Lab Programs

1.1 Develop a program to read the student details like Name, USN, and Marks in three subjects. Display the student details, total marks and percentage with suitable messages.

```
stName = input("Enter the name of the student : ")
stUSN = input("Enter the USN of the student : ")
stMarks1 = int(input("Enter marks in Subject 1 : "))
stMarks2 = int(input("Enter marks in Subject 2 : "))
stMarks3 = int(input("Enter marks in Subject 3 : "))

print("Student Details\n=========================")
print("%12s"%("Name :"), stName)
print("%12s"%("USN :"), stUSN)
print("%12s"%("Marks 1 :"), stMarks1)
print("%12s"%("Marks 2 :"), stMarks2)
print("%12s"%("Marks 3 :"), stMarks3)
print("%12s"%("Total :"), stMarks1+stMarks2+stMarks3)
print("%12s"%("Percent :"), "%.2f"%((stMarks1+stMarks2+stMarks3)/3))
print("=========================")
```

1.2 Develop a program to read the name and year of birth of a person. Display whether the person is a senior citizen or not.

```
from datetime import date


perName = input("Enter the name of the person : ")
perDOB = int(input("Enter his year of birth : "))

curYear = date.today().year
perAge = curYear - perDOB

if (perAge > 60):
    print(perName, "aged", perAge, "years is a Senior Citizen.")
else:
    print(perName, "aged", perAge, "years is not a Senior Citizen.")
```

2.1 Develop a program to generate Fibonacci sequence of length (N). Read N from the console.

```
num = int(input("Enter the Fibonacci sequence length to be generated : "))

firstTerm = 0
secondTerm = 1
print("The Fibonacci series with", num, "terms is :")
print(firstTerm, secondTerm, end=" ")
for i in range(2,num):
curTerm = firstTerm + secondTerm
print(curTerm, end=" ")
firstTerm = secondTerm
secondTerm = curTerm
print()
```

2.2 Write a function to calculate factorial of a number. Develop a program to compute binomial coefficient (Given N and R).

```
def fact(num):
    if num == 0:
        return 1
    else:
        return num * fact(num-1)

n = int(input("Enter the value of N : "))
r = int(input("Enter the value of R (R cannot be negative or greater than N): "))
nCr = fact(n)/(fact(r)*fact(n-r))
print(n,'C',r," = ","%d"%nCr, sep="")
```

3. *Read N numbers from the console and create a list. Develop a program to print mean, variance and standard deviation with suitable messages.*

```
from math import sqrt

myList = []

num = int(input("Enter the number of elements in your list : "))

for i in range(num):
val = int(input("Enter the element : "))
myList.append(val)

print('The length of list1 is', len(myList))

print('List Contents', myList)


total = 0
for elem in myList:
total += elem

mean = total / num

total = 0
for elem in myList:
total += (elem - mean) * (elem - mean)
variance = total / num

stdDev = sqrt(variance)

print("Mean =", mean)
print("Variance =", variance)
print("Standard Deviation =", "%.2f"%stdDev)
```

4. *Read a multi-digit number (as chars) from the console. Develop a program to print the frequency of each digit with suitable message.*

```
num = input("Enter a number : ")
print("The number entered is :", num)
uniqDig = set(num)
```

```
    for elem in uniqDig:
        print(elem, "occurs", num.count(elem), "times")
```

5. *Develop a program to print 10 most frequently appearing words in a text file. [Hint: Use dictionary with distinct words and their frequency of occurrences. Sort the dictionary in the reverse order of frequency and display dictionary slice of first 10 items]*

```
import sys
import string
import os.path

fname = input("Enter the filename : ") #sample file text.txt also provided

if not os.path.isfile(fname):
    print("File", fname, "doesn't exists")
    sys.exit(0)

infile = open(fname, "r")

filecontents = ""

for line in infile:
    for ch in line:
        if ch not in string.punctuation:
            filecontents = filecontents + ch
        else:
            filecontents = filecontents + ' ' #replace punctuations and \n with space

wordFreq = {}
wordList = filecontents.split()
#Calculate word Frequency

for word in wordList:
    if word not in wordFreq.keys():
        wordFreq[word] = 1
    else:
        wordFreq[word] += 1

#Sort Dictionary based on values in descending order
sortedWordFreq = sorted(wordFreq.items(), key=lambda x:x[1], reverse=True )

#Display 10 most frequently appearing words with their count

print("\n==================================================")
print("10 most frequently appearing words with their count")
print("==================================================")
for i in range(10):
print(sortedWordFreq[i][0], "occurs", sortedWordFreq[i][1], "times")
```

6. *Develop a program to sort the contents of a text file and write the sorted contents into a separate text file. [Hint: Use string methods strip(), len(), list methods sort(), append(), and file methods open(), readlines(), and write()].*

```
import os.path
import sys
```

```
fname = input("Enter the filename whose contents are to be sorted : ")

if not os.path.isfile(fname):
        print("File", fname, "doesn't exists")
        sys.exit(0)

infile = open(fname, "r")

myList = infile.readlines()
# print(myList)

#Remove trailing \n characters
lineList = []
for line in myList:
lineList.append(line.strip())

lineList.sort()
#Write sorted contents to new file sorted.txt

outfile = open("sorted.txt","w")


for line in lineList:
        outfile.write(line + "\n")

infile.close() # Close the input file
outfile.close() # Close the output file


if os.path.isfile("sorted.txt"):
        print("\nFile containing sorted content sorted.txt created successfully")
        print("sorted.txt contains", len(lineList), "lines")
        print("Contents of sorted.txt")
        print("================================================================")
        rdFile = open("sorted.txt","r")
        for line in rdFile:
                print(line, end="")
```

