Implement python generator and decorators

TASK 8.

Date: 17-9-25

Aim:

To write a python program to implement python generator and decorator.

8.1 Write a python program that includes a generator function to produce a sequence of numbers. The generator should be able to.

a. produce a sequence of numbers

b. produce a default sequenced numbers starting from 0, ending at 10,

Algorithm:

1. Define Generator function
2. Initialize current value
3. Generate sequence
4. Get user input
5. Generate Generate object.
6. Print Generated sequence

Program.

```
def number_sequence(start, end, step=1):
    current = start
    while current <= end:
        yield current
        current += step
start = int(input("Enter starting number:"))
```

OUTPUT

0

1

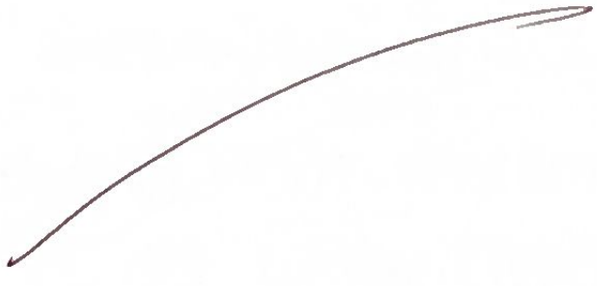
2

```
end = int(input("Enter ending number:"))
step = int(input("Enter step value:"))
sequence_generator = sequence of numbers
for numbers in sequence_generator:
    print(numbers).
```

8.1(b)

Program:

```
def my_generator(n):
    value = 0
    while value < n:
        yield value
        value += 1
for value in my_generator(3):
    print(value)
```



TASK 8.1

Imagine you are working on a messaging application that needs to format messages differently based on user's preferences. User can choose to have their message automatically converted to uppercase or to lowercase. You are provided with two decorators: uppercase_decorator, and lowercase_decorator. These decorators modify the behaviour of the function they decorate by converting the text to uppercase or lowercase respectively. Write a program to implement it.

Algorithm:

1. Create Decorators
2. Define Function
3. Define Street Function
4. Execute the program

Program:

```
def uppercase_decorator(func):
    def wrapper(text):
        return func(text).upper()
    return wrapper

def lowercase_decorator(func):
    def wrapper(text):
        return func(text).lower()
```

OUTPUT

HI, I AM CREATED BY A FUNCTION PASSED AS AN ARGUMENT.

hi, iam created by a function passed as an argument.

```
    return wrapper.
@upper_case_decorator
def shout(text):
    return text.
@lower_case_decorator
def whisper(text):
    return text
def greet(func):
    greeting = func("Hi, I am created by a function
                    passed as an argument")
    print(greeting)
greet(shout)
greet(whisper)
```

VEL TECH - CSE	
EX NO.	6
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	3
VIVA VOCE (3)	3
RECORD (4)	4
TOTAL (15)	
SIGN WITH DATE	15

Result:

Thus the python program to implement python generator and decorator was successfully executed. OUTPUT was verified