num = int(7)

for i in range(1,num,+2):

print(' '\*(num-i),' $'\*i)

for i in range(num,0,-2):

print(' '\*(num-i),' $'\*i)

Pattern printing

num = int(4)

for i in range(1,num,+1):

print(' '\*(num-i),'$'\*i)

for i in range(num,0,-1):

print(' '\*(num-i),'$'\*i)

maximum\_number\_of\_stars = int(input("please give odd number of lines do you want :"))

middle = int(maximum\_number\_of\_stars/2)

middle2 = int(maximum\_number\_of\_stars/2)

char = "\*"

lgap = middle

def mod(a):

if a < 0 :

return(a\*(-1))

else:

return a

#print(" "\*middle,char)

for i in range(1,maximum\_number\_of\_stars+1):

if i == 1 or i == maximum\_number\_of\_stars :

print(" "\*middle2," \*"," "\*middle2)

lgap -= 1

else:

mgap = (((maximum\_number\_of\_stars - mod(lgap)) - 2)-mod(lgap))

print(" "\*mod(lgap),char," "\*mgap,char," "\*mod(lgap))

lgap -= 1

Printing 1 to 100 without loop

def PrintMessage (i):

n = i

if n > 100:

exit

else:

print (n)

n+=1

PrintMessage(n)

print ("This is Python program for printing 1 to 100 without loop...")

PrintMessage(1

1 to 100 prime numbers printing

def vishnu(x):

if x > 2 :

for i in range(2,x):

if (x%i)==0:

break

else:

print(x)

else:

print(x)

a = int(input('give a minimum number :'))

b = int(input('give a maximum number :'))

while a<=b:

vishnu(a)

a+=1

Salary calculator

def GetTotalDaysOfMonth (salary\_month) :# function for day calculation

pyear = int (salary\_month [1:4])

pmonth = int (salary\_month [5:6])

days\_in\_month = 0

if pmonth == 2 :

if pyear % 4 == 0 :

days\_in\_month = 29

else :

days\_in\_month = 28

elif pmonth==4 or pmonth==6 or pmonth==9 or pmonth==11 :

days\_in\_month = 30

else:

days\_in\_month = 31

print ('Total days of month ',days\_in\_month)

return days\_in\_month

def salary\_calculation(pds,pd,od):# function for salary calculation

final\_salary = round ((pd\*pds)+(od\*pds))

print('your salary is',final\_salary)

salary\_month = input('please input Salary Month - YYYYMM:')# code for value input

days\_in\_month = GetTotalDaysOfMonth (salary\_month)

present\_days = int(input('please input present days :'))

overtime\_days = int(input('please Enter over time day :'))

salary = int(input('please give your base salary per month :'))

per\_day\_salary = salary/days\_in\_month

# function calling for salary calculation

salary\_calculation(per\_day\_salary,present\_days,overtime\_days)

Guessing game(papa)

import random

def CompareNumbers (i,r):# variables comparision

if i == r:

print('Congratulations ! You have guessed right number ')

return 1

else:

print('Sorry ! You have guessed wrong number . Right number is ' + str (r))

return 0

j = 0# variables declaration

while j == 0:

i = int(input('Please guess a Number: '))

r = random.randint(0,10)

j = CompareNumbers (i,r)

Guessing game (my own )

import random

def PlayGame ():# declaring function

b = random.randint(0,10)

i = int(input('Please guess a Number: '))

if i == b:

print('Congratulations ! You have guessed right number ')

exit

else:

print('Sorry ! You have guessed wrong number . Right number is ' + str (b))

PlayGame() #recursion calling

PlayGame ()# calling function

Guessing game (v2)

import random

def PlayGame ():# declaring function

b = random.randint(0,10)

count = 1

while count <= 3 :

i = int(input('Please guess a Number: '))

if i == b :

print('congratulation your guessed number is correct!!!')

break

else:

print('your guessed number is wrong!!')

count+=1

if i != b :

print('correct number is',b)

PlayGame ()# calling function

Area calculator

def square\_area(s):

Area=s\*s

return Area

def rectangle\_area(l,b):

Area = l \* b

return Area

def triangle\_area(h,b):

Area = (1/2)\*b\*h

return Area

def trapezium\_area(a,b,h):

Area = (1/2)\*(a+b)\*h

return Area

def circle\_area(r):

Area = 22/7\*(r\*\*2)

return Area

def sector\_area(r,angle):

Area = angle/360\*(22/7\*(r\*\*2))

return round(Area)

def shapes\_area\_calculation(pinput):

if pinput == '1' :

side = int(input('please give length of side of square:'))

print('your area is = ',square\_area(side))

elif pinput=='2' :

length = int(input('please give length of the rectangle :'))

breadth = int(input('please give breadth of th rectangle :'))

print('your area is = ',rectangle\_area(length,breadth))

elif pinput=='3':

height = int(input('please give your height of triangle :'))

base = int(input('please give your base of the triangle :'))

print('your area is = ',triangle\_area(height,base))

elif pinput=='4' :

side1 = int(input('please give length of your parallel side of trapezium :'))

side2 = int(input('please give length of your second parallel of trapezium :'))

height= int(input('please give height of trapezium :'))

print('your area is = ',trapezium\_area(side1,side2,height))

elif pinput=='6' :

radius = int(input('please give your radius :'))

print('your area is = ',circle\_area(radius))

elif pinput=="7" :

radius = int(input('please give your radius of the circle :'))

angle = int(input('please give your angle of sector :'))

print('your area is = ',sector\_area(radius,angle))

else:

print("did not recognize input !!!")

def printing\_list\_in\_3d\_calculation():

count = 1

list\_of\_3d\_shapes\_calculation = ('TSA','CSA','VOLUME')

for i in list\_of\_3d\_shapes\_calculation:

print('('+str(count)+')',i)

count+=1

def calculation\_for\_cube(pinput):

side = int(input('give length of side of the cube :'))

if pinput=='1' :

area = 6\*side\*\*2

print('TSA of shape is = ',area)

elif pinput=='2' :

area = 4 \* side\*\*2

print('CSA of shape is = ',area)

elif pinput=='3' :

volume = side\*\*3

print('volume of shape is =',volume)

def calculation\_for\_cuboid(pinput):

length = int(input('give length of side of the cube :'))

height = int(input('give height of shape :'))

breadth = int(input('give breadth of shape :'))

if pinput=='1' :

area = 2\*((length\*breadth)+(breadth\*height)+(height\*length))

print('TSA of shape is = ',area)

elif pinput=='2' :

area = 2\*height\*(length+breadth)

print('CSA of shape is = ',area)

elif pinput=='3' :

volume = length\*breadth\*height

print('volume of shape is =',volume)

def calculation\_for\_cone(pinput):

height = int(input('give height of shape :'))

radius = int(input('give radius of shape :'))

length = ((height\*\*2)+(radius\*\*2))\*\*-2

if pinput=='1' or pinput=='tsa':

area = 22/7\*radius\*(length+radius)

print('TSA of shape is = ',area)

elif pinput=='2' or pinput=='csa':

area = 22/7\*radius\*length

print('CSA of shape is = ',area)

elif pinput=='3' or pinput=='volume':

volume = 1/3\*22/7\*(radius\*\*3)

print('volume of shape is =',volume)

def calculation\_for\_sphere(pinput):

radius = int(input('give radius of shape :'))

if pinput=='1' :

area = 4 \* 22/7 \* (radius\*\*2)

print('CSA of shape is = ',area)

elif pinput=='2' :

area = 4 \* 22/7 \* (radius\*\*2)

print('CSA of shape is = ',area)

elif pinput=='3' :

