NAUT B05 Assessment Scenario-5

NAME: REUBEN THOMAS
PS NO: 10685230
BATCH: 5

Note: Paste the final screenshot and code for every question once completed

Lab time: 8hrs

Python assessment

1. Take minimum 5 Inputs from User: E.g.("apple",15), ("banana",30) & store it in a tuple.

   a) Sort the input based upon the 2nd Item & print it, then make a pair for prime numbers at the End & print the same.
   b) E.g.("apple",15), ("banana",30), ("watermelon",5)
   c) Sample Output: [("apple",15,"banana",30,"watermelon"), (5)]

2. Take max 5 inputs from user & check whether the frequencies can become same.

Input E.g.: xxyyyzzz

3. Fetch the character which occurs the least amount of time in a String.

4. Take input from User & eliminate words from the string whose length is less than or equal to 4.

5. Perform recursive deletion on a string to make it empty. String & substring should be taken from the user as an input.

6. Create a script to extend Password expiry date for a User.

   a. Request can only be raised by a manager.
   b. Provide two options for Password expiry:1. Password Never Expires 2. Temporary Password.
   c. Under Temporary password provide options such as: 1week,1month,6months
   d. Display the Updated Date in the output.
   e. Take 3 inputs from the User in a form of Key Value Pair. (Key being Manager & Value being Reportees).
   f. Reportees can be more than 1 under same manager. (Assume current date as expiry Date)

7. Using a decorator function check if input is containing upper case, special characters, integers in different functions.

8. Create a User & set the password of the user based upon complexity. Categorize the password as Complex or simple based upon Complexity.

a. Number of passwords remembered will be Five. Password should contain min 1 capital letter,1 number & 1 special character. (Password length should be minimum 8 Char)

9. Add or Remove a user based upon Date. Take inputs for Date as well as Username from End User. Task should not be performed in case of Past Date.

10. Your assignment: Create a Tic Tac Toe game. You are free to use any IDE you like.

Here are the requirements:

- 2 players should be able to play the game (both sitting at the same computer)
- The board should be printed out every time a player makes a move
- You should be able to accept input of the player position and then place a symbol on the board

Solutions:

Q.1 Take minimum 5 Inputs from User: E.g.("apple",15), ("banana",30) & store it in a tuple. Sort the input based upon the 2nd Item & print it, then make a pair for prime numbers at the End & print the same. E.g.("apple",15), ("banana",30), ("watermelon",5) Sample Output: [("apple",15,"banana",30,"watermelon"), (5)]

```
mainlist = []
n = int(input("Enter the number of entries :"))
for i in range(1,n+1):
    st = input("\nEnter the text value of the item : ")
    num = int(input("Enter the numeric value of the item : "))
    l = [st,num]
    t = tuple(l)
    mainlist.append(t)
```

```
maintuple = tuple(mainlist)
print("Original : ",maintuple)
sortedtuple = sorted(maintuple,key=lambda x: x[1])
print("Sorted : ",sortedtuple)

primelist = []
for x in sortedtuple:
    n = x[1]
    f=0
    for i in range(2,int(n/2)+1):
        if n%i==0:
            f=1
            break;
    if f==0:
        primelist.append(n)
primetuple=tuple(primelist)

a = list(sum(sortedtuple, ()))
for x in a:
    if x in primelist:
        a.remove(x)

finala = tuple(a)

finallist = []
finallist.append(finala)
finallist.append(primetuple)
print(finallist)
```

Output:

```
Enter the number of entries : 6

Enter the text value of the item :  Reuben
Enter the numeric value of the item :  5

Enter the text value of the item :  Thomas
Enter the numeric value of the item :  14

Enter the text value of the item :  Automation
Enter the numeric value of the item :  4

Enter the text value of the item :  Cat
Enter the numeric value of the item :  13

Enter the text value of the item :  Dog
Enter the numeric value of the item :  17

Enter the text value of the item :  Animal
Enter the numeric value of the item :  6
```

```
Original :  (('Reuben', 5), ('Thomas', 14), ('Automation', 4), ('Cat',
13), ('Dog', 17), ('Animal', 6))
Sorted :  [('Automation', 4), ('Reuben', 5), ('Animal', 6), ('Cat', 13),
('Thomas', 14), ('Dog', 17)]
```

```
[('Automation', 4, 'Reuben', 'Animal', 6, 'Cat', 'Thomas', 14, 'Dog'), (5,
13, 17)]
```

Q.2 Take max 5 inputs from user & check whether the frequencies can become same.

