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

#del s

print(s)

#The clear() keyword empties the dictionary:

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

s.clear()

print(s)

#Copy a Dictionary

#You cannot copy a dictionary simply by typing dict2 = dict1,....

#  because: dict2 will only be a reference to dict1,.....

#  and changes made in dict1 will automatically also be made in dict2.

#There are ways to make a copy, one way is to use the built-in Dictionary method copy().

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

x=s.copy()

print(x)

#Another way to make a copy is to use the built-in method dict().

s={

    'name':'mahmud',

    'roll':2,

    'city':'Rajshhai',

    'to':'dhaka'

}

print(s)

x=dict(s)

print(x)

#Nested Dictionaries

#A dictionary can also contain many dictionaries, this is called nested dictionaries.

myfamily = {

  "child1" : {

    "name" : "Emil",

    "year" : 2004

  },

  "child2" : {

    "name" : "Tobias",

    "year" : 2007

  },

  "child3" : {

    "name" : "Linus",

    "year" : 2011

  }

}

print(myfamily)

#Create three dictionaries, than create one dictionary that will contain the other three dictionaries:

child1 = {

  "name" : "Emil",

  "year" : 2004

}

child2 = {

  "name" : "Tobias",

  "year" : 2007

}

child3 = {

  "name" : "Linus",

  "year" : 2011

}

myfamily = {

  "child1" : child1,

  "child2" : child2,

  "child3" : child3

}

#Dictionary Methods

#Method Description

#clear()    Removes all the elements from the dictionary

#copy() Returns a copy of the dictionary

#fromkeys() Returns a dictionary with the specified keys and values

#get()  Returns the value of the specified key

#items()    Returns a list containing a tuple for each key value pair

#keys() Returns a list containing the dictionary's keys

#pop()  Removes the element with the specified key

#popitem()  Removes the last inserted key-value pair

#setdefault()   Returns the value of the specified key. If the key does not exist: insert the key, with the specified value

#update()   Updates the dictionary with the specified key-value pairs

#values()   Returns a list of all the values in the dictionary

# #...........................condisional statement ,if,elif,else.................

def fun(x,y):

    if x>y:

        print('x is big:',x-y)

    elif x<y:

        print('y is big:',y-x)

    else:

        print('x and y is equal:',x and y)

go=int(input('enter the value of x:'))

back=int(input('enter the value of y:'))

fun(go,back)

#Short Hand If

x=2

y=1

if x>y:print('x is big:')

#Short Hand If ... Else

a = 2

b = 330

print("A") if a > b else print("B")

#You can also have multiple else statements on the same line:

a = 330

b = 330

print("A") if a > b else print("=") if a == b else print("B")

#The and keyword is a logical operator, and is used to combine conditional statements:

a=2

b=3

c=4

if a<b and b<c:

    print('both condition is true')

#The or keyword is a logical operator, and is used to combine conditional statements:

a=2

b=3

c=4

if a<b or b<c:

    print('only condition  is true')

#Nested If

#You can have if statements inside if statements, this is called nested if statements.

x=22

if x>10:

    print('x is big')

    if x<50:

        print('x is small')

        if x==22:

            print('both are equal')

else:

    print('i dont know')

#The pass Statement

a = 33

b = 200

if b > a:

  pass

# #.............................Python While Loops.....................

x=10

while x<=20:

    print(x)

    x=x+1

#The break Statement

x=10

while x<50:

    print(x)

    if x==30:

        break

    x=x+1

print(x)

#The continue Statement

i = 0

while i < 6:

  i += 1

  if i == 3:

    continue

  print(i)

#The else Statement

i = 1

while i < 6:

  print(i)

  i += 1

else:

  print("i is no longer less than 6")

# #.............................for loop....................

#Looping Through a String

#Even strings are iterable objects, they contain a sequence of characters:

x='lamyaa'

for s in x:

    print(s)

#The break Statement in for loop

x=[1,2,3,4,5,6,7,8,9]

for s in x:

    print(s)

    if s==6:

        break

#OR

fruits = ["apple", "banana", "cherry"]

for x in fruits:

  print(x)

  if x == "banana":

    break

#The continue Statement in for loop

#Do not print 6 AND banana:

x=[1,2,3,4,5,6,7,8,9]

for s in x:

    if s==6:

        continue

    print(s)

