**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.

data\_str = "RonakChandgadhia"

data\_uppercase = [i for i in data\_str if i == i.upper()]

print(data\_uppercase)

OUTPUT:

['R', 'C']

**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’}

# using zip function

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

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

print(dict(zip(students,subjects)))

# using loop

res = {}

for keys in students:

    for value in subjects:

        res[keys] = value

        subjects.remove(value)

        break

print("resultant dictionary:"+str(res))

# using list comprehension

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

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

data\_dict = {key:value for (key,value) in zip(students,subjects)}

print("resultant dictionary:" + str(data\_dict))

OUTPUT:

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

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

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

**3.** Learn More about Yield, next and Generators

def infinite\_sequence():

    num = 0

    while True:

        yield num

        num += 1

a= infinite\_sequence()

print(a.\_\_next\_\_())

print(a.\_\_next\_\_())

print(a.\_\_next\_\_())

for i in a:

    print(i)

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

Input String = “Consultadd Training”

input\_string ="Consultadd Training"

def reverse\_s(input\_string):

    for i in range(len(input\_string)-1, -1, -1):

        yield input\_string[i]

a= reverse\_s(input\_string)

print(a.\_\_next\_\_())

print(a.\_\_next\_\_())

print(a.\_\_next\_\_())

for i in a:

    print(i)

OUTPUT:

g

n

i

n

i

a

r

T

d

d

a

t

l

u

s

n

o

C

**5.** Write an example on decorators.

def decorator\_function(original\_function):

    def wrapper\_function(\*args, \*\*kwargs):

        print("Wrapper function is executed before {}".format(original\_function.\_\_name\_\_))

        return original\_function(\*args, \*\*kwargs)

    return wrapper\_function

@decorator\_function

def display():

    print("This is an display function")

display()

@decorator\_function

def sum(a,b):

    print(a+b)

sum(3,3)

OUTPUT:

Wrapper function is executed before display

This is an display function

Wrapper function is executed before sum

6

**END OF TASK SIX**