Input E.g.: xxyyyzzz

```
def countfreq(st):
    freq = {}
    countlist = []
    for items in st:
        freq[items] = st.count(items)
```

```
    print(freq)
    countlist = freq.values()
    return list(countlist)


def checklist(c):
    i = c[0]
    chk=0
    for x in c:
        if x != i:
            chk=0
            return chk
    chk=1
    return chk

st = input("Enter the string : ")
countfreq(st)

for i in range(5):
    ch = input("\nEnter the character : ")
    st=st+ch
    c=countfreq(st)
    val=checklist(c)
    if val==1:
        print("The frequencies of the characters - Match")
        break
    else:
        print("The frequencies of the characters Do Not Match")
```

Output:

```
Enter the string :  xxyyyzzz
{'x': 2, 'y': 3, 'z': 3}

Enter the character :  z
{'x': 2, 'y': 3, 'z': 4}
The frequencies of the characters Do Not Match

Enter the character :  y
{'x': 2, 'y': 4, 'z': 4}
The frequencies of the characters Do Not Match

Enter the character :  x
{'x': 3, 'y': 4, 'z': 4}
The frequencies of the characters Do Not Match

Enter the character :  x
{'x': 4, 'y': 4, 'z': 4}
The frequencies of the characters - Match
```

Q.3 Fetch the character which occurs the least amount of time in a String.

```
def countfreq(st):
    freq = {}
    countlist = []
    for items in st:
        freq[items] = st.count(items)
    return freq
st = input("Enter the string : ")
f = {}
f=countfreq(st)
count=f.values()
countlist = list(count)
mvalue=min(countlist)
for key,value in f.items():
    if value == mvalue:
        mchar=key
```

```
print(mchar+" occurs the least number of times ie : "+str(mvalue))
```

Output:

```
Enter the string :  aaaaaaaaabbbbbccccccssssddddd
s occurs the least number of times ie : 4
```

Q. 4 Take input from User & eliminate words from the string whose length is less than or equal to 4.

```
word = []
sent = input("Enter your sentence : ")
word = sent.split(" ")
newlist=[]
for x in word:
    c=len(x)
    if c > 4:
        newlist.append(x)
print("The input string is :")
print(sent)
print("\nThe trimmed string is :")
newst=""
for x in newlist:
    newst=newst+x+" "
print(newst)
```

Output:

```
Enter your sentence :  Hello my name is Reuben
The input string is :
Hello my name is Reuben

The trimmed string is :
Hello Reuben
```

Q.5 Perform recursive deletion on a string to make it empty. String & substring should be taken from the user as an input.

```
st = input("Enter the string: ")

subst = input("Enter the sub string: ")

sublen = len(subst)


while(st!=''):
   loc=st.find(subst)
   if loc == -1:
      print("\nRecursive deletion is not possible. String after deletion is "+st)
      break
   else:
      st=st[:loc]+st[(loc+sublen):]
         #print(st)


if(len(st)==0):
   print("\nRecursive deletion is possible. String after deletion is empty")
```

Output:

```
Enter the string:  benbebenbenn
Enter the sub string:  ben

Recursive deletion is possible. String after deletion is empty
```

String in which recursive deletion is not possible

```
Enter the string:  bennbenbenbennnbben
Enter the sub string:  ben

Recursive deletion is not possible. String after deletion is nnnb
```

Q.6 Create a script to extend Password expiry date for a User.