volume = 4/3 \*(22/7 \* (radius\*\*3))

print('volume of shape is =',volume)

def calculation\_for\_cylinder(pinput):

radius = int(input('give radius of shape :'))

height = int(input("please give height of the shape"))

if pinput=='1' :

area = 2 \* 22/7 \* radius \* (height + radius)

print('tSA of shape is = ',area)

elif pinput=='2' :

area = 2 \* 22/7 \* radius \* height

print('CSA of shape is = ',area)

elif pinput=='3' :

volume = 22/7 \* (radius\*\*3)

print('volume of shape is =',volume)

def calculation\_for\_hemisphere(pinput):

radius = int(input('give radius of shape :'))

if pinput=='1' :

area = 3 \* 22/7 \* (radius\*\*2)

print('tSA of shape is = ',area)

elif pinput=='2' :

area = 2 \* 22/7 \* (radius\*\*2)

print('CSA of shape is = ',area)

elif pinput=='3' :

volume = 2/3 \* (radius\*\*3)

print('volume of shape is =',volume)

def calculation\_for\_3d\_shapes(pinput):

if pinput == '1' :

printing\_list\_in\_3d\_calculation()

asking\_tsa\_csa\_volume = input('which calculation you want from above :')

calculation\_for\_cube(asking\_tsa\_csa\_volume)

elif pinput=='2' :

printing\_list\_in\_3d\_calculation()

asking\_tsa\_csa\_volume = input('which calculation you want from above :')

calculation\_for\_cuboid(asking\_tsa\_csa\_volume)

elif pinput=='3' :

printing\_list\_in\_3d\_calculation()

asking\_tsa\_csa\_volume = input('which calculation you want from above :')

calculation\_for\_cone(asking\_tsa\_csa\_volume)

elif pinput=='5' :

printing\_list\_in\_3d\_calculation()

asking\_tsa\_csa\_volume = input('which calculation you want from above :')

calculation\_for\_sphere(asking\_tsa\_csa\_volume)

elif pinput=='4' :

printing\_list\_in\_3d\_calculation()

asking\_tsa\_csa\_volume = input('which calculation you want from above :')

calculation\_for\_cylinder(asking\_tsa\_csa\_volume)

elif pinput=='6' :

printing\_list\_in\_3d\_calculation()

asking\_tsa\_csa\_volume = input('which calculation you want from above :')

calculation\_for\_hemisphere(asking\_tsa\_csa\_volume)

else:

print("did not recognize input !!!")

def printing\_\_2d\_shapes\_list():

count = 1

list\_of\_2d\_shapes = ('square','rectangle','triangle','trapezium','rohmbus','circle','sector')

for i in list\_of\_2d\_shapes :

print('('+str(count)+')',i)

count+=1

def printing\_\_3d\_shapes\_list():

count = 1

list\_of\_3d\_shapes = ('cube','cuboid','cone','cylinder','sphere','hemisphere')

for i in list\_of\_3d\_shapes :

print('('+str(count)+')',i)

count+=1

asking\_2d\_or\_3d = input('which type of shape do you want to calculate :')

if asking\_2d\_or\_3d == '2d':

printing\_\_2d\_shapes\_list()

my\_input = input("which shape's area you want to calculate from above :")

shapes\_area\_calculation(my\_input)

elif asking\_2d\_or\_3d=='3d':

printing\_\_3d\_shapes\_list()

my\_input = input('which shape do you want to calculate from above :')

calculation\_for\_3d\_shapes(my\_input)

Calculating no.of alphabets and digits in a string

string = 'nagpur-440010'

count\_num = 0

count\_aplhabet = 0

for i in string:

if str.isdigit(i):

count\_num+=1

elif str.isalpha(i):

count\_aplhabet+=1

else:

continue

print(count\_aplhabet)

print(count\_num)

Reversing number with loop

def reverse\_string(str):

str1 = "" # Declaring empty string to store the reversed string

for i in str:

str1 = i + str1

return str1 # It will return the reverse string to the caller function

str = input('please give a number :') # Given String

print("The original string is: ",str)

print("The reverse string is",reverse\_string(str)) # Function call

if str==reverse\_string(str):

print('original string and reverse string are equal ')

else:

print('original string and reverse string are not equal ')

Calculating n factorial

n\_term = int(input('please give your n term for getting n! factorial :'))

x = 1

for i in range(1,n\_term+1):

x = x\*i

print(x)

All pythagorean trilpates

limits = 20

m = 2

c = 0

while c <limits:

for n in range(1, m) :

a = m \* m - n \* n

b = 2 \* m \* n

c = m \* m + n \* n

if c > limits :

break

print(a, b, c)

m = m + 1

Reading excel files

import pandas as pd

pd.\_\_version\_\_

data = pd.read\_excel('C:\\Users\\Administrator\\Desktop\\NAITIK FILE ENTRY.xlsx', index\_col=0,header=0,usecols='A,C,D', sheet\_name='Sheet2', skiprows=[0,2],names=['name','no1','no2'])

print(data)

Fibonacci series

previous\_number = int(input('please give previous number:'))

present\_number = int(input('please give present number:'))

count = 0

print(previous\_number)

print(present\_number)

while count<=10:

next\_number = present\_number + previous\_number

print(next\_number)

my\_var = present\_number

present\_number = next\_number

previous\_number = my\_var

count+=1

Palindrome number

import pyttsx3 # install library by typing in terminal "pip install pyttsx3"

engine = pyttsx3.init('sapi5')

voices= engine.getProperty('voices')

engine.setProperty('voices', voices[0].id)

def speak(audio):

engine.say(audio)

engine.runAndWait()

def reverse\_string(str):

str1 = ""

for i in str:

str1 = i + str1

return int(str1)

minimum\_number = int(input('please give minimum number : ')) # Given String

maximum\_number = int(input('please give maximum number :'))

while minimum\_number <= maximum\_number:

if minimum\_number>10:

if minimum\_number==reverse\_string(str(minimum\_number)):

print(minimum\_number)

speak(minimum\_number)

minimum\_number+=1

else:

minimum\_number+=1

else:

minimum\_number+=1

Printing all alphabets in reverse

alphabets = ['A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z']

no\_aplhabets = 25

my\_str = ' '

while no\_aplhabets>=0:

my\_str = my\_str + ' '+str(alphabets[no\_aplhabets])

no\_aplhabets-=1

if no\_aplhabets==0:

print(my\_str)

import datetime

from datetime import date

def calculation\_for\_days(start\_date,end\_date):

s\_date = int(start\_date[:2])

s\_month = int(start\_date[3:5])

s\_year = int(start\_date[-4:])

p\_date = int(end\_date[:2])

p\_month = int(end\_date[4:5])

p\_year = int(end\_date[-4:])

if p\_date < s\_date :

p\_date = p\_date + 30

p\_month = p\_month - 1

elif p\_month < s\_month :

p\_month = p\_month + 12

p\_year = p\_year - 1

no\_years = p\_year - s\_year

no\_months = p\_month - s\_month

no\_days = p\_date - s\_date

total\_days = (no\_years\*365) + (no\_months\*30) + no\_days + 10

print(no\_years,'years')

print(no\_months,'months')

print(no\_days,'days')

print(total\_days,"total days")

s\_date\_month\_year = input('please give your starting date (dd-mm-yyyy):')

p\_date\_month\_year = input('please give your your end date (dd-mm-yyyy) :')

calculation\_for\_days(s\_date\_month\_year,p\_date\_month\_year)

Day teller zeller test

def verify\_date(date,month , year):

