1. Define a class with a generator which can iterate the numbers, which are divisible by 7, between a given range 0 and n.

A)

class Task1:

def gen(self, n):

for i in range(n+1):

if i%7 == 0:

yield i

nums = Task1()

lst=[i for i in nums.gen(100)]

print(lst)

2. Write a program to compute the frequency of the words from the input. The output should output after sorting the key alphanumerically.

A)

word='New to Python or choosing between Python 2 and Python 3? Read Python 2 or Python 3'

wrd=word.split(' ')

dic={}

for i in wrd:

dic[i]=wrd.count(i)

for k, v in sorted(dic.items()):

print(k, ':', v)

3. Define a class Person and its two child classes: Male and Female. All classes have a method "getGender" which can print "Male" for Male class and "Female" for Female class.

A)

class Person:

def getGender(self):

pass

class Male(Person):

def getGender(self):

print('Male')

class Female(Person):

def getGender(self):

print('Female')

male = Male()

female= Female()

male.getGender()

female.getGender()

4. Please write a program to generate all sentences where subject is in ["I", "You"] and verb is in ['Play', "Love"] and the object is in ["Hockey","Football"].

A)

subject=["I", "You"]

verb=["Play", "Love"]

obj=["Hockey","Football"]

sentence\_list = []

for i in subject:

for j in verb:

for k in obj:

sentence\_list.append(i + " " + j + " " + k)

for sentence in sentence\_list:

print(sentence)

5. Please write a program to compress and decompress the string "hello world!hello world!hello world!hello world!"

import zlib

A)

string = 'hello world!hello world!hello world!hello world!'

compress\_string = zlib.compress(bytes(s, 'utf-8'))

print(f"Compressed string is: {compress\_string}")

print(f"Decompressed string is: {zlib.decompress(compress\_string)}")

6. Please write a binary search function which searches an item in a sorted list. The function should return the index of element to be searched in the list.

A)

def binarySearch(list1, n):

low = 0

high = len(list1) - 1

mid = 0

while low <= high:

mid = (high + low) // 2

if list1[mid] < n:

low = mid + 1

elif list1[mid] > n:

high = mid - 1

else:

return mid

return -1

sorted\_list = [1, 4, 5, 12, 54, 678, 1235]

print(binarySearch(sorted\_list, 54))