a. **Request can only be raised by a manager.**
b. **Provide two options for Password expiry:1. Password Never Expires 2. Temporary Password.**
c. **Under Temporary password provide options such as: 1week,1month,6months**
d. **Display the Updated Date in the output.**
e. **Take 3 inputs from the User in a form of Key Value Pair. (Key being Manager & Value being Reportees).**
f. **Reportees can be more than 1 under same manager. (Assume current date as expiry Date)**

```
import datetime
usermapping = {"Reuben":["Alice","Bob","Thomas","Rob"]}
userpasswd = {}
managers = usermapping.keys()
sch = int(input("Enter your choice:\n1.Extend Password Expiry\n2.Exit the script.\n"))
finallist = []
while sch!=2:
    mname = input("Manager Verification\nEnter your name: ")
    if mname in managers:
        print("{} successfully verified!".format(mname))
        uname=input("\nEnter the name of the user you want to update : ")
        if uname in usermapping[mname]:
            print("You can change the password for this user, Please provide your choice:")
            ch =  int(input("\n1.Password Never Expires\n2.Temporary Password\n"))
            utuple = {}
            if ch == 2:
                currenttime = datetime.date.today()
                extendedtime = datetime.date.today()
                print(currenttime)
                tch = int(input("\nExtend duration:\n1. 1 Week\n2. 1 Month\n3. 6 Months\n"))
                if tch == 1:
                    addtime = datetime.timedelta(days=7)
```

```
                extendedtime = currenttime + addtime
                print("Password of {} Extended till : {}".format(uname,extendedtime))
            elif tch == 2:
                addtime = datetime.timedelta(days=30)
                extendedtime = currenttime + addtime
                print("Password of {} Extended till : {}".format(uname,extendedtime))
            elif tch ==3:
                addtime = datetime.timedelta(days=180)
                extendedtime = currenttime + addtime
                print("Password of {} Extended till : {}".format(uname,extendedtime))
            else:
                pass
            utuple.update({uname:extendedtime.isoformat()})
            finallist.append(utuple)

        elif ch == 1:
            print("Password of {} has no Expiry".format(uname))
            utuple.update({uname:"Never Expires"})
            finallist.append(utuple)

    else:
        print("\nUser is not mapped under you, cannot change password for
{}".format(uname))
else:
    print("\nManager cannot be verified, Please try again!")

sch = int(input("\nEnter your choice:\n1.Extend Password Expiry\n2.Exit the script.\n"))
print("\n\nUsers updated and their updated password expiry dates:\n{}".format(finallist))
```

Output:

Updating user's password expiry date:

Selecting the never expires option

```
Enter your choice:
1.Extend Password Expiry
2.Exit the script.
 1
Manager Verification
Enter your name:  Reuben
Reuben successfully verified!

Enter the name of the user you want to update :  Alice
You can change the password for this user, Please provide your choice:

1.Password Never Expires
2.Temporary Password
 1
Password of Alice has no Expiry

Enter your choice:
1.Extend Password Expiry
2.Exit the script.
 1
Manager Verification
Enter your name:  Reuben
Reuben successfully verified!
```

Selecting the temporary option

```
Enter the name of the user you want to update :  Bob
You can change the password for this user, Please provide your choice:

1.Password Never Expires
2.Temporary Password
 2
2021-12-27

Extend duration:
1. 1 Week
2. 1 Month
3. 6 Months
 1
Password of Bob Extended till : 2022-01-03

Enter your choice:
1.Extend Password Expiry
2.Exit the script.
 1
Manager Verification
Enter your name:  Reuben
Reuben successfully verified!

Enter the name of the user you want to update :  Thomas
You can change the password for this user, Please provide your choice:

1.Password Never Expires
2.Temporary Password
 2
2021-12-27

Extend duration:
1. 1 Week
2. 1 Month
3. 6 Months
 3
Password of Thomas Extended till : 2022-06-25
```

Exiting the script, we get the details of all the updated users and the new extended expiry dates.

```
Enter your choice:
1.Extend Password Expiry
2.Exit the script.
 2


Users updated and their updated password expiry dates:
[{'Alice': 'Never Expires'}, {'Bob': '2022-01-03'}, {'Thomas': '2022-06-25'}]
```

Check for incorrect manager:

```
Enter your choice:
1.Extend Password Expiry
2.Exit the script.
 1
Manager Verification
Enter your name:  Thomas

Manager cannot be verified, Please try again!
```

Check for incorrect user:

```
Enter your choice:
1.Extend Password Expiry
2.Exit the script.
 1
Manager Verification
Enter your name:  Reuben
Reuben successfully verified!

Enter the name of the user you want to update :  Alex

User is not mapped under you, cannot change password for Alex
```

Q.7 Using a decorator function check if input is containing upper case, special characters, integers in different functions.

```
import re

def upperchk(func):

    def uchk(st):

        if (not re.search("[A-Z]",st)):
```

```
            print("\nUpper case not found")
        else:
            print("\nUpper case character present!")
        res = func(st)
        return res
    return uchk

def specialchk(func):
    def spchk(st):
        if (not re.search("[!@#$%^&*]",st)):
            print("\nSpecial chars not found")
        else:
            print("\nSpecial character is present!")
        res = func(st)
        return res
    return spchk

def integerchk(func):
    def ichk(st):
        if (not re.search("[0-9]",st)):
            print("\nNumeric value not found")
        else:
            print("\nNumeric value is present!")
        res = func(st)
        return res
    return ichk

@upperchk
@specialchk
@integerchk
def userinput(st):
```

```
    print("\nThe user input is: "+st)


st = input("Enter your string/input : ")
userinput(st)

#OR (if not using @)
#chk = upperchk(specialchk(integerchk(userinput)))
#chk(st)
```

Output:

With correct input:

```
Enter your string/input :  Reuben123#

Upper case character present!

Special character is present!

Numeric value is present!

The user input is: Reuben123#
```

With incorrect input:

```
Enter your string/input :  reuben123

Upper case not found

Special chars not found

Numeric value is present!

The user input is: reuben123
```

Calling the function without '@' method:

```
chk = upperchk(specialchk(integerchk(userinput)))
chk(st)
```

```
Enter your string/input :  Reuben123

Upper case character present!

Special chars not found

Numeric value is present!

The user input is: Reuben123
```

Q.8 Create a User & set the password of the user based upon complexity. Categorize the password as Complex or simple based upon Complexity. Number of passwords remembered will be Five. Password should contain min 1 capital letter,1 number & 1 special character. (Password length should be minimum 8 Char)

```
import re
def passwordcheck(passwd):
    passlist=[]
    invalid=True
    while True:
        if len(passwd)<8:
            break;
        elif (not re.search("[a-z]",passwd)):
            break
        elif (not re.search("[A-Z]",passwd)):
            break
        elif (not re.search("[0-9]",passwd)):
            break
        elif (not re.search("[$#@!%]",passwd)):
            break
```

```
        else:
            print("The password is Complex")
            invalid=False
            break

    if invalid == True:
        print("The password is Simple, Please retry with a stronger password")
        passlist.append(passwd)
    return passwd

finalpasslist = []
uname = input("Enter the name of the user: ")
n = int(input("Enter the number of passwords to enter: "))
for i in range(n):
    passwd = input("\nEnter your password : ")
    password = passwordcheck(passwd)
    finalpasslist.append(password)

if (len(finalpasslist) > 5):
    hist=(len(finalpasslist)-5)
    for i in range(0,hist):
        finalpasslist.remove(finalpasslist[i])

print("\nYour password history : ",finalpasslist)
```

Output:

```
Enter the name of the user:  Reuben
Enter the number of passwords to enter:  6

Enter your password :  Pa$$w0rd123
The password is Complex

Enter your password :  Password
The password is Simple, Please retry with a stronger password

Enter your password :  myPass123
The password is Simple, Please retry with a stronger password

Enter your password :  myPass123!@#
The password is Complex

Enter your password :  password!@#
The password is Simple, Please retry with a stronger password

Enter your password :  test123
The password is Simple, Please retry with a stronger password

Your password history :  ['Password', 'myPass123', 'myPass123!@#', 'password!@#', 'test123']
```