#print ("your date is ",date,month,year)

x = 0

if month == 1 or month == 3 or month == 5 or month == 7 or month == 8 or month == 10 or month == 12 :

if date > 31 :

return x

if month == 4 or month == 6 or month == 9 or month == 11 :

if date >30 :

return x

if year % 4 == 0 :

if month == 2 :

if date > 29 :

return x

if (year % 4) != 0 :

if month == 2 :

if date > 28 :

return x

def get\_day\_of\_week(key):

my\_var = str(key)

my\_get\_day = {

'0':'sunday',

'1':'monday',

'2':'tuseday',

'3':'wednesday',

'4':'thursday',

'5':'friday',

'6':'saturday'

}

day = my\_get\_day[my\_var]

#print(day)

return day

def get\_month(month):

my\_var = str(month)

my\_month\_key = {

"3":"1",

"4":"2",

"5":"3",

"6":"4",

"7":"5",

"8":"6",

"9":"7",

"10":"8",

"11":"9",

"12":"10",

"1":"11",

"2":"12"

}

key\_value\_month = int(my\_month\_key[my\_var])

return key\_value\_month

my\_date = input('please give your date (dd-mm-yyyy):')

k = int(my\_date[:2])

#print(k)

m = int(my\_date[3:5])

#print(m)

pyear = str(my\_date[-4:])

#print(pyear)

changed\_month = get\_month(m)

v\_year = int(pyear)

#print(changed\_month)

if m == 1 or m== 2 :

pyear = int(pyear) - 1

changed\_year\_s = str(pyear)

d = int(changed\_year\_s[-2:])

#print(d)

c = int(changed\_year\_s[:2])

#print(c)

verify\_\_d = verify\_date(k,m,v\_year)

#print(verify\_\_d)

if verify\_\_d == 0 :

print('your given date is wrong!!')

else:

f = int(k) + int(((13\*changed\_month)-1)/5) + d + int(d/4) + int(c/4)- int(2\*c)

day\_code = f % 7

#print(day\_code)

day = get\_day\_of\_week(day\_code)

print(day)

Day teller key value method

from datetime import datetime

def verify\_date(date,month , year):

print ("your date is ",date,month,year)

x = 0

if month == 1 or month == 3 or month == 5 or month == 7 or month == 8 or month == 10 or month == 12 :

if date > 31 :

print ("Date is > 31")

return x

if month == 4 or month == 6 or month == 9 or month == 11 :

if date >30 :

print ('Date is > 30')

return x

if year % 4 == 0 :

if month == 2 :

if date > 29 :

print ("Date is > 29")

return x

if (year % 4) != 0 :

if month == 2 :

if date > 28 :

print ("Date is > 28")

return x

def get\_key\_month(month):

if month == 2 or month== 3 or month == 11 :

key\_value\_month = 4

elif month == 1 or month== 10 :

key\_value\_month = 1

elif month == 4 or month == 7 :

key\_value\_month = 0

elif month == 5 :

key\_value\_month = 2

elif month == 6 :

key\_value\_month = 5

elif month == 8 :

key\_value\_month = 3

elif month == 9 or month == 12 :

key\_value\_month = 6

return key\_value\_month

def get\_key\_century(century):

if century == 17 :

key\_century = 4

elif century == 18 :

key\_century = 2

elif century == 19 :

key\_century = 0

elif century == 20 :

key\_century = 6

return key\_century

def get\_day\_of\_week(key):

if key == 0:

day = 'saturday'

elif key == 1:

day = 'sunday'

elif key == 2:

day = 'monday'

elif key == 3:

day = 'tuesday'

elif key == 4:

day = 'wednesday'

elif key == 5:

day = 'thursday'

elif key == 6:

day = 'friday'

return day

def reduce\_century(year):

retyear = year

if retyear >=2100 :

while retyear >= 2100 :

retyear-=400

elif retyear < 1700 :

while retyear < 1700:

retyear+=400

return retyear

my\_date = input('please give your date (dd-mm-yyyy):')

pdate = int(my\_date[:2])

pmonth = int(my\_date[3:5])

pyear = int(my\_date[-4:])

reduced\_year = str(reduce\_century(pyear))

print("Reduced year ",reduced\_year)

first\_digit\_year = int(reduced\_year[:2])

last\_digit\_year = int(reduced\_year[2:])

print ("first\_digit\_year ",first\_digit\_year)

print ("last\_digit\_year ",last\_digit\_year)

modulus\_first\_digit = first\_digit\_year%4

print ("modulus first digit ",modulus\_first\_digit)

modulus\_last\_digit = last\_digit\_year%4

print ("modulus last digit ",modulus\_last\_digit)

my\_var = last\_digit\_year - modulus\_last\_digit

print ("My Var ",my\_var)

day\_code = my\_var / 4

print ("Day code ",day\_code)

day\_code = day\_code + pdate

print ("Day code ",day\_code)

keyOfmonth = int(get\_key\_month(pmonth))

print ("key of month ",keyOfmonth)

day\_code = day\_code + keyOfmonth

print ("Day code ",day\_code)

if modulus\_last\_digit == 0 and (pmonth == 1 or pmonth == 2) :

day\_code-=1

if last\_digit\_year == 0 and modulus\_first\_digit != 0 and (pmonth == 1 or pmonth == 2) :

day\_code+=1

keyOfcentury = int(get\_key\_century(first\_digit\_year))

print ("key of century ",keyOfcentury)

day\_code = day\_code + keyOfcentury

day\_code = day\_code + last\_digit\_year

print ("Day code final ",day\_code)

day\_code = day\_code % 7

print ("Day code remainder ",day\_code)

day = get\_day\_of\_week(day\_code)

print("Day ",day)

Html

<html>

<style>

.active{

background-color: rgb(255, 213, 213);

height: 90%;

width: 10%;

float: left;

font-size: 80%;

text-align: center;

font-style: italic;

padding-top: 1%;

}

.notactive{

background-color: rgb(255, 228, 217);

height: 90%;

width: 10%;

float: left;

font-size: 80%;

text-align: center;

font-style: italic;

padding-top: 1%;

}

.notactive2{

background-color: rgb(255, 228, 217);

height: 90%;

width: 10%;

float: left;

font-size: 80%;

text-align: center;

font-style: italic;

padding-top: 1%;

}

.notactive\_lasttab{

background-color: rgb(255, 228, 217);

height: 90%;

width: 9%;

padding-right: 1%;

font-size: 100%;

float: left;

text-align: center;

font-style: italic;

padding-top: 1%;

}

.header1{

width: 80%;

height: 500%;

margin-left: 10%;

margin-right: 10%;

}

.header{

width: 80%;

margin-left: 10%;

margin-right: 10%;

}

.photo\_slide{

background-color: rgb(207, 202, 196);

width: 100%;

height: 60%;

}

.left-list{

float: left;

width: 25%;

height: 90%;

background-color: burlywood;

}

.preview-photo{

width: 75%;

height: 90%;

float: right;

background-color: floralwhite;

}

.topheader{

width: 100%;

height: 7%;background-color: darkgrey;

}

.previewpic{

float: left;

margin-top: 5%;

margin-left: 5%;

height: 40%;

width: 26%;

background-color: darkorange;

}

.logoheader{

width: 80%;

height: 10%;

background-color: darkorange;

margin-left: 10%;

margin-right: 10%;

}

.DIVIDER{

width: 100%;

height: 20%;

