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

In [2]:
```
class DivisibleBySeven:
    def __init__(self, n):
        self.n = n

    def __iter__(self):
        for i in range(self.n):
            if i % 7 == 0:
                yield i
divisible_by_seven = DivisibleBySeven(50)

for num in divisible_by_seven:
    print(num)
```

```
0
7
14
21
28
35
42
49
```

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

In [ ]:
```
# from collections import defaultdict

# read input from the user
input_str = input("Enter a string: ")

# split the input into words
words = input_str.split()

# create a dictionary to store the frequency of each word
freq_dict = defaultdict(int)

# count the frequency of each word
for word in words:
    freq_dict[word] += 1

# sort the keys of the dictionary alphabetically
sorted_keys = sorted(freq_dict.keys())

# print the frequency of each word, sorted alphabetically
for word in sorted_keys:
    print(word, freq_dict[word])
```

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.

In [6]:
```
class Person:
    def getGender(self):
        pass

class Male(Person):
    def getGender(self):
        print("Male")

class Female(Person):
    def getGender(self):
        print("Female")
```

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"].

In [7]:
```
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)
```

```
['I Play Hockey.', 'I Play Football.', 'I Love Hockey.', 'I Love Football.', 'You Play Hockey.', 'You Play Foot
ball.', 'You Love Hockey.', 'You Love Football.']
```

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

In [9]:
```
import zlib

# Define the string to be compressed
string_to_compress = "hello world!hello world!hello world!hello world!"

# Compress the string using zlib
compressed_string = zlib.compress(bytes(string_to_compress, 'utf-8'))

# Print the compressed string
print("Compressed string:", compressed_string)

# Decompress the string using zlib
decompressed_string = zlib.decompress(compressed_string)

# Convert the decompressed bytes to a string
decompressed_string = decompressed_string.decode('utf-8')

# Print the decompressed string
print("Decompressed string:", decompressed_string)
```

```
Compressed string: b'x\x9c\xcbH\xcd\xc9\xc9W(\xcf/\xcaIQ\xcc \x82\r\x00\xbd[\x11\xf5'
Decompressed string: hello world!hello world!hello world!hello world!
```

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.

In [10]:
```
def binary_search(arr, x):
    """
    Searches for the given element in the given sorted array using binary search.

    Args:
    arr: The sorted array to search in.
    x: The element to search for.

    Returns:
    The index of the element in the array if found, or -1 if not found.
    """

    # Set initial values for the search range
    left = 0
    right = len(arr) - 1

    # Loop until the search range is exhausted
    while left <= right:
        mid = (left + right) // 2
        if arr[mid] == x:
            return mid
        elif arr[mid] < x:
            left = mid + 1
        else:
            right = mid - 1

    # If the loop exits without finding the element, it is not in the array
    return -1
```

In [ ]:

In [ ]: