**Task Six**

**Generators, List Comprehension & Decorators**

1) Write a program in Python to find out the character in a string which is uppercase using list comprehension.

s="ConsultAdd"

w=[i for i in s if i.isupper()]

print(w)

2) Write a program to construct a dictionary from the two lists containing the names of students and their corresponding subjects. The dictionary should map the students with their respective subjects. Let’s see how to do this using for loops and dictionary comprehension.

students=['Smit','Jaya','Rayyan']

subject=['CSE','Networking','Operating System']

x=zip(students,subject)

print(dict(x))

3) Learn More about Yield, next and Generators

Yield= The yield statement suspends function’s execution and sends a value back to the user, but also helps enable the function where it is left off, when resumed the function continues execution immediately.

Next= The next() function returns the next item in an iterator, the syntax is as *next(iterable, default)*

Generators= Generators are used to create iterators, it is a function which returns an object/iterator which can iterate more than one value at a time.

4) Write a program in Python using generators to reverse the string.

**def** reversing(my\_str):

    length = len(my\_str)

    for i in range(length-1, -1, -1):

        yield my\_str[i]

for char in reversing("pythontrain"):

    print(char)

5) Write an example on decorators.

A decorator is a function which adds on functionality to another function without modifying it.

**def** hi():

    print("inside hi")

**def** greet():

        print("inside greet")

**def** welcome():

            print("inside welcome")

        greet()

        welcome()

        print("still inside hi")

hi()