}

</style>

<body>

<div class="topheader"></div>

<div class="logoheader"></div>

<div class="header">

<div style="width: 100%; height: 5%;">

<div class="active">HOME</div>

<div class = "notactive"><center>SHOP</center> </div>

<div class = "notactive2"><center>BLOG</center></div>

<div class = "notactive2"><center>CONTACT</center> </div>

<div class = "notactive2"></div>

<div class = "notactive2"></div>

<div class = "notactive2"></div>

<div class = "notactive2"></div>

<div class = "notactive2"></div>

<div class = "notactive\_lasttab"></div>

</div>

<div class="photo\_slide">

<marquee> </marquee>

</div>

<div class="left-list"></div>

<div class="preview-photo">

<div class="previewpic"><img src="D:\Projects\my.jpg" height="100%" width="100%"></div>

<div class="previewpic"><img src="D:\Projects\my1.jpg" height="100%" width="100%"></div>

<div class="previewpic"><img src="D:\Projects\my2.jpg" height="100%" width="100%"></div>

<div class="previewpic"><img src="D:\Projects\my3.jpg" height="100%" width="100%"></div>

<div class="previewpic"><img src="D:\Projects\my4.jpg" height="100%" width="100%"></div>

<div class="previewpic"><img src="D:\Projects\my5.jpg" height="100%" width="100%"></div>

</div>

</div>

<div class="header1">

<div class="DIVIDER"></div>

</div>

</body>

</html>

Marks\_teller

my\_marks\_dictionary = {

"aakash" : {"maths":78,"english":71,"s.s":76,"science":80},

"vishnu" : {'maths':64,'english':70,'s.s':74,"science":71},

"tulsi" : {'maths':"Null",'english':"Null",'s.s':"out of syllabus",'science':"Null"}

}

name\_of\_student = input("please give name of student:")

subject = input("please give name of subject:")

for i in my\_marks\_dictionary.keys():

if i == name\_of\_student:

for n in my\_marks\_dictionary[i] :

if n == subject:

print("your marks in ",subject,"are",my\_marks\_dictionary[i][n])

Sign in page logic

username\_password = {

"srikanth":"srikanth",

"vishnu":"a\_123456",

"kartik":"2005",

"aakash":"Abc\_123456",

"chintu":"chintu\_anna",

"yamini":"yamini\_akka",

"tulsi":"tulsi\_akka",

"kaveri":"kaveri\_choti",

"bachi":"bachi\_chelle"

}

username = input("please give your username :")

password = input('please give your password :')

if username in username\_password.keys():

if password == username\_password[username]:

print('you are welcomed here in login page !!!!!')

else:

print('you password is wrong !!!!')

else:

print('username not found !!!!')

Student\_master

student\_master = {}

student\_master = {

"1":{"Name":"Vishnu","class":"11","section":"D","number":"7624033247"},

"2":{"Name":"Tulsi","class":"12","section":"A","number":"9510756465"},

"3":{"Name":"Aakash","class":"11","section":"C","number":"9825111444"},

"4":{"Name":"kaveri","class":"8","section":"D","number":"9687000888"}

}

list\_what\_to\_do = ["Add","Modify","Delete","List","Exit"]

loopx = 0

while loopx == 0 :

count=1

for i in list\_what\_to\_do:

print(count,":",i)

count+=1

asking\_what\_do\_you\_want\_to\_do = int(input("please give key code of what do you want to do from above :"))

if asking\_what\_do\_you\_want\_to\_do == 1:

student\_id = input("please give student id :")

if student\_id in student\_master.keys() :

print("your given id is in use !!")

continue

student\_name = input("please give student name :")

student\_class = input("please give student class :")

student\_section = input("please give student section :")

student\_number = input("please give student number :")

asking\_ensurement = input("are you sure to add the data (y/n) : ")

if asking\_ensurement == "n" :

print("your data has cancelled !!!")

continue

else:

student\_master[student\_id] = {"Name":student\_name,"class":student\_class,"section":student\_section,"number":student\_number}

print("your data has saved successfully!!")

continue

if asking\_what\_do\_you\_want\_to\_do == 2:

student\_id = input("please give student id :")

if student\_id not in student\_master.keys() :

print("there is no such data for given id!!")

continue

print("student id :",student\_id)

for z in student\_master[student\_id]:

print(z,":",student\_master[student\_id][z])

asking\_ensurement = input("are you sure to modify the data (y/n) : ")

if asking\_ensurement == "n" :

print("your data modification has cancelled !!")

continue

else:

student\_name = input("please give student name :")

student\_class = input("please give student class :")

student\_section = input("please give student section :")

student\_number = input("please give student number :")

student\_master[student\_id] = {"Name":student\_name,"class":student\_class,"section":student\_section,"number":student\_number}

print("your data has modified successfully!!")

continue

if asking\_what\_do\_you\_want\_to\_do == 3:

student\_id = input("please give student id :")

if student\_id not in student\_master.keys() :

print("there is no such data for given id!!")

continue

print("student id :",student\_id)

for z in student\_master[student\_id]:

print(z,":",student\_master[student\_id][z])

asking\_ensurement = input("are you sure to delete the data (y/n) : ")

if asking\_ensurement == "y" :

del(student\_master[student\_id])

print("your data has deleted !!")

continue

else:

print("your data deletion is cancelled !!!")

continue

if asking\_what\_do\_you\_want\_to\_do == 4:

student\_id = input("please give student id :")

if student\_id == "":

print('Name'," ","Class"," ","Section"," ","Number")

for j in student\_master.keys ():

print (student\_master[j]["Name"],' ',student\_master[j]["class"]," ",student\_master[j]["section"]," ",student\_master[j]["number"])

else:

if student\_id not in student\_master.keys() :

print("there is no such data for given id!!")

continue

print()

print('Name'," ","Class"," ","Section"," ","Number")

print (student\_master[student\_id]["Name"]," ",student\_master[student\_id]["class"]," ",student\_master[student\_id]["section"]," ",student\_master[student\_id]["number"])

continue

if asking\_what\_do\_you\_want\_to\_do == 5:

loopx+=1

continue

student\_master = []

loopx = 0

while loopx == 0 :

student\_id = int(input("give id :"))

name = input('give name :')

standard = input("give class :")

number = input("please give number :")

list\_data = [student\_id,name , standard , number]

length\_of\_list = len(list\_data) - 1

#print(length\_of\_list)

student\_master.append(list\_data)

#print(student\_master)

for x in student\_master :

for y in x :

print(y)

Student data management

#student\_master = {}

student\_master = [

["ID","NAME","CLASS","SECTION","NUMBER"]

]

list\_what\_to\_do = ["Add","Modify","Delete","List","Exit"]

loopx = 0

while loopx == 0 :

count=1

for i in list\_what\_to\_do:

print(count,":",i)

count+=1

asking\_what\_do\_you\_want\_to\_do = int(input("please give key code of what do you want to do from above :"))

if asking\_what\_do\_you\_want\_to\_do == 1:

student\_id = len(student\_master)

print("your ID is :", student\_id)

student\_name = input("please give student name :")

student\_class = input("please give student class :")

student\_section = input("please give student section :")

student\_number = input("please give student number :")

studnet\_data = [student\_id,student\_name,student\_class,student\_section,student\_number]

asking\_ensurement = input("are you sure to add the data (y/n) : ")

if asking\_ensurement == "n" :

print("your data addition has cancelled !!!")

continue

else:

student\_master.append(studnet\_data)

print("your data has saved successfully!!")

continue

if asking\_what\_do\_you\_want\_to\_do == 2:

student\_id = input("please give student id :")

if int(student\_id) >= len(student\_master):

print("your given id has no data !!")

continue

print(student\_master[0][0],student\_master[0][1],student\_master[0][2],student\_master[0][3],student\_master[0][4])

print(student\_master[int(student\_id)] [0],student\_master[int(student\_id)] [1],student\_master[int(student\_id)] [2],student\_master[int(student\_id)] [3],student\_master[int(student\_id)] [4])

asking\_ensurement = input("are you sure to modify the data (y/n) : ")

if asking\_ensurement == "n" :

print("your data modification has cancelled !!")

