| Question 1: |
| --- |
|  |

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

Sol:-

class DivisibleBySeven:

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

self.n = n

def numbers(self):

for i in range(self.n):

if i % 7 == 0:

yield i

Question 2:

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

| Suppose the following input is supplied to the program: |
| --- |
|  |

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

| Then, the output should be: |
| --- |
|  |

| 2:2 |
| --- |
|  |

| 3.:1 |
| --- |
|  |

| 3?:1 |
| --- |
|  |

| New:1 |
| --- |
|  |

| Python:5 |
| --- |
|  |

| Read:1 |
| --- |
|  |

| and:1 |
| --- |
|  |

| between:1 |
| --- |
|  |

| choosing:1 |
| --- |
|  |

| or:2 |
| --- |
|  |

to:1

Sol:-

from collections import Counter

sentence = input("Enter a sentence: ")

# split the sentence into words

words = sentence.split()

# count the frequency of each word using Counter

word\_freq = Counter(words)

# sort the words alphabetically

sorted\_words = sorted(word\_freq.keys())

# print the frequency of each word

for word in sorted\_words:

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

| 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.

Sol:-

class Person:

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

self.name = name

def getGender(self):

pass

class Male(Person):

def getGender(self):

print("Male")

class Female(Person):

def getGender(self):

print("Female")

Question 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"].

Sol:-

subjects = ["I", "You"]

verbs = ["Play", "Love"]

objects = ["Hockey", "Football"]

sentences = []

for subject in subjects:

for verb in verbs:

for obj in objects:

sentence = f"{subject} {verb} {obj}."

sentences.append(sentence)

print(sentences)

Question 5:

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

Sol:-

import zlib

# Define the string to be compressed

string\_to\_compress = "hello world!hello world!hello world!hello world!"

# Compress the string

compressed\_string = zlib.compress(string\_to\_compress.encode())

# Print the compressed string

print(compressed\_string)

# Decompress the string

decompressed\_string = zlib.decompress(compressed\_string)

# Print the decompressed string

print(decompressed\_string.decode())

Question 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.

Sol:-

def binary\_search(arr, x):

"""

Searches for element x in a sorted list arr using binary search.

Parameters:

arr (list): Sorted list of elements to search in.

x (any): Element to search for.

Returns:

int: Index of the element in the list if found, otherwise -1.

"""

# Initialize start and end indices of the search range

start, end = 0, len(arr) - 1

while start <= end:

# Calculate the middle index

mid = (start + end) // 2

# If the element is present at the middle

if arr[mid] == x:

return mid

# If the element is smaller than mid, ignore right half

elif arr[mid] > x:

end = mid - 1

# If the element is larger than mid, ignore left half

else:

start = mid + 1

# If element is not present in the list

return -1