Q.9 Add or Remove a user based upon Date. Take inputs for Date as well as Username from End User. Task should not be performed in case of Past Date.

```
from datetime import datetime
userslist=[]
v=1
while (v==1 or v==2):
    print("Enter your choice:")
    v=int(input("\n1. ADD User \n2. REMOVE User\n3. EXIT:\n"))
    if v == 1:
        n=int(input("\nEnter the number of users you want to ADD: "))
        for i in range(n):
            uname=input("\nEnter the Name of the User:")
            udate=input("Enter the date-time in this format - (yyyy-mm-dd hh:mm): ")
```

```
        dateentry = datetime.strptime(udate,"%Y-%m-%d %H:%M")
        currenttime=datetime.now()
        if (dateentry >= currenttime):
            userslist.append(uname)
            print("\nSucessfully Entered the user")
        else:
            print("Please enter the correct date and time, you cannot enter past dates.")

        print("User List after Addition : ", userslist)

elif v==2:
    n=int(input("\nEnter the number of users you want to REMOVE : "))
    for i in range(n):
        uname=input("\nEnter the Name of the User: ")
        udate=input("Enter the date-time in this format - (yyyy-mm-dd hh:mm):")
        dateentry = datetime.strptime(udate,"%Y-%m-%d %H:%M")
        currenttime=datetime.now()
        if(dateentry >= currenttime):
            if(uname in userslist):
                userslist.remove(uname)
                print("\nUser has been removed successfully")
            else:
                print("\nNo user found")
        else:
            print("Please enter the correct date and time, you cannot enter past dates.")
    print("User List after Deletion : ", userslist)

else:
    break
```

Output:

Adding new users

```
Enter your choice:

1. ADD User
2. REMOVE User
3. EXIT:
 1

Enter the number of users you want to ADD:  3

Enter the Name of the User: Reuben
Enter the date-time in this format - (yyyy-mm-dd hh:mm):  2021-12-27 16:00

Sucessfully Entered the user
User List after Addition :  ['Reuben']

Enter the Name of the User: Thomas
Enter the date-time in this format - (yyyy-mm-dd hh:mm):  2021-12-27 19:00

Sucessfully Entered the user
User List after Addition :  ['Reuben', 'Thomas']

Enter the Name of the User: Alice
Enter the date-time in this format - (yyyy-mm-dd hh:mm):  2021-12-27 19:30

Sucessfully Entered the user
User List after Addition :  ['Reuben', 'Thomas', 'Alice']
```

Trying to add a user with past date

```
Enter your choice:

1. ADD User
2. REMOVE User
3. EXIT:
 1

Enter the number of users you want to ADD:  1

Enter the Name of the User: Bob
Enter the date-time in this format - (yyyy-mm-dd hh:mm):  2021-09-20 19:00
Please enter the correct date and time, you cannot enter past dates.
```

Removing existing users:

```
Enter your choice:

1. ADD User
2. REMOVE User
3. EXIT:
 2

Enter the number of users you want to REMOVE :  2

Enter the Name of the User:  Alice
Enter the date-time in this format - (yyyy-mm-dd hh:mm): 2021-12-27 20:00

User has been removed successfully

Enter the Name of the User:  Thomas
Enter the date-time in this format - (yyyy-mm-dd hh:mm): 2021-11-20 19:00
Please enter the correct date and time, you cannot enter past dates.
User List after Deletion :  ['Reuben', 'Thomas']
```

Removing a user with past date gives an error and does not delete the user

Attempting to remove a user that does not exist

```
Enter your choice:

1. ADD User
2. REMOVE User
3. EXIT:
 2

Enter the number of users you want to REMOVE :  1

Enter the Name of the User:  Alice
Enter the date-time in this format - (yyyy-mm-dd hh:mm): 2021-12-27 18:00

No user found
```

Q.10 Your assignment: Create a Tic Tac Toe game. You are free to use any IDE you like.