continue

else:

student\_name = input("please give student name :")

student\_class = input("please give student class :")

student\_section = input("please give student section :")

student\_number = input("please give student number :")

if len(student\_name) > 0 :

student\_master[int(student\_id)][1] = student\_name

if len(student\_class) > 0 :

student\_master[int(student\_id)][2] = student\_class

if len(student\_section) > 0 :

student\_master[int(student\_id)][3] = student\_section

if len(student\_number) > 0 :

student\_master[int(student\_id)][4] = student\_number

print("your data has modified successfully!!")

continue

if asking\_what\_do\_you\_want\_to\_do == 3:

student\_id = input("please give student id :")

if int(student\_id) >= len(student\_master):

print("your given id has no data !!")

continue

print(student\_master[0][0],student\_master[0][1],student\_master[0][2],student\_master[0][3],student\_master[0][4])

print(student\_master[int(student\_id)] [0],student\_master[int(student\_id)] [1],student\_master[int(student\_id)] [2],student\_master[int(student\_id)] [3],student\_master[int(student\_id)] [4])

asking\_ensurement = input("are you sure to delete the data (y/n) : ")

if asking\_ensurement == "y" :

student\_master[int(student\_id)][0] = student\_id

student\_master[int(student\_id)][1] = ""

student\_master[int(student\_id)][2] = ""

student\_master[int(student\_id)][3] = ""

student\_master[int(student\_id)][4] = ""

print("your data has deleted !!")

continue

else:

print("your data deletion is cancelled !!!")

continue

if asking\_what\_do\_you\_want\_to\_do == 4:

student\_id = input("please give student id :")

row1 = len(student\_master)

if student\_id == "":

for row in range(0,row1):

print(student\_master[row][0],student\_master[row] [1],student\_master[row] [2],student\_master[row] [3],student\_master[row] [4])

continue

else:

if int(student\_id) >= len(student\_master):

print("your given id has no data !!")

continue

print(student\_master[0][0],student\_master[0][1],student\_master[0][2],student\_master[0][3],student\_master[0][4])

print (student\_master[int(student\_id)][0]," ",student\_master[int(student\_id)][1]," ",student\_master[int(student\_id)][2]," ",student\_master[int(student\_id)][3]," ",student\_master[int(student\_id)][4])

continue

if asking\_what\_do\_you\_want\_to\_do == 5:

loopx+=1

continue

Student data management with 2d array

#student\_master = {}

student\_master = [

["ID","NAME","CLASS","SECTION","NUMBER"]

]

list\_what\_to\_do = ["Add","Modify","Delete","List","Exit"]

loopx = 0

while loopx == 0 :

count=1

for i in list\_what\_to\_do:

print(count,":",i)

count+=1

asking\_what\_do\_you\_want\_to\_do = input("please give key code of what do you want to do from above :")

if asking\_what\_do\_you\_want\_to\_do.isdigit()==False:

print("please give an valid code!!!")

continue

if int(asking\_what\_do\_you\_want\_to\_do) == 1:

student\_id = len(student\_master)

print("your ID is :", student\_id)

student\_name = input("please give student name :")

student\_class = input("please give student class :")

student\_section = input("please give student section :")

student\_number = input("please give student number :")

studnet\_data = [student\_id,student\_name,student\_class,student\_section,student\_number]

asking\_ensurement = input("are you sure to add the data (y/n) : ")

if asking\_ensurement == "n" :

print("your data addition has cancelled !!!")

continue

elif asking\_ensurement=="y":

student\_master.append(studnet\_data)

print("your data has saved successfully!!")

continue

else:

print("you have given an wrong code!!!")

print("your data addition has cancelled !!!")

continue

if int(asking\_what\_do\_you\_want\_to\_do) == 2:

student\_id = input("please give student id :")

if int(student\_id) >= len(student\_master):

print("your given id has no data !!")

continue

print(student\_master[0][0],student\_master[0][1],student\_master[0][2],student\_master[0][3],student\_master[0][4])

print(student\_master[int(student\_id)] [0],student\_master[int(student\_id)] [1],student\_master[int(student\_id)] [2],student\_master[int(student\_id)] [3],student\_master[int(student\_id)] [4])

asking\_ensurement = input("are you sure to modify the data (y/n) : ")

if asking\_ensurement == "n" :

print("your data modification has cancelled !!")

continue

else:

student\_name = input("please give student name :")

student\_class = input("please give student class :")

student\_section = input("please give student section :")

student\_number = input("please give student number :")

if len(student\_name) > 0 :

student\_master[int(student\_id)][1] = student\_name

if len(student\_class) > 0 :

student\_master[int(student\_id)][2] = student\_class

if len(student\_section) > 0 :

student\_master[int(student\_id)][3] = student\_section

if len(student\_number) > 0 :

student\_master[int(student\_id)][4] = student\_number

print("your data has modified successfully!!")

continue

if int(asking\_what\_do\_you\_want\_to\_do) == 3:

student\_id = input("please give student id :")

if int(student\_id) >= len(student\_master):

print("your given id has no data !!")

continue

print(student\_master[0][0],student\_master[0][1],student\_master[0][2],student\_master[0][3],student\_master[0][4])

print(student\_master[int(student\_id)] [0],student\_master[int(student\_id)] [1],student\_master[int(student\_id)] [2],student\_master[int(student\_id)] [3],student\_master[int(student\_id)] [4])

asking\_ensurement = input("are you sure to delete the data (y/n) : ")

if asking\_ensurement == "y" :

student\_master[int(student\_id)][0] = student\_id

student\_master[int(student\_id)][1] = ""

student\_master[int(student\_id)][2] = ""

student\_master[int(student\_id)][3] = ""

student\_master[int(student\_id)][4] = ""

print("your data has deleted !!")

continue

else:

print("your data deletion is cancelled !!!")

continue

if int(asking\_what\_do\_you\_want\_to\_do) == 4:

student\_id = input("please give student id :")

row1 = len(student\_master)

if student\_id == "":

for row in range(0,row1):

print(student\_master[row][0],student\_master[row] [1],student\_master[row] [2],student\_master[row] [3],student\_master[row] [4])

continue

else:

if int(student\_id) >= len(student\_master):

print("your given id has no data !!")

continue

print(student\_master[0][0],student\_master[0][1],student\_master[0][2],student\_master[0][3],student\_master[0][4])

print (student\_master[int(student\_id)][0]," ",student\_master[int(student\_id)][1]," ",student\_master[int(student\_id)][2]," ",student\_master[int(student\_id)][3]," ",student\_master[int(student\_id)][4])

continue

if int(asking\_what\_do\_you\_want\_to\_do) == 5 :

loopx+=1

continue

if int(asking\_what\_do\_you\_want\_to\_do) > 5 :

print("please give an valid code !!!")

continue

Sort (bubble sort )

list1 = [5,7,6,1,2,0]

for i in range(len(list1),0,-1):

for j in range(0,i-1) :

print(list1)

if list1[j] > list1[j+1]:

list1[j] , list1[j+1] = list1[j+1] ,list1[j]

print(list1)

Sort (selection sort )

#selection sort

import random

list1 = []

print ("Before sort")

for k in range (0,100) :

list1.append (random.randint (0,200))

print (list1)

print ("After sort")

for i in range(len(list1)) :

minvalue = i

for j in range(i+1,len(list1)):

if list1[minvalue] > list1[j] :

list1[minvalue] , list1[j] = list1[j] , list1[minvalue]

print (list1)

