In [31]:
```
#Question 1:
#Define a class with a generator which can iterate the numbers, which are divisible
#7, between a given range 0 and n.
n=int(input("Enter the values of N = "))
class generator:
    def __init__(self,n):
        print("\n number of values = ",n)
        self.n=n
        print("\n self.n  is = ",self.n)
    def solver(self):
        self.n=n
        c=[]
        count=0
        for a in range(1,self.n):
            #print(a)
            #count=1
            if(a%7==0):
                #print("\n \n \n Is divisible by 7 ---->",a)
                c.append(a)
                count=count+1
        print("\n range  number from  1 to ",n," is divisible by 7 is--->",count,"r

g=generator(n)
g.solver()
```

```
 number of values =  60

 self.n  is =  60

 range  number from  1 to  60  is divisible by 7 is---> 8 number will exist with i
n the range which is divisible by 7  ----> [7, 14, 21, 28, 35, 42, 49, 56]
```

In [36]:
```
#Question 1:
#Define a class with a generator which can iterate the numbers, which are divisible
#7, between a given range 0 and n.
#members operators
n=int(input("Enter the values of N = "))
class generator:
    def __init__(self,n):
        print("\n number of values = ",n)
        self.n=n
        print("\n self.n  is = ",self.n)
    def solver(self):
        #self.n=n
        self.c=[]
        self.count=0
        for self.a in range(1,self.n+1):
            #print(a)
            #count=1
            if(self.a%7==0):
                #print("\n \n \n Is divisible by 7 ---->",a)
                self.c.append(self.a)
                self.count=self.count+1
        print("\n range  number from  1 to ",self.n," is divisible by 7 is--->",sel

g=generator(n)
g.solver()
g.solver()  # Call the solver method once
```

```
# Access the attributes from the instance
print("g.n:", g.n)
print("g.c:", g.c)
```

```
 number of values =  80

 self.n  is =  80

 range  number from  1 to  80  is divisible by 7 is---> 11 number will exist with
in the range which is divisible by 7  ----> [7, 14, 21, 28, 35, 42, 49, 56, 63, 7
0, 77]

 range  number from  1 to  80  is divisible by 7 is---> 11 number will exist with
in the range which is divisible by 7  ----> [7, 14, 21, 28, 35, 42, 49, 56, 63, 7
0, 77]
g.n: 80
g.c: [7, 14, 21, 28, 35, 42, 49, 56, 63, 70, 77]
```

In [3]:

```
#2Write a program to compute the frequency of the words from the input. The output
#should output after sorting the key alphanumerically.
a=input()
b=a.split()
print(b)
from collections import Counter
frequency = Counter(b)
print(frequency)
```

```
['hi', 'hi', 'how', 'are', 'are', 'you', 'you', 'dear']
Counter({'hi': 2, 'are': 2, 'you': 2, 'how': 1, 'dear': 1})
```

In [5]:

```
#3 Define a class Person and its two child classes: Male and Female. All classes ha
#method &quot;getGender&quot; which can print &quot;Male&quot; for Male class and &
#class.
class Person(object):
    def getGender( self ):
        return "Unknown"

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

class Female( Person ):
    def getGender( self ):
        return "Female"

aMale = Male()
aFemale= Female()
print(aMale.getGender())
print(aFemale.getGender())
```

```
Male
Female
```

In [8]:

```
#3 Define a class Person and its two child classes: Male and Female. All classes ha
#method &quot;getGender&quot; which can print &quot;Male&quot; for Male class and &
#class
class Person(object):
    def __init__(self):
        self.gender = "unknown"
    def getGender(self):
        print(self.gender)

class Male(Person):
    def __init__(self):
        self.gender = "Male"

class Female(Person):
    def __init__(self):
        self.gender = "Female"

sharon = Female()
doug = Male()
sharon.getGender()
doug.getGender()
```

```
Female
Male
```

