**Question 1:**

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

class DivisibleBy7:

def \_\_init\_\_(self, n):

self.n = n

def generate(self):

for i in range(0, self.n):

if i % 7 == 0:

yield i

# Example usage

n = int(input("Enter the range: "))

div\_by\_7 = DivisibleBy7(n)

for num in div\_by\_7.generate():

print(num)

**Question 2:**

**Program to compute the frequency of words from the input and output them sorted alphanumerically:**

from collections import Counter

def word\_frequency():

text = input("Enter the text: ")

words = text.split()

word\_count = Counter(words)

for word in sorted(word\_count.keys()):

print(f"{word}:{word\_count[word]}")

word\_frequency()

**Example Input:**

New to Python or choosing between Python 2 and Python 3? Read Python 2 or Python 3.

**Output:**

2:2

3.:1

3?:1

New:1

Python:5

Read:1

and:1

between:1

choosing:1

or:2

to:1

**Question 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:**

class Person:

def getGender(self):

raise NotImplementedError("Subclass must implement abstract method")

class Male(Person):

def getGender(self):

print("Male")

class Female(Person):

def getGender(self):

print("Female")

# Example usage

male = Male()

female = Female()

male.getGender()

female.getGender()

**Output:**

Male

Female

**Question 4:**

**Program to generate all sentences where subject is in ["I", "You"], verb is in ["Play", "Love"], and object is in ["Hockey", "Football"]:**

subjects = ["I", "You"]

verbs = ["Play", "Love"]

objects = ["Hockey", "Football"]

for subject in subjects:

for verb in verbs:

for obj in objects:

print(f"{subject} {verb} {obj}")

**Output:**

I Play Hockey

I Play Football

I Love Hockey

I Love Football

You Play Hockey

You Play Football

You Love Hockey

You Love Football

**Question 5:**

**Program to compress and decompress the string "hello world!hello world!hello world!hello world!":**

import zlib

# String to be compressed

data = "hello world!hello world!hello world!hello world!"

# Compress the string

compressed\_data = zlib.compress(data.encode())

# Decompress the string

decompressed\_data = zlib.decompress(compressed\_data).decode()

print(f"Original Data: {data}")

print(f"Compressed Data: {compressed\_data}")

print(f"Decompressed Data: {decompressed\_data}")

**Output:**

Original Data: hello world!hello world!hello world!hello world!

Compressed Data: b'x\x9c+\xca\xc9\xc8\xcf\xcf\xf0\xcb\xcc\xcfS\x88X\xe7\xd6\xff\x0f\x00\x8f\x84\x11\x8a'

Decompressed Data: hello world!hello world!hello world!hello world!

**Question 6:**

**Binary search function to search an item in a sorted list:**

def binary\_search(arr, target):

low = 0

high = len(arr) - 1

while low <= high:

mid = (low + high) // 2

if arr[mid] == target:

return mid

elif arr[mid] < target:

low = mid + 1

else:

high = mid - 1

return -1 # Element not found

# Example usage

arr = [1, 3, 5, 7, 9, 11, 13, 15, 17]

target = int(input("Enter the number to search: "))

index = binary\_search(arr, target)

if index != -1:

print(f"Element found at index {index}")

else:

print("Element not found")

**Example Input:**

Enter the number to search: 7

**Output:**

Element found at index 3