7. *Develop a program to backing Up a given Folder (Folder in a current working directory) into a ZIP File by using relevant modules and suitable methods.*

```
import os
import sys
import pathlib
import zipfile

dirName = input("Enter Directory name that you want to backup : ")
if not os.path.isdir(dirName):
        print("Directory", dirName, "doesn't exists")
        sys.exit(0)
curDirectory = pathlib.Path(dirName)
with zipfile.ZipFile("myZip.zip", mode="w") as archive:
        for file_path in curDirectory.rglob("*"):
                archive.write(file_path, arcname=file_path.relative_to(curDirectory))
```

```
if os.path.isfile("myZip.zip"):
        print("Archive", "myZip.zip", "created successfully")
else:
        print("Error in creating zip archive")
```

8. *Write a function named DivExp which takes TWO parameters a, b and returns a value c (c=a/b). Write suitable assertion for a greater than 0 in function DivExp and raise an exception for when b=0. Develop a suitable program which reads two values from the console and calls a function DivExp.*

```
import sys

def DivExp(a,b):
assert a>0, "a should be greater than 0"
try:
        c = a/b
except ZeroDivisionError:
        print("Value of b cannot be zero")
        sys.exit(0)
else:
        return c

val1 = int(input("Enter a value for a : "))
val2 = int(input("Enter a value for b : "))
val3 = DivExp(val1, val2)
print(val1, "/", val2, "=", val3)
```

9. *Define a function which takes 2 objects representing complex numbers and returns new complex number with a addition of two complex numbers. Define a suitable class Complex to represent the complex number. Develop a program to read N (N greater than 2) complex numbers and to compute the addition of N complex numbers.*

```
class Complex:
    def __init__(self, realp = 0, imagp=0):
            self.realp = realp
            self.imagp = imagp

    def setComplex(self, realp, imagp):
            self.realp = realp
            self.imagp = imagp

    def readComplex(self):
            self.realp = int(input("Enter the real part : "))
            self.imagp = int(input("Enter the real part : "))

    def showComplex(self):
            print('(',self.realp,')','+i','(',self.imagp,')',sep="")

    def addComplex(self, c2):
            c3 = Complex()
            c3.realp = self.realp + c2.realp
                c3.imagp = self.imagp + c2.imagp
            return c3
    def add2Complex(a,b):
            c = a.addComplex(b)
            return c
```

```
def main():
        c1 = Complex(3,5)
        c2 = Complex(6,4)

        print("Complex Number 1")
        c1.showComplex()
        print("Complex Number 2")
        c2.showComplex()

        c3 = add2Complex(c1, c2)

        print("Sum of two Complex Numbers")
        c3.showComplex()

        #Addition of N (N >=2) complex numbers

        compList = []

        num = int(input("\nEnter the value for N : "))

        for i in range(num):
                print("Object", i+1)
                obj = Complex()
                obj.readComplex()
                compList.append(obj)

        print("\nEntered Complex numbers are : ")
        for obj in compList:
                obj.showComplex()

        sumObj = Complex()
        for obj in compList:
                sumObj = add2Complex(sumObj, obj)

        print("\nSum of N complex numbers is", end = " ")
        sumObj.showComplex()
main()
```

10. *Develop a program that uses class Student which prompts the user to enter marks in three subjects and calculates total marks, percentage and displays the score card details. [Hint: Use list to store the marks in three subjects and total marks. Use init method to initialize name, USN and the lists to store marks and total, Use getMarks() method to read marks into the list, and display() method to display the score card details.*

```
class Student:
        def __init__(self, name = "", usn = "", score = [0,0,0,0]):
                self.name = name
                self.usn = usn
                self.score = score

        def getMarks(self):
                self.name = input("Enter student Name : ")
```

```
                self.usn = input("Enter student USN : ")
                self.score[0] = int(input("Enter marks in Subject 1 : "))
                self.score[1] = int(input("Enter marks in Subject 2 : "))
                self.score[2] = int(input("Enter marks in Subject 3 : "))
                self.score[3] = self.score[0] + self.score[1] + self.score[2]

        def display(self):
                percentage = self.score[3]/3
                spcstr = "=" * 81
                print(spcstr)
                print("SCORE CARD DETAILS".center(81))
                print(spcstr)
                print("%15s"%("NAME"), "%12s"%("USN"), "%8s"%"MARKS1", "%8s"%"MARKS2",
                "%8s"%"
                MARKS3","%8s"%"TOTAL","%12s"%("PERCENTAGE"))
                print(spcstr)
                print("%15s"%self.name, "%12s"%self.usn, "%8d"%self.score[0],"%8d"%self.
                score[1],"%8d"%self.score[2],"%8d"%self.score[3],"%12.2f"%percentage)
                print(spcstr)

def main():
        s1 = Student()
        s1.getMarks()
        s1.display()
main()
```