In [12]:

```
#4.Please write a program to generate all sentences where subject is in [&quot;I&qu
#verb is in [&quot;Play&quot;, &quot;Love&quot;] and the object is in [&quot;Hockey
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)

for sentence in sentences:
    print(sentence)
```

```
I Play Hockey.
I Play Football.
I Love Hockey.
I Love Football.
You Play Hockey.
You Play Football.
You Love Hockey.
You Love Football.
```

In [3]:

```
#4.Please write a program to generate all sentences where subject is in [&quot;I&qu
#verb is in [&quot;Play&quot;, &quot;Love&quot;] and the object is in [&quot;Hockey
a=input()
b=a.split()
subjects=['i','I','you','YOU','he','HE','SHE','she','they','THEY','THOSE','those']
objects=['football','Football','hockey','Hockey']
verbs=['play','love','PLAY','LOVE']
for i in range(0,len(b)):
    for j in range(0,len(subjects)):
        if(b[i]==subjects[j]):
            print("subjects in the sentance is ----->",b[i],"===",subjects[j])
            #print("aubjects in the sentance is ------->",b[i])
    print('_______________________________________________________________________
    for k in range(0,len(objects)):
        if(b[i]==objects[k]):
            print("objects in the sentance is ----->",b[i],"===",objects[k])
            #print("object in the sentance is ------->",b[i])
    print('_______________________________________________________________________
    for p in range(0,len(verbs)):
        if(b[i]==verbs[p]):
            print("verbs in the sentance is ----->",b[i],"===",verbs[p])
            #print("verbs in the sentance is ------->",b[i])
```

```
subjects in the sentance is -----> he === he
_____________________________________________________________________________
_____________________________________________________________________________
subjects in the sentance is -----> HE === HE
_____________________________________________________________________________
_____________________________________________________________________________
subjects in the sentance is -----> SHE === SHE
_____________________________________________________________________________
_____________________________________________________________________________
subjects in the sentance is -----> she === she
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
_____________________________________________________________________________
verbs in the sentance is -----> play === play
_____________________________________________________________________________
objects in the sentance is -----> football === football
_____________________________________________________________________________
_____________________________________________________________________________
objects in the sentance is -----> hockey === hockey
_____________________________________________________________________________
subjects in the sentance is -----> they === they
_____________________________________________________________________________
_____________________________________________________________________________
subjects in the sentance is -----> those === those
_____________________________________________________________________________
_____________________________________________________________________________
```

In [4]:

```
#Please write a program to compress and decompress the string &quot;hello world!hel
#world!hello world!hello world!&quot;.
#
import zlib

# Original string
original_string = "hello world!hello world!hello world!hello world!"

# Compress the string
compressed_data = zlib.compress(original_string.encode('utf-8'))

# Decompress the string
decompressed_data = zlib.decompress(compressed_data)

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

# Print the results
print("Original String:", original_string)
print("Compressed Data:", compressed_data)
print("Decompressed String:", decompressed_string)
```

```
Original String: hello world!hello world!hello world!hello world!
Compressed Data: b'x\x9c\xcbH\xcd\xc9\xc9W(\xcf/\xcaIQ\xcc \x82\r\x00\xbd[\x11\xf
5'
Decompressed String: hello world!hello world!hello world!hello world!
```

"""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.""" a=[12,999,90,34,56,90,45,65,9923,765,345,231,234] lower=0; upper=len(a) mid=round((lower+upper)/2) print(lower,upper,mid)

In [1]:

```
"""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."""
def binary_search(arr, target):
    left, right = 0, len(arr) - 1

    while left <= right:
        mid = (left + right) // 2

        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1

# Example usage:
sorted_list = [1, 3, 5, 7, 9, 11, 13, 15, 17]
target_element = 7
result = binary_search(sorted_list, target_element)

if result != -1:
    print(f"Element {target_element} found at index {result}.")
else:
    print(f"Element {target_element} not found in the list.")
```

```
Element 7 found at index 3.
```

In [ ]: