Task 8 : Implement python generator and decorators

a) Fibonacci Number Generator

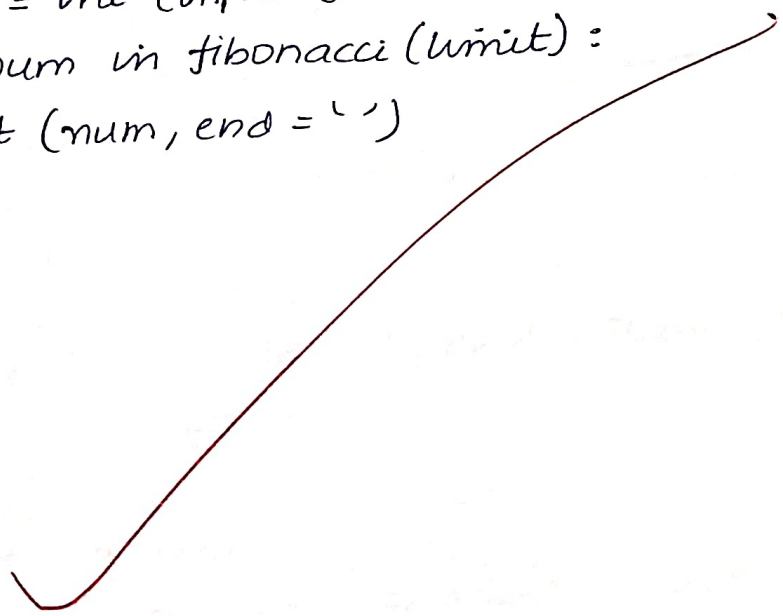
Aim :- To create a python program using a generator function that produce Fibonacci number using user defined limit

Algorithm :-

1. Start
2. Define a generator fibonacci(n)
   - Initialize a=0, b=1
   - Yield values while a≤n
3. Input limit n from user
4. Call generator and print each value
5. End

program :-

```
def fibonacci(n):
    a, b = 0, 1
    while a <= n:
        yield a
        a, b = b, a+b
limit = int (input ("Enter limit : "))
for num in fibonacci (limit):
    print (num, end = ' ')
```

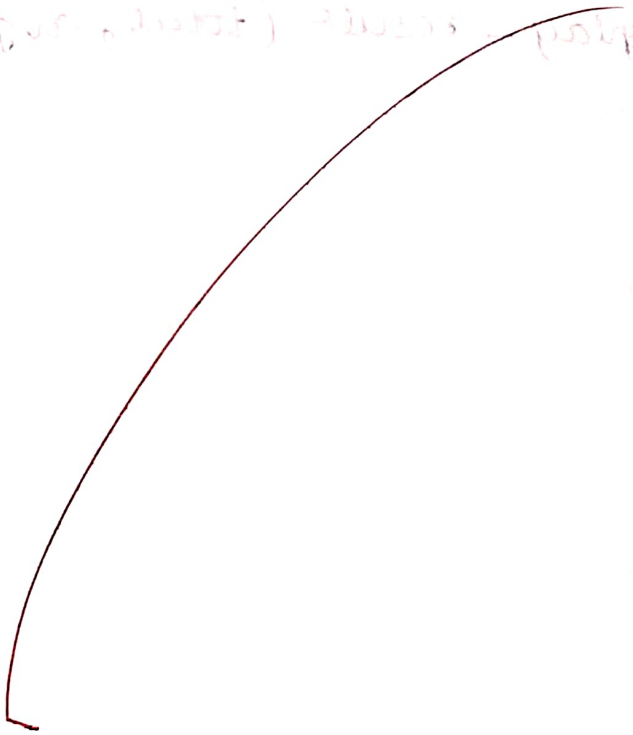


Result :- Thus, python program to generate fibonacci sequence using user defined limit is implemented successfully.

Sample output

Enter limit : 50

0 1 1 2 3 5 8 13 21 34



26/9/25

Task 8(b): Measuring Function Execution Time with a Decorator

Aim: To create a python program using a decorator that measures and displays the execution time of a function that sorts a list of random numbers.

Algorithm:

1. Start
2. Import time and random modules
3. Define a decorator timer(func)
   - Record start end time
   - print time taken
4. Define a function sort_numbers()
   - Generate random list
   - Sort the list
5. Apply the decorator to sort_numbers()
6. Call the function
7. End

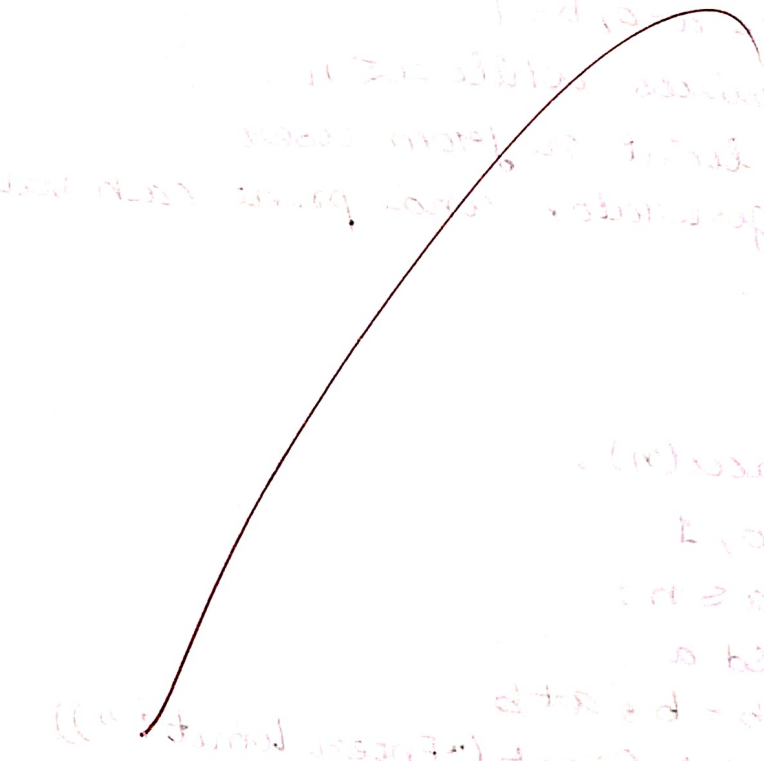
program:

```
import time, random
def timer(func):
    def wrapper():
        start = time.time()
        func()
        end = time.time()
        print("Time taken:", end-start, "seconds")
    return wrapper
@timer
def sort_numbers():
    nums = [random.randint(1,1000) for _ in range(10000)]
    nums.sort()

sort_numbers()
```

Sample output:

Time taken : 0.0042 seconds



VEL TECH	
EX No.	8
PERFORMANCE (5)	5
RESULT AND ANALYSIS (3)	3
VIVA VOCE (3)	3
RECORD (4)	4
TOTAL (15)	15
SIGN WITH DATE	

Result : Thus, python program for measuring function Execution time with a decorator is executed Successfully.