Computer Science Practicals

Ojas Mittal XII-K

Q1

```
num = int(input('enter num: '))
digit = int(input('enter digit: '))

def count(n):
   return len(str(n))

def reverse(n):
   return int(str(n)[::-1])

def hasDigit(n,d):
   return str(d) in str(n)

print('hasdigit: ', hasDigit(num,digit))
print('count:', count(num), '\nreverse: ', reverse(num))
```

```
ojasmittal@pop-os ~/D/Code [1]> python3 q1.py
enter num: 2384658924450
enter digit: 1
hasdigit:  False
count: 13
reverse:  544298564832

ojasmittal@pop-os ~/D/Code> python3 q1.py
enter num: 947437439
enter digit: 9
hasdigit:  True
count: 9
reverse:  934734749
```

Q2

```
number = int(input('enter num: '))

def generateFactors(num):
   factors = []
   div = 1
   while div < num:
       if num % div == 0:
           factors.append(div)
       div += 1
   return factors

def isPrimeNo(num):
   if len(generateFactors(num)) < 2:
       return 'Prime'
   return 'Not Prime'

def isPerfectNo(num):
   if sum(generateFactors(num)) == num:
       return 'Perfect'
   return 'Not Perfect'

print(isPerfectNo(number), isPrimeNo(number))
```

```
ojasmittal@pop-os ~/D/Code> python3 q1.py
enter num: 56
Not Perfect Not Prime

ojasmittal@pop-os ~/D/Code> python3 q1.py
enter num: 6
Perfect Not Prime

ojasmittal@pop-os ~/D/Code> python3 q1.py
enter num: 17
Not Perfect Prime
```

Q3

```
def romanToInt(n):
    roman = {
        "I":1, "V":5, "X":10,
        "L":50, "C":100,
        "D":500, "M":1000
    }

    total = 0
    i = 0
    while i < len(n):
        if i + 1 < len(n) and roman[n[i]] < roman[n[i+1]]:
            total += roman[n[i+1]] - roman[n[i]]
            i += 2
        else:
            total += roman[n[i]]
            i += 1
```

```
   return total

num = input("Enter Roman numeral: ")
print(romanToInt(num))
```

```
ojasmittal@pop-os ~/D/Code (main)> python3 q3.py
Enter Roman numeral: MCMXIV
1914

ojasmittal@pop-os ~/D/Code (main)> python3 q3.py
Enter Roman numeral: MMMDCCCLXXXVIII
3888

ojasmittal@pop-os ~/D/Code (main) [1]> python3 q3.py
Enter Roman numeral: CD
400
```

Q4

```
num = int(input('enter decimal num: '))
con = input('B:binary H:Hex O:Octal\nenter conversion:')

def B(n):
   f = n
   binary = ''
   while f >= 1:
       binary = binary + str(f%2)
       f = f//2
   return binary[::-1]

def H(n):
   f = n
   hex = ''
   j = ord('A')
   while f >= 1:
       digit = f%16
       if digit > 9:
           digit = chr(j + digit - 10)
       hex = hex + str(digit)
       f = f//16
   return hex[::-1]

def O(n):
   f = n
   octal = ''
   while f >= 1:
       octal = octal + str(f%8)
       f = f//8
   return octal[::-1]

def joinList(list):
   string = ''
   for i in list:
       string = string + i
   return string

dict = {
   'B': B(num),
   'H': H(num),
   'O': O(num)
}

print(joinList(dict[con]))
```

```
ojasmittal@pop-os ~/D/Code> python3 q4.py
enter decimal num: 999
B:binary H:Hex O:Octal
enter conversion:H
3E7

ojasmittal@pop-os ~/D/Code> python3 q4.py
enter decimal num: 999
B:binary H:Hex O:Octal
enter conversion:O
1747

ojasmittal@pop-os ~/D/Code> python3 q4.py
enter decimal num: 999
B:binary H:Hex O:Octal
enter conversion:B
1111100111
```