Sort (insertion sort)

list1 = [5,7,6,1,2,0]

for i in range(0,len(list1)):

for j in range(0,i):

if list1[i] < list1[j]:

list1[i],list1[j] = list1[j] , list1[i]

print(list1)

Sort (quick sort )

def quick\_sort (list1):

length = len(list1)

if length <= 1 :

return list1

else:

pivot = list1.pop()

lower\_list = []

higher\_list = []

for i in list1 :

if i < pivot :

lower\_list.append(i)

elif i >= pivot :

higher\_list.append(i)

return quick\_sort(lower\_list) + [pivot] + quick\_sort(higher\_list)

print(quick\_sort([5,6,1,2,9]))

Sort (merge sort)

list1 = [5,7,6,8,2,1,4]

def merge\_sort(list2):

if len(list2) > 1:

mid = len(list2)//2

left\_list = list2[:mid]

right\_list = list2[mid:]

merge\_sort(left\_list)

merge\_sort(right\_list)

i = 0

j = 0

k = 0

while i < len(left\_list) and j < len(right\_list):

if left\_list[i] < right\_list[j] :

list2[k] = left\_list[i]

i+=1

k+=1

else:

list2[k] = right\_list[j]

j+=1

k+=1

while i < len(left\_list):

list2[k] = left\_list[i]

i+=1

k+=1

while j < len(right\_list):

list2[k] = right\_list[j]

j+=1

k+=1

merge\_sort(list1)

print(list1)

sort(alphabets sort)

Ncert (exercise)

def findinng\_character(char,word):

count = 0

for index in word:

if index == char :

count+=1

return count

pword = input("please give your word :")

pcharacter = input("llease give a character to count :")

print(findinng\_character(pcharacter,pword))

def chan(word):

length = len(word)

vowels = ["a","i","e","o","u"]

for x in range(0,length):

if word[x] in vowels:

word = word.replace(word[x],"\_")

return word

pword = input("please give your word :")

print(chan(pword))

def counting(sentence):

no\_char = len(sentence)

no\_alphabets = no\_char

no\_digits = 0

no\_words = 0

no\_special\_symbol = 0

no\_space = 0

for index in sentence :

if index ==" ":

no\_space+=1

no\_words += 1

if int(index.isdigit()):

no\_digits += 1

no\_alphabets-=1

no\_alphabets -= no\_space

print("no of character",no\_char)

print("no of alphabets",no\_alphabets)

print("no of digit",no\_digits)

print("no of words",no\_words)

print("no of space",no\_space)

sentence = input("please give your sentence :")

counting(sentence)

def pisitalpha(pchar):

alphabets = ["q","w","e","r","t","y","u","i","o","p","a","s","d","f","g","h","j","k","l","z","x","c","v","b","n","m"

,"Q","W","E","R","T","Y","U","I","O","P","A","S","D","F","G","H","J","K","L","Z","X","C","V","B","N","M"]

if pchar in alphabets:

return True

else:

return False

def pisitnum(pchar):

numbers = ["1","2","3","4","5","6","7","8","9","0"]

if pchar in numbers :

return True

else:

return False

def pisitsymbol(pchar):

symbols = ["~","`","!","@","#","$","%","^","&","\*","(",")","\_","-","=","+","|","{","}","[","]",":",";",' " '," ' "," <",">","?",",",".","/"]

if pchar in symbols:

return True

else:

return False

def pisitspace (pchar):

if pchar == " ":

return True

else:

return False

my\_str = input("give a character :")

print("is aplha",pisitalpha(my\_str))

print("is num ",pisitnum(my\_str))

print("is symbol",pisitsymbol(my\_str))

print("is space",pisitspace(my\_str))

def cap(string2):

split\_str = string2.split()

for i in split\_str:

print(i.capitalize(),end="!")

string1 = input("give:")

cap(string1)

def cap(string2):

split\_str = string2.split()

for i in split\_str:

print(i.capitalize(),end=" ")

string1 = input("give:")

cap(string1)

def cal(string):

x = 0

for i in string :

if i.isdigit():

x = x + int(i)

return x

str1 = "aa\_123456789"

print(cal(str1))

def addition(list1,add):

for i in range(0,len(list1)):

list1[i] = list1[i] + add

return list1

list2 = [10,20,30.40,50,60,70,80,90]

print(list2)

add = int(input("what to add:"))

print(addition(list2,add))

def marks\_average(nstudents,marks):

sum\_of\_all\_n\_marks = 0

for i in marks:

sum\_of\_all\_n\_marks = sum\_of\_all\_n\_marks + i

average = sum\_of\_all\_n\_marks / nstudents

return average

no\_of\_students = int(input("plese give number of students :"))

count = no\_of\_students

marks = []

while count>=1:

marks\_of\_student = int(input("please give mark:"))

marks.append(marks\_of\_student)

count-=1

average\_marks = marks\_average(no\_of\_students,marks)

print(average\_marks)

minimum = int(input("please give minimum number :"))

maximum = int(input("please give maximum number:"))

num\_list = []

while minimum<=maximum:

num\_list.append(minimum)

minimum+=1

asking\_num = int(input("please give a number to check is it in range or not :"))

if asking\_num in num\_list:

print(int(num\_list.index(asking\_num))+1)

else:

print("your given number is not in range")

def no\_of\_time\_num(list1,num):

count = 0

for x in list1:

if num == x :

count+=1

return count

num\_list = [1,12,4,2,4,2,4,2,4,3,6,4,1,9,85,10,10,14,0,96,0,-1,-2,-45,-23,-22,-22,-4,-432,-1,-45455,-445,-1222,-41,-1,8,-445,-4454,-663,-96-9,-74-4,-4,-4]

print(no\_of\_time\_num(num\_list,4))

def no\_of\_time\_num(list1,negative,positive,what):

for x in list1:

if x < 0 :

negative.append(x)

elif x >= 0 :

positive.append(x)

if what == "p":

return positive

elif what == "n":

return negative

num\_list = [1,12,4,2,4,2,4,0,0,0,0,2,4,3,6,4,1,9,85,10,10,14,0,96,0,-1,-2,-45,-23,-22,-22,-4,-432,-1,-45455,-445,-1222,-41,-1,8,-445,-4454,-663,-96-9,-74-4,-4,-4]

negative = []

positive = []

print(no\_of\_time\_num(num\_list,negative,positive,"n"))

num\_list = [1,1,1,2,542,52,1,4,2,2,4,441,7452,442,122,6633,36333,333,333,33,33,33,33,33,3,33,33,33,33,33,333,3,11,11,11,11,1,10,410,4110,7850,10,10,10,10,10,10,10,10,10,10,1,1]

list1 = []

for i in num\_list:

if i not in list1:

list1.append(i)

print(list1)

import random

num\_list = []

for y in range(1,101):

num\_list.append(random.randint(1,5))

print(list(set(num\_list))[::-1])

import random

num\_list = []

for y in range(1,101):

num\_list.append(random.randint(1,78))

print(num\_list)

for z in range(0,2):

for i in num\_list:

for x in num\_list:

count\_i=num\_list.count(i)

if count\_i>1:

num\_list.remove(i)

print(num\_list)

def finding\_sum\_of\_n\_terms(n):

sum = 0

for x in range(1,n+1):

sum = sum + x

return sum

n\_terms = int(input("please n term :"))

print(finding\_sum\_of\_n\_terms(n\_terms))

def finding\_sum\_of\_n\_terms(n):

sum = 0

for x in range(1,n+1):

sum = sum + x

return sum

n\_terms = int(input("please n term :"))

print(finding\_sum\_of\_n\_terms(n\_terms))