Here are the requirements:

- **2 players should be able to play the game (both sitting at the same computer)**
- **The board should be printed out every time a player makes a move**
- **You should be able to accept input of the player position and then place a symbol on the board**

```
import random
randplayer = random.randint(0,1)
def beginSetup():
    players[0] = input("Player 1: ")
    players[1] = input("Player 2: ")

    print("\nX for ",players[0])
    print("0 for ",players[1])
    print("\nFirst turn is for player : ",players[rand])

    print("\n\nBoard Positions:")
    print("  1 | 2 | 3 ")
    print(" ———┼———┼———")
    print("  4 | 5 | 6 ")
    print(" ———┼———┼———")
    print("  7 | 8 | 9 ")

    start = int(input("Press 1 to start the game"))
    return start

def printBoard(board):
    print(" %c | %c | %c " % (board[1],board[2],board[3]))
    print("___|___|___")
    print(" %c | %c | %c " % (board[4],board[5],board[6]))
    print("___|___|___")
    print(" %c | %c | %c " % (board[7],board[8],board[9]))

def startGame():
    printBoard(pos)
    player_turn = randplayer
```

```
    for i in range(9):
        if player_turn % 2 == 0:
            print("\n{}'s turn to play:".format(players[0]))
            p = int(input("Enter the position you want to fill"))
            ch = 'X'
            pos[p] = ch
            printBoard(pos)
            win = checkBoard(ch)
            if win == "nil":
                player_turn = 1
                continue
            else:
                print("\n{} is the winner!!".format(players[0]))
                break
        else:
            print("\n{}'s turn to play:".format(players[1]))
            p = int(input("Enter the position you want to fill"))
            ch = '0'
            pos[p] = ch
            printBoard(pos)
            win = checkBoard(ch)
            if win is "nil":
                player_turn = 0
                continue
            else:
                print("\n{} is the winner!!".format(players[1]))
                break
    else:
        print("\n\nGame Draw!")

def checkBoard(ch):
```

```
    for i in win_cond:
        if (pos[i[0]], pos[i[1]], pos[i[2]]) == (ch,ch,ch):
            winner = players[0]
            break
        elif (pos[i[0]], pos[i[1]], pos[i[2]]) == (ch,ch,ch):
            winner = players[1]
            break
        else:
            winner = "nil"
    return winner


pos = [' ',' ',' ',' ',' ',' ',' ',' ',' ']
players = ['','']
win_cond = [(1,2,3),(4,5,6),(7,8,9),(1,4,7),(2,5,8),(3,6,9),(1,5,9),(3,5,7)]
start = beginSetup()
if start == 1:
    startGame()
else:
    print("Game will not start since you chose to exit!")
```

Output:

```
Player 1:  Reuben
Player 2:  Alice

X for  Reuben
0 for  Alice

First turn is for player :  Reuben


Board Positions:
  1 | 2 | 3
 ___|___|___
  4 | 5 | 6
 ___|___|___
  7 | 8 | 9
Press 1 to start the game 1
    |   |
___|___|___
    |   |
___|___|___
    |   |

Reuben's turn to play:
Enter the position you want to fill 1
 X |   |
___|___|___
    |   |
___|___|___
    |   |
```

```
Alice's turn to play:
Enter the position you want to fill 3
 X |   | 0
___|___|___
   |   |
___|___|___
   |   |

Reuben's turn to play:
Enter the position you want to fill 7
 X |   | 0
___|___|___
   |   |
___|___|___
 X |   |

Alice's turn to play:
Enter the position you want to fill 6
 X |   | 0
___|___|___
   |   | 0
___|___|___
 X |   |

Reuben's turn to play:
Enter the position you want to fill 9
 X |   | 0
___|___|___
   |   | 0
___|___|___
 X |   | X

Alice's turn to play:
Enter the position you want to fill 5
 X |   | 0
___|___|___
   | 0 | 0
___|___|___
 X |   | X

Reuben's turn to play:
Enter the position you want to fill 8
 X |   | 0
___|___|___
   | 0 | 0
___|___|___
 X | X | X

Reuben is the winner!!
```

Randomizing the player who starts first:

```
Player 1:  Reuben
Player 2:  Alice

X for  Reuben
0 for  Alice

First turn is for player :  Alice


Board Positions:
  1 | 2 | 3
 ___|___|___
  4 | 5 | 6
 ___|___|___
  7 | 8 | 9
Press 1 to start the game 3
Game will not start since you chose to exit!
```

The game will not start if you do not enter '1'

Draw condition:

```
Bob's turn to play:
Enter the position you want to fill 9
 X | 0 | 0
___|___|___
 0 | 0 | X
___|___|___
 X | X | 0


Game Draw!
```

****