#OR

fruits = ["apple", "banana", "cherry"]

for x in fruits:

  if x == "banana":

    continue

  print(x)

#The range() Function

#To loop through a set of code a specified number of times, we can use the range() function,

#Note that range(6) is not the values of 0 to 6, but the values 0 to 5.

for x in range(10):

    print(x)

#range in starting and ending value

for x in range(5,10):

    print(x)

# it is possible to specify the increment value by adding a third parameter: range(2, 30, 3):

for x in range(2,10,3):

    print(x)

#Else in For Loop

for x in range(10):

    print(x)

else:

    print('end')

#Nested Loops

x=[1,2,3]

y=['apple','jam','kla']

for s in x:

    for t in y:

     print(s,t)

          #OR

adj = ["red", "big", "tasty"]

fruits = ["apple", "banana", "cherry"]

for x in adj:

  for y in fruits:

    print(x, y)

#The pass Statement

if x in range(5):

    pass

# #.............................Python Functions................

#Creating a Function

#In Python a function is defined using the def keyword:

def fun():

    print('my first')

#calling a funtion

def fun():

    print('my first')

fun()

#Arguments

#Information can be passed into functions as arguments.

#Arguments are specified after the function name, inside the parentheses.

def fun(name):

    print('my name is:',name)

fun('mahmud')

#You can add as many arguments as you want, just separate them with a comma.

def fun(name,roll):

    print('my name:',name)

    print('my roll:',roll)

fun('mahmud',120)

#Parameters or Arguments?

#A parameter is the variable listed inside the parentheses in the function definition.

#An argument is the value that are sent to the function when it is called.

#Number of Arguments

def my\_function(fname, lname):

  print(fname + " " + lname)

my\_function("Emil", "Refsnes")

#If you try to call the function with 1 or 3 arguments, you will get an error:

def my\_function(fname, lname):

  print(fname + " " + lname)

#my\_function("Emil")

#Arbitrary Arguments, \*args

def fun(\*args):

    print(args)

fun('m','n',1,2,3)

#Keyword Arguments

def my\_function(child3, child2, child1):

  print("The youngest child is " + child3)

my\_function(child1 = "Emil", child2 = "Tobias", child3 = "Linus")

#Arbitrary Keyword Arguments, \*\*kwargs

def my\_function(\*\*kid):

  print("His last name is " + kid["lname"])

my\_function(fname = "Tobias", lname = "Refsnes")

#Default Parameter Value

def my\_function(country = "Norway"):

  print("I am from " + country)

my\_function("Sweden")

my\_function("India")

my\_function()

my\_function("Brazil")

#Passing a List as an Argument

food = ["apple", "banana", "cherry"]

for x in food:

    print(x)

my\_function(food)

#Return Values

def my\_function(x):

  return 5 \* x

print(my\_function(3))

print(my\_function(5))

print(my\_function(9))

#The pass Statement

def myfunction():

    pass

# Python Lambda

#Syntax

#lambda arguments : expression

x = lambda a : a + 10

print(x(5))

#A lambda function that multiplies argument a with argument b and print the result:

x = lambda a, b : a \* b

print(x(5, 6))

#A lambda function that sums argument a, b, and c and print the result:

x = lambda a, b, c : a + b + c

print(x(5, 6, 2))

#Why Use Lambda Functions?

#The power of lambda is better shown when you use them as an anonymous function inside another function.

def myfunc(n):

  return lambda a : a \* n

#Use that function definition to make a function that always doubles the number you send in:

def myfunc(n):

  return lambda a : a \* n

mydoubler = myfunc(2)

print(mydoubler(11))

#use the same function definition to make a function that always triples the number you send in:

def myfunc(n):

  return lambda a : a \* n

mytripler = myfunc(3)

print(mytripler(11))

# use the same function definition to make both functions, in the same program:

def myfunc(n):

  return lambda a : a \* n

mydoubler = myfunc(2)

mytripler = myfunc(3)

print(mydoubler(11))

print(mytripler(11))

"""