Python(2180711)

Name : Vishal Sankhat

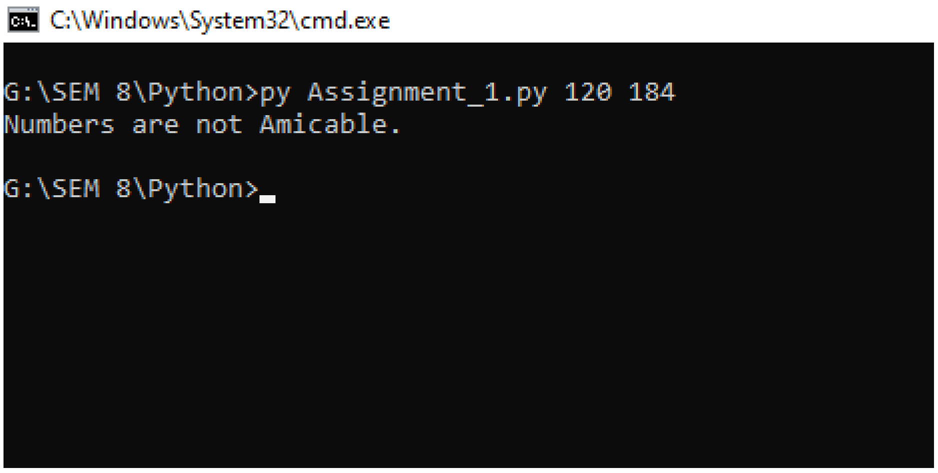
Enrollment No. : 170200107108

Division/Batch : F/F2

Assignment-1 :

```
import math
import sys

def divSum(n) :

	result = 0

	for index in range(2, int(math.sqrt(n)) + 1) :
		if (n % index == 0) :
			if (index == int(n / index)) :
				result = result + index
			else :
				result = result +(index + int(n / index))
	return (result + 1)

def areAmicable(x, y) :

	if (divSum(x) != y) :
		return False

	return (divSum(y) == x)

if((len(sys.argv)) < 3 or (len(sys.argv)) > 3):
	print('Missing required inputs')
	sys.exit()

number1 = int(sys.argv[1])
number2 = int(sys.argv[2])

if (areAmicable(number1,number2)) :
	print ('Numbers are Amicable.')
else :
        print ('Numbers are not Amicable.')
```

OUTPUT :



C:\Windows\System32\cmd.exe

```
G:\SEM 8\Python>py Assignment_1.py 120 184
Numbers are not Amicable.

G:\SEM 8\Python>_
```