def mixed\_fraction(numerator,denominator):

remainder = numerator % denominator

quotient = int(numerator/denominator)

print(str(quotient)+"("+str(remainder)+"/"+str(denominator)+")")

num = int(input("give numerator:"))

den = int(input("give denominator:"))

mixed\_fraction(num,den)

def exp(base,power):

result = 1

for i in range(0,power):

result = result \* base

return result

num = int(input("please give base number:"))

base = int (input("please give exponent number:"))

print(exp(num,base))

def factorial(x):

result = 1

while x >=1:

result = result \* x

x-=1

print(result)

x = int(input("give n number:"))

factorial(x)

def get\_pay(principle):

if principle >=500 and principle <1000 :

discount = 5

if principle >=1000 and principle <2000 :

discount = 8

if principle >=2000:

discount = 10

if principle<500:

discount = 0

final\_amount= principle - (discount/100\*principle)

return final\_amount

principle = int(input("please give total amount: "))

print("your payable amount is "+str(get\_pay(principle)))

from PIL import Image

def printing\_options(answers):

z = 97

for y in answers:

print(chr(z),":",y)

z+=1

def image\_question(list1,answers):

index = 0

an = 97

for x in list1:

x.show()

z = 97

printing\_options(answers)

asking = input("which animal is in the picture :")

if asking == chr(an):

print("your answer is write")

index+=1

an+=1

else:

print("your answer is wrong" +" write answer is :"+str(answers[index]))

index+=1

an+=1

#import random

img1 = Image.open("D:\\vsihnu\\lion.jpg")

img2 = Image.open("D:\\vsihnu\\tiger.jpg")

img3 = Image.open("D:\\vsihnu\\cat.jpg")

img4 = Image.open("D:\\vsihnu\\rat.jpg")

img5 = Image.open("D:\\vsihnu\\bat.jpg")

images = [img1,img2,img3,img4,img5]

answer\_images = ["lion","tiger","cat","rat","bat"]

image\_question(images,answer\_images)

string = input("please give a string:")

count = 0

my\_dict = {}

for i in string :

for x in string:

if i == x:

count+=1

my\_dict[i] = str(count)

count = 0

for z in my\_dict:

print(str(z)+":"+str(my\_dict[z]))

my\_dict = {

0:"Zero",

1:"One",

2:"Two",

3:"Three",

4:"Four",

5:"Five",

6:"Six",

7:"Seven",

8:"Eight",

9:"Nine"

}

my\_num = input("plese give a number")

str\_num =""

for i in my\_num:

str\_num = str\_num + " " + my\_dict[int(i)]

print(my\_num,":",str\_num)

my\_dict = (

"Zero",

"One",

"Two",

"Three",

"Four",

"Five",

"Six",

"Seven",

"Eight",

"Nine"

)

my\_num = input("plese give a number :")

str\_num =""

for i in my\_num:

str\_num = str\_num + " " + my\_dict[int(i)]

print(my\_num,":",str\_num)

email\_ids = ("vishnuchityalani@gmail.com","aakashchityala@gmail.com","siliconinfo@silicon.com")

for name in email\_ids:

x = name.split("@")

name = x[0]

domain = x[1]

print(name,":","\t",domain)

bank\_data\_management

#for getting bank account holder name use bank\_data[account\_number][0]

#for getting bank account holder amount use bank\_data[account\_number][2]

#for getting bank account holder interest\_rate use bank\_data[account\_number][3]

#for getting bank account holder matuirity\_date use bank\_data[account\_number][4]

#for getting bank account holder age use bank\_data[account\_number][1]

#for getting bank account holder account type use bank\_data[account\_number][5]

#for getting bank account holder password use bank\_data[account\_number][6]

import random

def get\_account\_number(data\_keys):

number = random.randint(1000,9999)

if number not in data\_keys:

return number

else:

get\_account\_number(data\_keys)

def interest\_rate\_generator(age,which\_type):

if which\_type == "s":

if age >=45 :

interest\_rate = 10

return interest\_rate

else:

if age <18 :

print("\n","you are not eligible for savings account !!!","\n")

return "e"

else :

interest\_rate = 7

return interest\_rate

elif which\_type == "p":

if age < 16 :

print("\n","you are not eligible for personnel account !!!","\n")

return "e"

else:

interest\_rate = 1

return interest\_rate

def print\_functions():

count = 1

list\_functions = ["open a bank account","Deposit money","Withdraw money","your details"]

for i in list\_functions:

print(count,"."," ",i)

count+=1

count = 1

def amount\_asking(which\_type):

if which\_type == "s":

amount = int(input("please give your amount : "))

return amount

if which\_type == "p":

amount = int(input("please give your amount : "))

return amount

def printing\_data(header,data,id):

output = ""

for item in header:

output = output + " " + item

print(output)

output2 = str(id)

for item1 in data[id]:

output2 += " " + str(item1)

print(output2)

headers = ["ID","Name","Age","Amount","Interest rate","matuirity time","account type","password"]

bank\_data = {

0:["Name","Age","Amount","Interest rate","matuirity time","account type","password"] , #my data!!!!!!!!!!!!!!!!

1000:["vishnu",19,0,2,0,"p",9353]

}

while True == True:

print\_functions()

asking\_service = int(input("please enter key code of service which do you want : "))

if asking\_service == 1 : # opening account

account\_number = int(get\_account\_number(bank\_data.keys()))

print("account\_number : ",account\_number)

which\_type = input("which type of account do you want to open savings or personal account(s/p) : ")

if which\_type == "canc":

print("your account creation has been cancelled !!")

continue

if which\_type != "s" and which\_type != "p":

print("you have given an invalid account type")

print("please give an valid account type")

which\_type = input("which type of account do you want to open savings or personal account(s/p) : ")

name = input("please give your name : ")

age = int(input("please give your age : "))

interest\_rate = interest\_rate\_generator(age,which\_type)

if interest\_rate == "canc":

print('account addition has been cancelled !! ')

continue

if interest\_rate != "e":

password = int(input("please generate your 4 digit password :"))

i = 0

while i == 0:

if len(str(password))<4 or len(str(password))>4 :

print("You have to give an valid password !!")

password = int(input("please generate your 4 digit password :"))

else :

i += 1

if which\_type == 'p':

amount = int(amount\_asking(which\_type))

maturity\_date = 0

elif which\_type == 's':

amount = int(amount\_asking(which\_type))

maturity\_date = int(input("please give your time period (years): "))

account\_data = [name , age , amount , interest\_rate , maturity\_date , which\_type , password]

bank\_data[account\_number] = account\_data

print("\n","Yor data has been added successfully!!")

else:

continue

if asking\_service== 2:

account\_id = int(input("please give your id : "))

fpassword = int(input("plese give your password : "))

if account\_id in bank\_data.keys():

if fpassword == bank\_data[account\_id][6] :

deposit = int(input("please give your deposit amount : "))

present\_amount = int(bank\_data[account\_id][2])

bank\_data[account\_id][2] = present\_amount + deposit

print(bank\_data[account\_id][2])

print("your amount is deposited successfully !! ")

else:

print("your given password is wrong!!!")

continue

else:

print("there is no such account !!")

continue

if asking\_service== 3:

account\_id = int(input("please give your id : "))

fpassword = int(input("plese give your password : "))

if account\_id in bank\_data.keys():

if fpassword == bank\_data[account\_id][6] :

withdrawl = int(input("please give your withdrawl amount : "))

present\_amount = int(bank\_data[account\_id][2])

wd = present\_amount - withdrawl

if wd <0:

print('not sufficient balance !!')

