**Problem 1: Dice Rolling Create a function named rollDice that simulates the rolling of five dice and returns "Yahoo" if all five dice are the same. Otherwise it should return "Try again". Write a main function to demonstrate the function.**

#pythoncode:

import random

def rollDice():

for i in range(1):

a=random.randint(1,6)

for i in range(1):

b=random.randint(1,6)

if (a==b):

for i in range(1):

c=random.randint(1,6)

if(c==a):

for i in range(1):

d=random.randint(1,6)

if(d==a):

for i in range(1):

e=random.randint(1,6)

if(e==a):

for i in range(1):

f=random.randint(1,6)

if(f==a):

print("Yahoo")

else:

print("Try again")

rollDice()

#output: Try again #output:Yahoo

**#method2:**

import random

def rollDice():

a=[]

for i in range(5):

a.append(random.randint(1,6))

print(a)

if(a.count(a[0])==len(a)):

return "yahoo"

else:

return "Try again"

print(rollDice())

#output:

[1,6,5,4,3,2]

Try again

#output:

[1,1,,1,1,1]

#Yahoo

**Problem 2: Random Password Create a function named randomPassword that generates a random password. The password should have a random length between 7 and 10 characters. Each character should be randomly selected from positions 33 to 126 in the ASCII table. Your function will not take any parameters. It will return the randomly generated password as its only result. Write a main program that calls your function and displays the randomly generated password.**

#pythoncode:

import random

import string

def randomPassword():

lower\_len=int(input())

upper\_len=int(input())

digit\_len=int(input())

symbol\_len=int(input())

pwd\_len=lower\_len+upper\_len+digit\_len+symbol\_len

if (7<=pwd\_len<=10):

lower=string.ascii\_lowercase

upper=string.ascii\_uppercase

digit=string.digits

symbol=string.punctuation

str=random.choices(lower,k=lower\_len)+random.choices(upper,k=upper\_len)+random.choices(digit,k=digit\_len)+random.choices(symbol,k=symbol\_len)

random.shuffle(str)

password=''.join(str)

return password

else:

print("Invalid password")

print(randomPassword())

#output1:

2

2

3

3

cE3{`.57cD

#output2:

1

2

3

4

9'^9I"kF\_8

**problem 3: Random License Plate In a particular jurisdiction, older license plates consist of three letters followed by three digits. When all of the license plates following that pattern had been used, the format was changed to four digits followed by three letters. Create a function named randomLicencePlate that generates a random license plate. Your function should have approximately equal odds of generating a sequence of characters for an old or new license plate. Write a main program that calls your function and displays the randomly generated license plate.**

#pythoncode:

def randomLicencePlate():

import random

import string

char\_len = 3

digit1\_len = 4

digit2\_len = 3

char=string.ascii\_lowercase+string.ascii\_uppercase

digit=string.digits

str1=random.choices(char,k=char\_len)+random.choices(digit,k=digit2\_len)

str2=random.choices(digit,k=digit1\_len)+random.choices(char,k=char\_len)

random.shuffle(str1)

password1=''.join(str1)

random.shuffle(str2)

password2=''.join(str2)

alphabets=[]

digits=[]

for ch in password1:

if ch.isalpha():

alphabets.append(ch)

else:

digits.append(ch)

password1=''.join(sorted(alphabets)+sorted(digits))

alphabets1=[]

digits1=[]

for ch in password2:

if ch.isalpha():

alphabets1.append(ch)

else:

digits1.append(ch)

password2=''.join(sorted(digits1)+sorted(alphabets1))

print(password1,password2)

if \_\_name\_\_=="\_\_main\_\_":

randomLicencePlate()

#output1:

BKr478 0236MVr

#output2:

BKr478 0236MVr

**Problem 4: Rock Paper Scissors Rock paper scissors is a hand game for two or more players. Participants say “rock, paper, scissors” and then simultaneously form their hands into the shape of a rock (a fist), a piece of paper (palm facing downward), or a pair of scissors (two fingers extended). The rules are straightforward: • Rock smashes scissors • Paper covers rock • Scissors cut paper • If both players say same thing, then it is a draw Let us simulate the game between a player and computer as an opponent. Create a function named getUserMove which reads user move from command line and returns that move. Create a function named getComputerMove which returns the random move of computer. Create a function named getResult that takes both user and computer moves as input, then returns 1, 0 or -1 for win, loss and draw respectively. Write a main function that simulates the game for about 10 times and displays the number of wins, losses and draws of a player.**

**#pythoncode:**

import random

def getResult():

wins=0

draw=0

loss=0

def getUserMove():

for i in range(10):

user\_action = input("Enter a choice (rock, paper, scissors): ")

return user\_action

def getComputerMove():

possible\_actions = ["rock", "paper", "scissors"]

computer\_action = random.choice(possible\_actions)

return computer\_action

a=getUserMove()

b=getComputerMove()

if a == b:

return 0

elif a== "rock":

if b == "scissors":

return 1

else:

return -1

elif a == "paper":

if b== "rock":

return 1

else:

return -1

elif a == "scissors":

if b == "paper":

return 1

else:

return -1

if result==0:

draw+=1

elif result==-1:

loss+=1

elif result==1:

wins+=1

print(wins)

print(loss)

print(draw)

result=print(getResult())

**Problem 5: Coin Flip In this program, we are interested to explore the following: What’s the minimum number of times you have to flip a coin before you can have two consecutive flips that result in the same outcome (either both are heads or both are tails)? What’s the maximum number of flips that might be needed? How many flips are needed on average? Create a function named getNumOfFlips that returns the number of flips required to get two consecutive heads or two consecutive tails. Write a main program that simulates the process for 10 times and displays minimum, maximum and average number of flips required to get the desired outcome**

#pytimport random

def getnumofflips():

flip=[]

for i in range(10):

a=random.randint(0,1)

flip.append(a)

print(flip)

sum=0

for k in range(len(flip)-1):

if flip[k]==flip[k+1]:

sum+=1

print(sum)

print(sum/2)

print(sum/2 -1)

print(sum/2 +1)

getnumofflips()

#output:

[1,0,0,0,1,1,1,0,0,0]

6

3

4

2