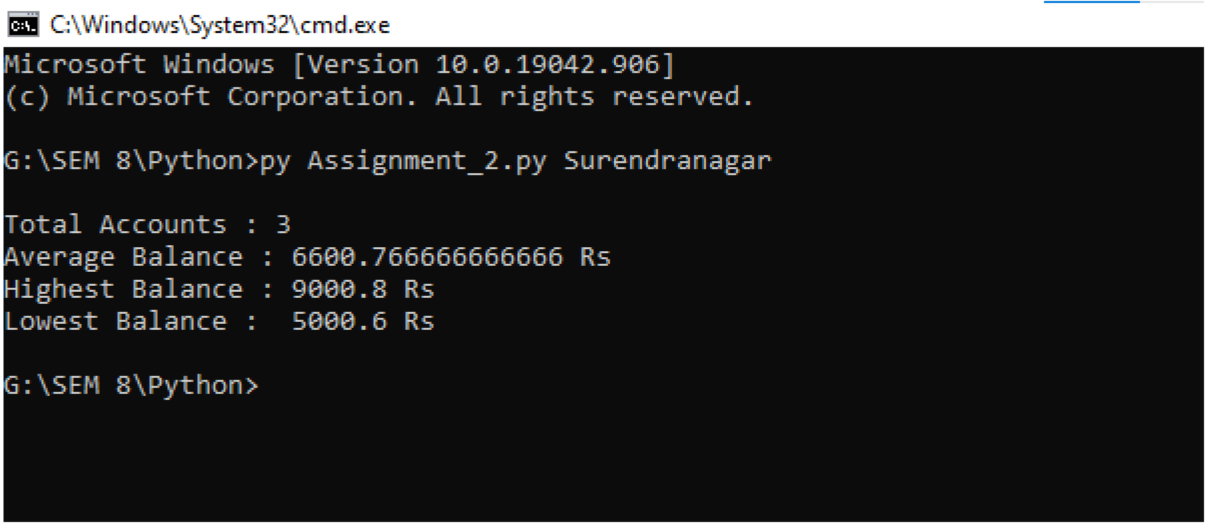
Assignment-2:

```
import sys

class Manager:
    def __init__(self,accountNumber=0,BranchName=None,Balance=0):
        self.accountNumber = accountNumber
        self.BranchName = BranchName
        self.Balance = Balance

    def readFromFile(self,file):
        list = []
        f1 = open(file,'r')
        Lines = f1.readlines()
        f1.close()
        for line in Lines:
            row = line.strip().split(' ')
            object = Manager(int(row[0]),row[1],float(row[2]))
            list.append(object)
        return list

    def getDataBranchWise(self,list,branchName):
        totalAccounts = 0
        totalBalance = 0.0
        higheshBalance = lowestBalance  = list[0].Balance
        isBranchPresent = False

        for object in list:
            if(object.BranchName.lower() == branchName.lower()):
                isBranchPresent = True
                totalAccounts += 1
                totalBalance += object.Balance

                if(object.Balance > higheshBalance):
                    higheshBalance = object.Balance

                if(object.Balance < lowestBalance):
                    lowestBalance = object.Balance

        if(not isBranchPresent):
            print('No Such Branch Found')
            return

        averageBalance = (totalBalance/totalAccounts)
        output ="\nTotal Accounts : {0}\n".format(totalAccounts)
        output +="Average Balance : {0} Rs\n".format(averageBalance)
        output +="Highest Balance : {0} Rs\n".format(higheshBalance)
        output +="Lowest Balance :  {0} Rs".format(lowestBalance)
        print(output)



if(len(sys.argv) > 2 or len(sys.argv) < 2):
    print('Missing required inputs')
    sys.exit()

p1 = Manager()
output = p1.readFromFile('accounts.txt')
p1.getDataBranchWise(output,sys.argv[1])
```

account.txt

```
101 Ahmedabad 8000.00

102 Rajkot 5000.52

103 Surat 4000.62

104 Baroda 8200.14

105 Surendranagar 5000.60

106 Surat 8000.00

107 Ahmedabad 8000.00

108 Surendranagar 9000.80

109 Rajkot 8000.00

110 Ahmedabad 8000.00

111 Jamnagar 9120.00

112 Ahmedabad 8000.00

113 Surendranagar 5800.90

114 Surat 3000.00
```

OUTPUT:



C:\Windows\System32\cmd.exe

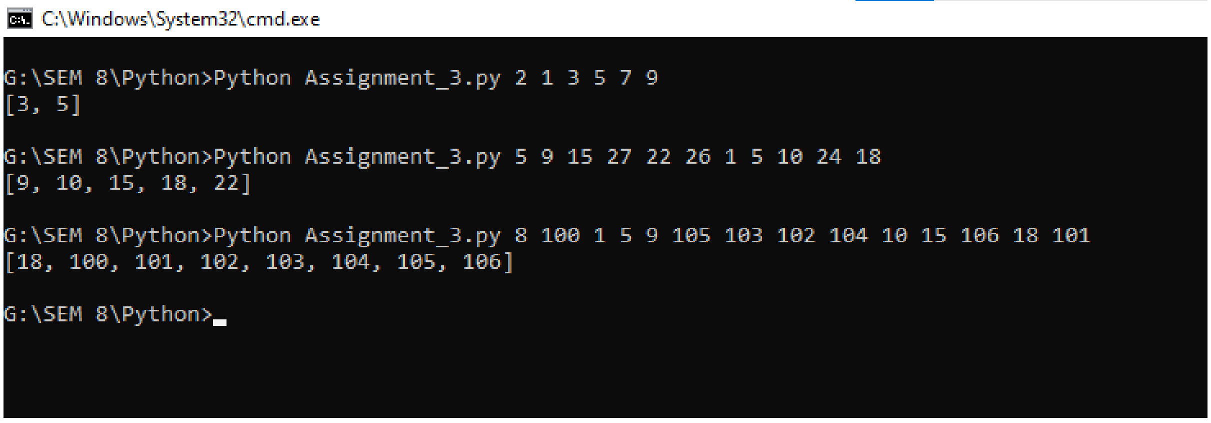
```
Microsoft Windows [Version 10.0.19042.906]
(c) Microsoft Corporation. All rights reserved.

G:\SEM 8\Python>py Assignment_2.py Surendranagar

Total Accounts : 3
Average Balance : 6600.766666666666 Rs
Highest Balance : 9000.8 Rs
Lowest Balance :  5000.6 Rs

G:\SEM 8\Python>
```

