**ADITYA AMIN**

**ASSIGN : 14**

|  |
| --- |
| 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.

class DivisibleBySevenGenerator:

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

self.n = n

def \_\_iter\_\_(self):

self.current = 0

return self

def \_\_next\_\_(self):

while self.current <= self.n:

if self.current % 7 == 0:

value = self.current

self.current += 7

return value

self.current += 1

raise StopIteration

# Example usage:

n = 50

generator = DivisibleBySevenGenerator(n)

for num in generator:

print(num)

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

from collections import defaultdict

def word\_frequency(input\_str):

# Create a defaultdict to store word frequencies

word\_freq = defaultdict(int)

# Split the input string into words

words = input\_str.split()

# Iterate through the words and update the word frequencies

for word in words:

word\_freq[word] += 1

# Sort the keys alphanumerically

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

# Print the word frequencies

for key in sorted\_keys:

print(key + ':' + str(word\_freq[key]))

# Example usage:

input\_str = "New to Python or choosing between Python 2 and Python 3? Read Python 2 or Python 3."

word\_frequency(input\_str)

|  |
| --- |
| 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):

return "Unknown"

class Male(Person):

def getGender(self):

return "Male"

class Female(Person):

def getGender(self):

return "Female"

# Example usage:

person = Person()

male = Male()

female = Female()

print(person.getGender()) # Outputs: Unknown

print(male.getGender()) # Outputs: Male

print(female.getGender()) # Outputs: 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"].

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 all generated sentences

for sentence in sentences:

print(sentence)

Question 5:

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

import gzip

# String to compress

input\_string = "hello world!hello world!hello world!hello world!"

# Compress the string

compressed\_data = gzip.compress(input\_string.encode())

# Decompress the compressed data

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

# Print the compressed and decompressed strings

print("Compressed data:", compressed\_data)

print("Decompressed data:", decompressed\_data)

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.

left = 0

right = len(arr) - 1

while left <= right:

mid = left + (right - left) // 2

if arr[mid] == x:

return mid

elif arr[mid] < x:

left = mid + 1

else:

right = mid - 1

return -1