**TASK 8: Implement python generator and decorators (CO2-K3)**

**Problem 1:**

Ranjith wants to print Fibonacci series for first 25 numbers. Help Ranjith to generate Fibonacci series using generator function in python.

**Program:**

def fibonacci\_generator(n=5):

"""Generate first n Fibonacci numbers."""

a, b = 0, 1

count = 0

while count < n:

yield a

a, b = b, a + b

count += 1

# Print first 5 Fibonacci numbers

for num in fibonacci\_generator(5):

print(num, end=" ")

**Problem 2:**

Ranjith wants to print the squares of the numbers 0 through 4 . create the generator object that will produce the squares of the numbers 0 through 4 when iterated over.

**Program:**# Program 2: Squares of numbers 0 through 4 using generator

def squares\_generator(start=0, end=4):

"""Generate squares of numbers from start through end."""

for i in range(start, end + 1):

yield i \* i

# Print squares of numbers from 0 to 4

for sq in squares\_generator(0, 4):

print(sq)

**Problem 3:**

Priya wants to print a message before and after doing some calculations.  
Write a Python program using a **decorator** that prints:

* "Starting..." before the function runs
* "Done." after the function finishes

Use this decorator for a function that calculates the **area of a rectangle**.

**def my\_decorator(func):**

**def wrapper(length, breadth):**

**print("Starting...")**

**result = func(length, breadth)**

**print("Done.")**

**return result**

**return wrapper**

**@my\_decorator**

**def area\_of\_rectangle(length, breadth):**

**return length \* breadth**

**# Test**

**print("Area:", area\_of\_rectangle(5, 10))**