Assignment-3:

```
import sys
if len(sys.argv)==1:
	print("Missing required input")
	quit()
K = int(sys.argv[1])
listofnumbers1 = list()
for i in range(2,len(sys.argv)):
	listofnumbers1.append(int(sys.argv[i]))
listofnumbers1.sort()
middle = int(len(listofnumbers1)/2)
median=0
if (len(listofnumbers1)%2)!=0:
#If total numbers are odd
	median = listofnumbers1[middle]
else:
#If total numbers are even
	median = ((listofnumbers1[middle]+listofnumbers1[middle-1])/2)
distancefromMedian = dict()
for num in listofnumbers1:
	if num>median:
#Finding distance
		distancefromMedian[num] = num-median
	else:
#Finding distance
		distancefromMedian[num] = median-num
```

```
sortedValues = sorted(distancefromMedian.values())

distancefromMedian1 = dict()

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

        for key in distancefromMedian:

#Matching sorted values with the keys and sorting the keys according to the distance
from median

                if distancefromMedian[key]==sortedValues[i]:

                        distancefromMedian1[key] = sortedValues[i]


listofnumbers = list()

listofnumbers1.clear()

for key in distancefromMedian1:

#Storing sorted keys in the list

        listofnumbers.append(key)


for i in range(0,K):

#Fetching first K numbers or fetching K numbers that are nearest to the median

        listofnumbers1.append(listofnumbers[i])

listofnumbers1.sort()

print(listofnumbers1)
```

OUTPUT:


C:\Windows\System32\cmd.exe

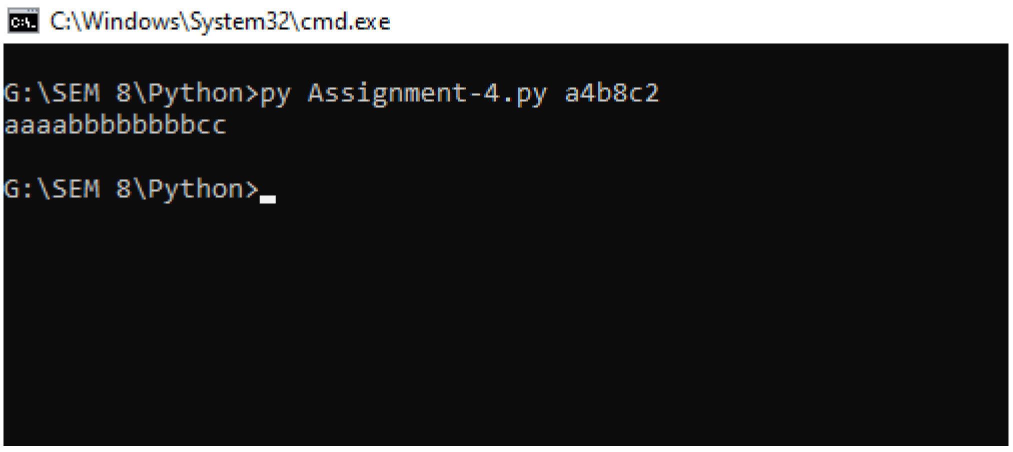
```
G:\SEM 8\Python>Python Assignment_3.py 2 1 3 5 7 9
[3, 5]

G:\SEM 8\Python>Python Assignment_3.py 5 9 15 27 22 26 1 5 10 24 18
[9, 10, 15, 18, 22]

G:\SEM 8\Python>Python Assignment_3.py 8 100 1 5 9 105 103 102 104 10 15 106 18 101
[18, 100, 101, 102, 103, 104, 105, 106]

G:\SEM 8\Python>
```

