TASK SIX
GENERATORS, LIST COMPREHENSION AND DECORATORS

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

```
s = "Anjali Byanjankar"
l = [i for i in s if i.isupper()]
print(l)
```

output:
['A', 'B']

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.
HINT - Use Zip function also

Sample input: students = ['Smit', 'Jaya', 'Rayyan'] subjects = ['CSE', 'Networking', 'Operating System']
Expected output: {'Smit' : 'CSE' , 'Jaya' : 'Networking' , 'Rayyan' : 'Operating System'}

```
students = ['Smit', 'Jaya', 'Rayyan']
subjects = ['CSE', 'Networking', 'Operating System']
d = {k:v for k,v in zip(students,subjects)}
print(d)
```

Output:
{'Smit': 'CSE', 'Jaya': 'Networking', 'Rayyan': 'Operating System'}

3. Learn More about Yield, next and Generators

Ans:

Yield is a keyword that is used like return, except the function will return a generator.
A function with yield, when called, returns a Generator

Generators are iterators, a kind of iterable you can only iterate over once.

Python 3 has a built-in function next() which retrieves the next item from the iterator by calling its __next__() method.

4. Write a program in Python using generators to reverse the string.
Input String = “Consultadd Training”

Ans:

```
def r():
  name="Consultadd Training"
  name= name[::-1]
  yield name
res=r()
print(res.__next__())
```

5. Write an example on decorators.
A decorator takes a function, extends it and returns. Yes, a function can return a function.

Ans:

```
def hello(func):
  def inner():
    print("Hello ")
    func()
  return inner

def name():
  print("Anjali")
```

```
obj = hello(name)
obj()
```