continue

bank\_data[account\_id][2] = wd

print(bank\_data[account\_id][2])

print("your amount is withdrawed successfully !! ")

else:

print("your given password is wrong!!!")

continue

else:

print("there is no such account !!")

continue

if asking\_service == 4 :

l\_id = int(input("please give your id :"))

if l\_id not in bank\_data.keys():

print("you have given an invalid id")

continue

l\_password = int(input("please give your password :"))

if l\_password != bank\_data[l\_id][6]:

print("you have given wrong password")

continue

gap\_name = len(bank\_data[l\_id][0]) - 4

gap\_age = len(" AGE") - 2

ide = "ID"

print("%s %s %s %s %s %s %s "%(ide,bank\_data[0][0],bank\_data[0][1],bank\_data[0][2],bank\_data[0][3],bank\_data[0][4],bank\_data[0][5]))

print("%d %s %d %d %d %d %s "%(l\_id,bank\_data[l\_id][0],int(bank\_data[l\_id][1]),int(bank\_data[l\_id][2]),int(bank\_data[l\_id][3]),int(bank\_data[l\_id][4]),bank\_data[l\_id][5]))

**def selection\_Sort(list1):**

**for i in range(len(list1)) :**

**minvalue = ord(list[i][0])**

**for j in range(i+1,len(list1)):**

**if minvalue > ord(list1[j][0]) :**

**list1[minvalue] , list1[j] = list1[j] , list1[minvalue]**

**return list1**

**def place\_finder(list,percentile):**

**place = (percentile/100)\*len(list)**

**marks = list[place]**

**print(place , marks )**

**import random**

**list1 = []**

**list2 = []**

**for i in range(6):**

**name = input("please give a name : ")**

**list1.append(name)**

**selection\_Sort(list1)**

**import random**

**def alphabets\_sorting(list):**

**for i in range(0,len(list)):**

**for z in range(i+1,len(list)):**

**if ord(list[i][0]) > ord(list[z][0]):**

**list[i] , list[z] = list[z] , list[i]**

**print(list)**

**list1 = []**

**i = 90**

**while i >=65:**

**list1.append(chr(i))**

**i-=1**

**print(list1)**

**alphabets\_sorting(list1)**

**Playing\_songs**

**from playsound import playsound**

**import time**

**functions\_list = ["playsound('mmusic1.mp3')","playsound('mmusic2.mp3')","playsound('mmusic3.mp3')"]**

**i = 0**

**ind = 0**

**while i == 0 :**

**if ind > len(functions\_list)-1:**

**i+=1**

**ind = 0**

**else:**

**print("playing songs!!!")**

**exec(functions\_list[ind])**

**ask\_want\_to\_stop = input("do you want to stop playing song!!!")**

**if ask\_want\_to\_stop == "y":**

**i+=1**

**ind=0**

**print("i am stoping playing songs")**

**continue**

**time.sleep(2)**

**ind+=1**

**Playing\_song\_by\_code**

**from playsound import playsound**

**import time**

**functions\_list = {"na kanulu apedu":"playsound('mmusic1.mp3')","shy emiway":"playsound('mmusic2.mp3')","kola kalle ilaa":"playsound('mmusic3.mp3')"}**

**count = 1**

**list1 = []**

**for z in functions\_list.keys():**

**list1.append(z)**

**for i in list1:**

**print("(",count,")",i)**

**count +=1**

**ask\_code = int(input("plese give your code of song :"))**

**song\_ind = list1[ask\_code-1]**

**song = functions\_list[song\_ind]**

**print("playing song "+song\_ind)**

**exec(song)**

**import webbrowser**

**from googlesearch import search**

**query = input("what do you want to search ????? ")**

**query\_answer\_list = search(query, num\_results=100)**

**query\_num = int(input("give how many number of links do you want ???? "))**

**if query\_num > len(query\_answer\_list):**

**print("\n""we can provide only ",len(query\_answer\_list)," links !!!")**

**z = 0**

**while z < len(query\_answer\_list):**

**print(query\_answer\_list[z])**

**webbrowser.open(query\_answer\_list[z])**

**z+=1**

**else:**

**z = 0**

**while z < query\_num:**

**print(query\_answer\_list[z])**

**webbrowser.open(query\_answer\_list[z])**

**z+=1**

**PHOTOS SORTING**

**import face\_recognition**

**import os**

**from face\_recognition.api import face\_distance, face\_encodings**

**import numpy as np**

**import shutil**

**tulsi\_image = face\_recognition.load\_image\_file(r"D:\PYTHON\JARVIS\photos\_sorting\089.jpg")**

**tulsi\_encode = face\_recognition.face\_encodings(tulsi\_image)**

**#print(list(face\_encode))**

**vishnu\_image = face\_recognition.load\_image\_file(r"D:\PYTHON\JARVIS\photos\_sorting\\_DSC1520.JPG")**

**vishnu\_encode = face\_recognition.face\_encodings(vishnu\_image)**

**kartik\_image = face\_recognition.load\_image\_file(r"D:\PYTHON\JARVIS\photos\_sorting\105.jpg")**

**kartik\_encode = face\_recognition.face\_encodings(kartik\_image)**

**vian\_image = face\_recognition.load\_image\_file(r"D:\PYTHON\JARVIS\photos\_sorting\035.jpg")**

**vian\_encode = face\_recognition.face\_encodings(vian\_image)**

**chachu\_image= face\_recognition.load\_image\_file(r"D:\PYTHON\JARVIS\photos\_sorting\103.jpg")**

**chachu\_encode = face\_recognition.face\_encodings(chachu\_image)**

**papa\_image = face\_recognition.load\_image\_file(r"D:\PYTHON\JARVIS\photos\_sorting\\_DSC1482 copy.jpg")**

**papa\_encode = face\_recognition.face\_encodings(papa\_image)**

**data = [**

**vishnu\_encode,**

**tulsi\_encode,**

**kartik\_encode,**

**vian\_encode,**

**chachu\_encode**

**]**

**face\_names = ["vishnu","tulsi","kartik","vian","chachu"]**

**photos\_dir = os.listdir(r"D:\PYTHON\JARVIS\photos\_sorting\unsorted\_photos")**

**#os.startfile("D:\\JARVIS\\photos\_sorting\\unsorted\_photos" + "\\" + str(os.listdir("unsorted\_photos")[0]))**

**Path = "D:\\PYTHON\\JARVIS\\photos\_sorting\\unsorted\_photos\\"**

**for i in photos\_dir:**

**uknownimage = face\_recognition.load\_image\_file(Path+i)**

**print(i)**

**uknown\_locations = face\_recognition.face\_locations(uknownimage)**

**unknown\_encodings = face\_recognition.face\_encodings(uknownimage,uknown\_locations)**

**for y in unknown\_encodings:**

**matches = face\_recognition.compare\_faces(data,y)**

**if matches[0][0]==True:**

**print('found',i)**

**File finder**

**import os**

**import shutil**

**path = "D:\\vishnu\_test"**

**directory = os.listdir(path)**

**file\_type = ".pdf"**

**destination = "D:\\output"**

**def finding\_file(path,file\_type,destination):**

**dirc = os.listdir(path)**

**print("scanning :",path)**

**for i in dirc :**

**if i.endswith(file\_type):**

**print("copying :",i,"----------------------------------------------------------------------")**

**fdir = path + "\\"+i**

**shutil.copy2(fdir,destination)**

**elif i[-4] != ".":**

**fdir = path + "\\"+i**

**print("scanning",fdir)**

**try :**

**finding\_file(fdir,file\_type,destination)**

**except:**

**fdir = path + "\\"+i**

**finding\_file(path,file\_type,destination)**