Assignment 14 Solutions

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 [1]:
```
class div_generator:
    def __init__(self,in_num):
        self.in_num = in_num
    def get_numbers(self):
        for ele in range(0,self.in_num+1):
            if ele%7 == 0:
                yield ele

output = div_generator(350)
for ele in output.get_numbers():
    print(ele,end=' ')
```

```
0 7 14 21 28 35 42 49 56 63 70 77 84 91 98 105 112 119 126 133 140 147 154 16
1 168 175 182 189 196 203 210 217 224 231 238 245 252 259 266 273 280 287 294
301 308 315 322 329 336 343 350
```

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
```

In [2]:
```
def checkFrequency():
    in_string = input("Enter the Input String: ")
    frequency = {}
    for ele in in_string.split(" "):
        if(frequency.get(ele) == None):
            frequency[ele] = 1
        else:
            frequency[ele] += 1
    for ele in sorted(frequency):
        print(f'{ele}:{frequency[ele]}',end=" ")
checkFrequency()
```

```
Enter the Input String: New to Python or choosing between Python 2 and Python
3? Read Python 2 or Python 3.
2:2 3.:1 3?:1 New:1 Python:5 Read:1 and:1 between:1 choosing:1 or:2 to:1
```

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 [3]:
```
class Person():
    def getGender():
        pass

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

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

Male.getGender()
Female.getGender()
```

```
Male
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 [4]:
```
def generateSentences():
    subject = ['I','You']
    verb = ['Play','Love']
    object = ['Hockey','Football']
    for s in subject:
        for v in verb:
            for o in object:
                print(f'{s} {v} {o}')

generateSentences()
```

```
I Play Hockey
I Play Football
I Love Hockey
I Love Football
You Play Hockey
You Play Football
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 [5]:
```
def compress(in_string):
    output = in_string[0]
    count = 1
    for ele in range(len(in_string)-1):
        if in_string[ele] == in_string[ele+1]:
            count +=1
        else:
            if count > 1:
                output += str(count)
            output += in_string[ele+1]
            count = 1
    if count > 1:
        output += str(count)
    print(output)


def decompress(in_string):
    output = ''
    for ele in range(len(in_string)):
        if in_string[ele].isdigit():
            output += output[-1]*(int(in_string[ele])-1)
        else:
            output += in_string[ele]
    print(output)


compress("hello world!hello world!hello world!hello world!")
decompress("hel2o world!hel2o world!hel2o world!hel2o world!")

compress('ineuron full stack datascience')
decompress('ineuron ful2 stack datascience')
```

```
hel2o world!hel2o world!hel2o world!hel2o world!
hello world!hello world!hello world!hello world!
ineuron ful2 stack datascience
ineuron full stack datascience
```

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 [6]:
```
sorted_list = [1,2,3,4,5,6,7,8,9,10]
def binary_search(in_list,in_num):
    low = 0
    high = len(in_list)-1
    while low <= high:
        mid = high+low//2
        if in_list[mid] < in_num:
            low = mid+1
        elif in_list[mid] > in_num:
            high = mid-1
        else:
            return mid
    else:
        return 'Input Element not in the list'

print(binary_search(sorted_list,8))
print(binary_search(sorted_list,100))
```

```
7
Input Element not in the list
```