**OSS LAB ASSIGNMENT-3**

# LAKSHAY NARULA

# 9918103075 F3

**Q1)** Character Frequency Counter

input = "LAKSHAY NARULA"

freq = {}

for char in input:

if char in freq:

freq[char] += 1

else:

freq[char] = 1

print (“frequency of every character in '{}' is :\n {}".format(input, str(freq)))

**Q2)** Wrap.py

import textwrap

value = """This Lab is conducted by Avinash Sir in our special semester and this will help us to get used to coding more and help us to enhance our skills and significantly increase our calibre to do more questions in less amount of time."""

x = textwrap.TextWrapper(width=30)

list = wrapper.wrap(text=value)

for element in list:

print(element)

**Q3)** map() using list comprehensions to subtract ‘1’ from each element.

list = [2,3,4,5,6,7,8,9]

def sub(n):

return n-1

update = map(sub, list)

print(list(update))

**Q4)** Demonstrating filter() to remove odd numbers from a list.

num = [1, 2, 4, 5, 7, 8, 10, 11]

def filterOdd(if\_num):

if(if\_num % 2) == 0:

return True

else:

return False

out = filter(filterOdd, num)

print("Filtered seq. is as follows: ", list(out))

**Q5) Find Triplet (a,b,c) such that a+b=c**

def findTriplet(arr, n)

arr.sort()

i = n - 1

while(i >= 0):

j = 0

k = i - 1

while (j < k):

if (arr[i] == arr[j] + arr[k]):

print "numbers are ", arr[i], arr[j], arr[k]

return

elif (arr[i] > arr[j] + arr[k]):

j += 1

else:

k -= 1

i -= 1

print "No such triplet exists"

arr = [ 5, 32, 1, 7, 10, 50, 19, 21, 2 ]

n = len(arr)

findTriplet(arr, n)

**Q6,7)** Parsing .CSV files

from pandas import DataFrame

X = {'Name': ['Lakshay', 'Jalaj', 'Manas'],

'Company': ['Delhivery', 'TCS', 'Amazon'],

'Joined': ['2021', '2020', '2022'],

'Skills': ['DSA', 'Web', '.ML'] }

df = DataFrame(X, columns=['Name', 'Company', 'Joined', 'Skills'])

export\_csv = df.to\_csv(r'Job\_Profile.csv', index=None, header=True)

result = pandas.read\_csv('Job\_Profile.csv')

print(result)

**Q8) mutate() function**

def mutate(d):

x=[d]

i=0

l=len(d)

alp=map(chr,range(97,123))

while i<l:

cop=d

x.append(cop[:i]+cop[i+1:])

if i<l-2:

x.append(cop[:i]+cop[i+1]+cop[i]+cop[i+2:])

elif i<l-1:

x.append(cop[:i]+cop[i+1]+cop[i])

for y in alp:

x.append(cop[:i]+y+cop[i+1:])

for z in alp:

x.append(d+z)

x.append(z+d)

x.append(cop[:i]+z+cop[i:])

i=i+1

return x

print 'HELLO' in mutate('HELLO')

print 'HELOO' in mutate('HELOO')

**Q9) Nearly Equal Function**

def nearly(str1,str2):

count=0

i=j=0

while(i<len(str1) and j<len(str2)):

if(str1[i]!=str2[j]):

count=count+1

if(len(str1)>len(str2)):

i=i+1

elif(len(str1)==len(str2)):

pass

else:

i=i-1

if(count>1):

return False

i=i+1

j=j+1

if(count<2):

return True

str1=input("Enter first string::\n")

str2=input("Enter second string::\n")

boolean=nearly(str1,str2)

if(boolean):

print("Strings are nearly equal.")

else:

print("Strings are not equal.")

**Q10) Checking Anangrams**

from collections import defaultdict

test = ['molest', 'martial', 'me', 'refill', 'now', 'plum']

print("The original list : " + str(test))

temp = defaultdict(list)

for elem in test:

temp[str(sorted(elem))].append(elem)

res = list(temp.values())

print("The grouped Anagrams : " + str(res))