Q5

```
matrix = eval(input('enter matrix: '))
rows = int(input('enter r: '))
columns = int(input('enter c: '))

def reshape(mat,r,c):
   ro = len(mat)
   co = len(mat[0])
   if ro*co != r*c:
       return 'Invalid Dimensions'

   new_mat = []
   flat = []
   for i in range(len(mat)):
       for j in mat[i]:
           flat.append(j)

   index = 0
   for i in range(r):
       new_mat.append([])
       for j in range(c):
           new_mat[i].append(flat[index])
           index += 1

   return new_mat

print(reshape(matrix,rows,columns))
```

```
ojasmittal@pop-os ~/D/Code> python3 q5.py
enter matrix: [[13, 14, 15], [16, 17, 18], [19, 20, 21], [22, 23,
24]]
enter r: 2
enter c: 6
[[13, 14, 15, 16, 17, 18], [19, 20, 21, 22, 23, 24]]
ojasmittal@pop-os ~/D/Code> python3 q5.py
enter matrix: [[1, 2, 3, 4], [5, 6, 7, 8], [9, 10, 11, 12]]
enter r: 39
enter c: 2
Invalid Dimensions
```

Q6

```
blob = "Neither apple nor pine are in pineapple. Boxing rings are
square.\nWriters write, but fingers dont fing.Overlook and oversee
are opposites.\nA house can burn up as it burns down. An alarm goes
off by going on. \n"

def readLines(file):
   file.seek(0)
   return file.readlines()

def dispLine(n,file):
   return readLines(file)[n-1]

def freqTrain(file):
   f.seek(0)
   text = file.read()
   text_break = text.split(" ")
   train = {}

   for i in text_break:
       if i[0] == '\n':
           continue
       elif i[0].lower() not in train:
           train[i[0].lower()] = 1
       else:
           train[i[0].lower()] += 1

   train_list = list(train.items())
   for i in train_list:
       print("Words beginning with ", i[0], ": ", i[1])


with open("blob.txt", "a+") as f:
   f.write(blob)
   lines = readLines(f)

   f.write("Apple seeds contain hydrogen cyanide. That stuff can kill
you")

   lines = readLines(f)
   for i in range(len(lines)):
       print(i+1,": ", lines[i])

   print("Last line-")
   print(lines[-1])
```

```
    print("10th char onwards-")
    print(lines[0][10::])

    index = int(input("Enter line no: ")) - 1
    print(dispLine(index,f))

    freqTrain(f)

'''
ojasmittal@pop-os ~/D/C/q6 (main)> python3 q6.py
1 :  Neither apple nor pine are in pineapple. Boxing rings are
square.

2 :  Writers write, but fingers dont fing.Overlook and oversee are
opposites.

3 :  A house can burn up as it burns down. An alarm goes off by going
on.

4 :  Apple seeds contain hydrogen cyanide. That stuff can kill you
Last line-
Apple seeds contain hydrogen cyanide. That stuff can kill you
10th char onwards-
ple nor pine are in pineapple. Boxing rings are square.

Enter line no: 1
Apple seeds contain hydrogen cyanide. That stuff can kill you
Words beginning with  n :  2
Words beginning with  a :  8
Words beginning with  p :  2
Words beginning with  i :  2
Words beginning with  b :  5
Words beginning with  r :  1
Words beginning with  s :  3
Words beginning with  w :  1
Words beginning with  f :  2
Words beginning with  d :  2
Words beginning with  o :  4
Words beginning with  h :  2
Words beginning with  c :  4
Words beginning with  u :  1
Words beginning with  g :  2
Words beginning with  t :  1
Words beginning with  k :  1
Words beginning with  y :  1
'''
```

Q7

```
def isVowel(l):
   blob = []
   for i in l:
       for j in i.split(' '):
           if j[0].lower() in "aeiou" and j != '\n':
               blob.append(j + " ")
   with open("file2.txt","w") as f:
       f.writelines(blob)


with open("file1.txt", "r") as f:
   lines = f.readlines()
   isVowel(lines)

'''
file1
I saw my name written on the foggy mirror. I live alone.
The dog barked at the fridge again. We don’t have a dog.
Someone keeps putting socks in the freezer. They’re always warm when
I take them out.
The TV turned on by itself last night. It was just static, but it
laughed once.
My phone rang. It was my number. I answered. It was me.
I watered the plant yesterday. Today it moved closer to my bed.
There’s a birthday cake in the oven. Nobody has a birthday.
I opened the door and saw myself walking in.
Every night, my toothbrush is wet before I use it.
The fan spins even when it’s unplugged.
file2
I on I alone.
at again. a in always I out.
on itself It it once.
It I answered. It I it a in oven. a I opened and in.
Every is I use it.
even it’s unplugged.
'''
```

Q8

```
def parseTSV(r):
   rows = []
   for i in r:
       row = i.strip().split("\t")
       rows.append(tuple(row))
   return rows

def regList(l):
   l_new = l
   for i in range(len(l_new)):
       for j in range(len(l_new) - i - 1):
           if int(l_new[j][2]) > int(l_new[j+1][2]):
               l_new[j], l_new[j+1] = l_new[j+1], l_new[j]
   return l_new



with open("stud.tsv","r+") as f:
   lines = f.readlines()
   stud_set = regList(parseTSV(lines))
   young = []
   dep_freq = {}

   for i in stud_set:
       if i[4] in dep_freq:
           dep_freq[i[4]] += 1
       else:
           dep_freq[i[4]] = 1

       if int(i[3]) < 3:
           young.append(i[0] + " " + i[1])

   print("Students w/ yr < 3: ",young,"\nDep_Freq",dep_freq)

'''
ojasmittal@pop-os ~/D/C/q8 (main)> python3 q8.py
Students w/ yr < 3:  ['Anu Sharma', 'Rajat Sen']
Dep_Freq {'MME': 1, 'Biology': 2, 'CSEE': 3}
'''
```

Q9

```
def find_longest_word(f):
   return f[max(list((f.keys())))]

def filter_long_words(f,n):
   words = []
   for j in f.items():
       if j[0] > n:
           words.extend(j[1])
   return words

with open("myfile.txt","r") as f:
   words = f.read().split(" ")

   hist = {}
   freq = {}
   common_word = []
   max_len = 0

   #hist
   for i in words:
       if i in hist:
           hist[i] += 1
       else:
           hist[i] = 1
   #freq
   for k in words:
       if len(k) in freq and k not in freq[len(k)]:
           freq[len(k)].append(k)
       elif len(k) not in freq:
           freq[len(k)] = [k]

   #comm word
   for j in hist.items():
       if j[1] > max_len:
           max_len = j[1]
           common_word = [j[0]]
       elif j[1] == max_len:
           common_word.append(j[0])

   print("words: ", sum(list(hist.values())))
   print("distinct words: ", len(hist))
   print("most common words: ", common_word)
   print("longest words: ",find_longest_word(freq))
   num = int(input("enter filter length: "))
   print("filtered words: ",filter_long_words(freq,num))

'''
file - shadow light mirror shadow light door forest mirror apple
night shadow light fog light shadow apple forest fog mirror light
shadow door apple night fog apple door night mirror forest shadow
light forest light mirror apple shadow fog door night
ojasmittal@pop-os ~/D/C/q9 (main)> python3 q9.py
words:  40
distinct words:  8
most common words:  ['shadow', 'light']
longest words:  ['shadow', 'mirror', 'forest']
enter filter length: 4
filtered words:  ['shadow', 'mirror', 'forest', 'light', 'apple',
'night']
'''
```

Q11

```
def applicants(appl):
   return (len(appl) - 1)

def score(list):
   score = 0
   for i in range(len(list)):
       if i > 1:
           score += int(list[i])
   return score


def n_top(d,n):

   l = list(d.items())
   names = []
   for i in range(len(l)):
       for j in range(len(l)-i-1):
               if l[j][1] > l[j+1][1]:
                   l[j],l[j+1] = l[j+1],l[j]

   l.reverse()

   for j in range(len(l)):
       if j < n:
           names.append(l[j][0])

   return names
```

```
ranks = {}

with open("placement.csv","r") as f:
   appl = f.readlines()

   for i in range(len(appl)):

       data = appl[i].split(',')

       for j in data:
           print(j, end = " ")
       print()


       if i > 0:
           ranks[data[1]] = score(data)

   print("no of applicants: ", applicants(appl))
   num = int(input("filter length: "))
   print("top",num,"applicants: ", n_top(ranks,num))

'''
sample csv
SNO,NAME,MARKS1,MARKS2,MARKS3,MARKS4,MARKS5
1,Aarav,4,5,3,4,5
2,Diya,3,2,4,5,3
3,Vihaan,5,5,5,4,5
4,Isha,2,3,4,2,1
5,Reyansh,4,4,4,4,4
6,Myra,1,2,3,2,1

ojasmittal@pop-os ~/D/C/q11 (main)> python3 q11.py
SNO NAME MARKS1 MARKS2 MARKS3 MARKS4 MARKS5
1 Aarav 4 5 3 4 5
2 Diya 3 2 4 5 3
3 Vihaan 5 5 5 4 5
4 Isha 2 3 4 2 1
5 Reyansh 4 4 4 4 4
6 Myra 1 2 3 2 1
no of applicants:  6
filter length: 4
top 4 applicants:  ['Vihaan', 'Aarav', 'Reyansh', 'Diya']
'''
```