Assignment-4:

```
import sys
import re

userString = (sys.argv[1])

if not re.match('^[a-zA-Z0-9]+$',userString):
    print('String must have only alphabets and digits.')
    sys.exit()

stringLength = len(userString)
number = ''
output = ''
list = []

for item in range(stringLength):
    currentInput = userString[item]
    if currentInput.isalpha():
        list.append(currentInput)
        number = ''
    else:
        number += str(currentInput)
        if  (item+1) != stringLength and userString[item+1].isalpha():
            list.append(int(number))

if number != '':
    list.append(int(number))

previousInput = ''
newList = []
for currentInput in list:
    if isinstance(currentInput,str) and isinstance(previousInput,str) and previousInput !=
'':
        newList.append(1)
        newList.append(currentInput)
    else:
        newList.append(currentInput)
    previousInput = currentInput

newList.append(1)

for item in range(len(newList)):
    if isinstance(newList[item],str):
        if (item+1) != len(newList):
            output += (newList[item] * newList[item+1])
```

```
print(output)
```

OUTPUT:



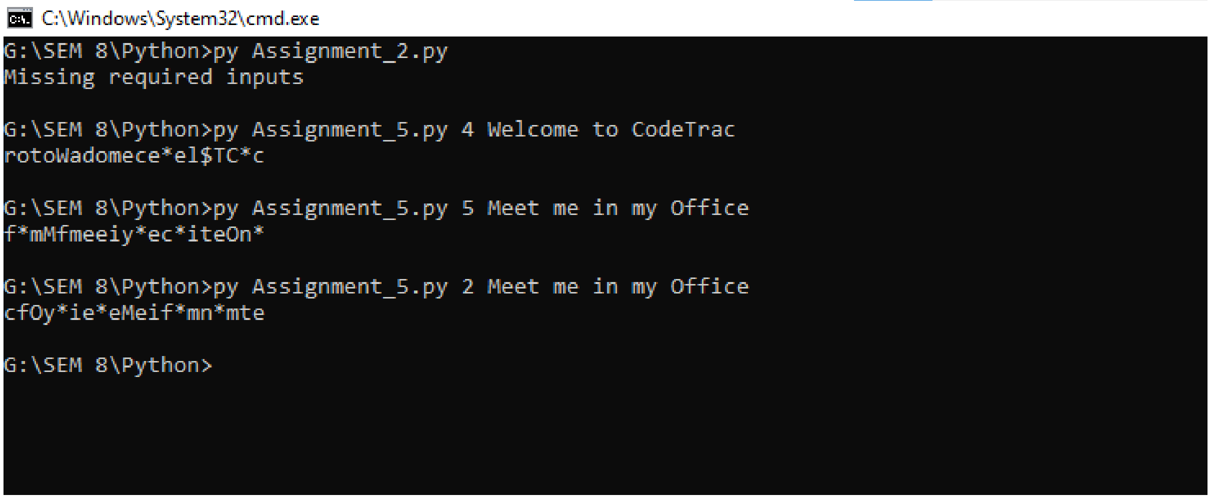
C:\Windows\System32\cmd.exe

```
G:\SEM 8\Python>py Assignment-4.py a4b8c2
aaaabbbbbbbbcc

G:\SEM 8\Python>_
```

Assignment-5

```
import sys
def function(str, k):
    list = []
    for i in range(len(str)):
        if i %k == 0:
            lst = []
            sub = str[i:i+k]
            for j in sub:
                lst.append(j)
            list.append(''.join(lst))
    return list
K = int(sys.argv[1])
userString = '*'.join(sys.argv[2:])
numberOfExtraCharacter = (K - (len(userString) % K))
if(numberOfExtraCharacter != K):
    paddingCharacter = "$" * numberOfExtraCharacter
    userString = userString + paddingCharacter
output = function(userString,K)
string = ''
for index in range(K):
    temp = ''
    for item in output:
        temp += item[index]
    string += temp[::-1]
print(string)
```

OUTPUT:



```
G:\SEM 8\Python>py Assignment_2.py
Missing required inputs

G:\SEM 8\Python>py Assignment_5.py 4 Welcome to CodeTrac
rotoWadomece*el$TC*c

G:\SEM 8\Python>py Assignment_5.py 5 Meet me in my Office
f*mMfmeeiy*ec*iteOn*

G:\SEM 8\Python>py Assignment_5.py 2 Meet me in my Office
cfOy*ie*eMeif*mn*mte

G:\SEM 8\Python>
```