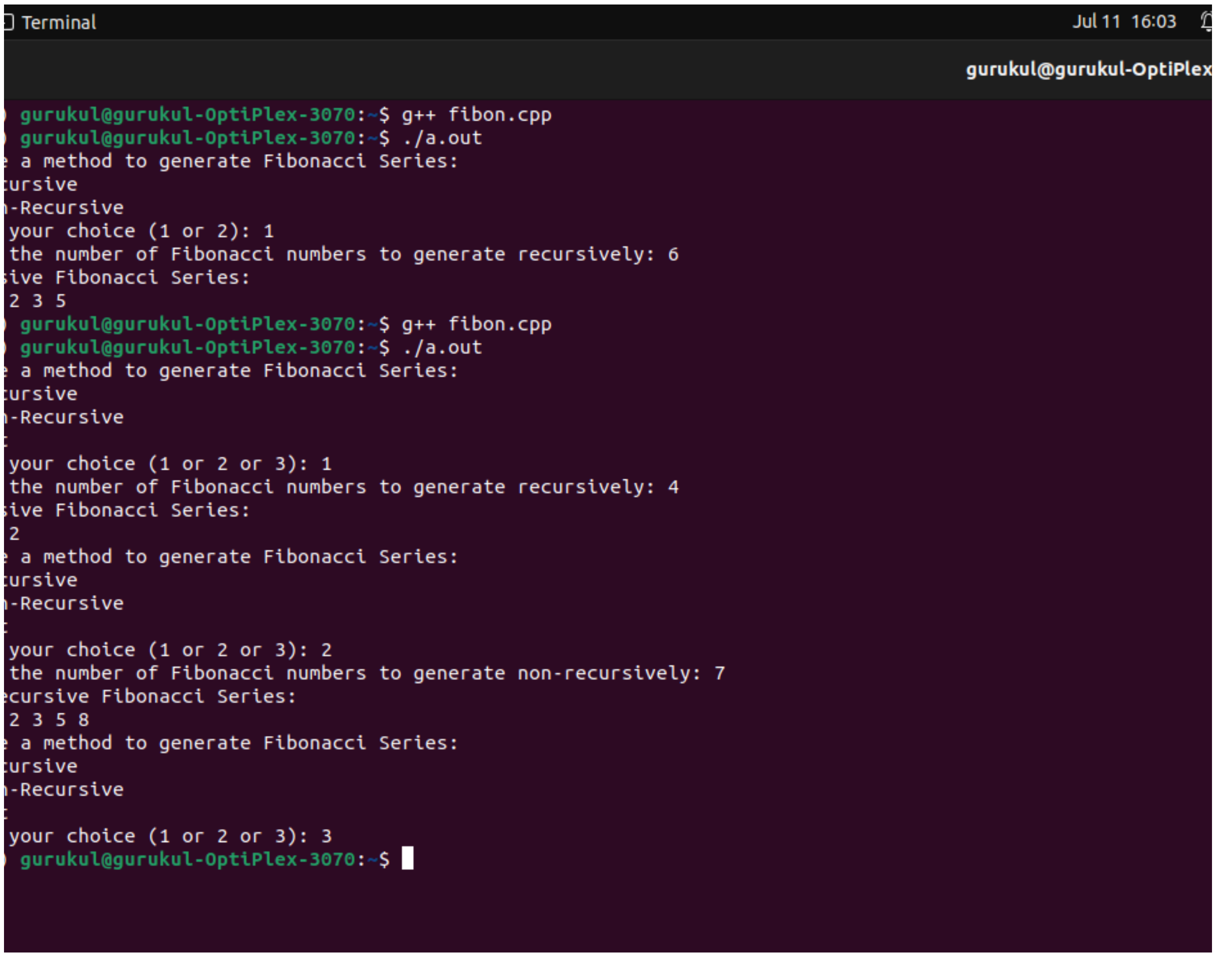
Name : Shraddha Rajkumar Kotwar
Roll No.: 14

Write a program non-recursive and recursive program to calculate Fibonacci numbers and analyze their time and space complexity.

```
#include <iostream>
using namespace std;

int fibonacciRecursive(int n) {
        if (n <= 1)
        return n;
        else
        return fibonacciRecursive(n-1) + fibonacciRecursive(n-2);
}

void fibonacciNonRecursive(int n) {
        int a = 0, b = 1, c;
        cout << "Non-Recursive Fibonacci Series:" << endl;
        cout << a << " " << b << " ";

        for (int i = 2; i < n; ++i) {
        c = a + b;
        cout << c << " ";
        a = b;
        b = c;
        }
        cout << endl;
}

int main() {
        int choice, n;
        do{
        cout << "Choose a method to generate Fibonacci Series:" << endl;
        cout << "1. Recursive" << endl;
        cout << "2. Non-Recursive" << endl;
        cout<<"3.EXit"<<endl;
        cout << "Enter your choice (1 or 2 or 3): ";
        cin >> choice;
        switch(choice) {
        case 1:
        cout << "Enter the number of Fibonacci numbers to generate recursively: ";
        cin >> n;
        cout << "Recursive Fibonacci Series:" << endl;
        for (int i = 0; i < n; ++i) {
                cout << fibonacciRecursive(i) << " ";
        }
        cout << endl;
        break;
        case 2:
        cout << "Enter the number of Fibonacci numbers to generate non-recursively: ";
        cin >> n;
        fibonacciNonRecursive(n);
        break;
        case 3:
        cout<<"Exit";
        break;
        default:
        cout << "Invalid choice. Please enter 1 or 2." << endl;
        break;
        }
        }while(choice!=3);

        return 0;
}
```



```
Terminal                                                         Jul 11  16:03
                                                        gurukul@gurukul-OptiPlex
) gurukul@gurukul-OptiPlex-3070:~$ g++ fibon.cpp
) gurukul@gurukul-OptiPlex-3070:~$ ./a.out
e a method to generate Fibonacci Series:
ursive
n-Recursive
 your choice (1 or 2): 1
 the number of Fibonacci numbers to generate recursively: 6
sive Fibonacci Series:
 2 3 5
) gurukul@gurukul-OptiPlex-3070:~$ g++ fibon.cpp
) gurukul@gurukul-OptiPlex-3070:~$ ./a.out
e a method to generate Fibonacci Series:
ursive
n-Recursive
t
 your choice (1 or 2 or 3): 1
 the number of Fibonacci numbers to generate recursively: 4
sive Fibonacci Series:
 2
e a method to generate Fibonacci Series:
ursive
n-Recursive
t
 your choice (1 or 2 or 3): 2
 the number of Fibonacci numbers to generate non-recursively: 7
ecursive Fibonacci Series:
 2 3 5 8
e a method to generate Fibonacci Series:
ursive
n-Recursive
t
 your choice (1 or 2 or 3): 3
) gurukul@gurukul-OptiPlex-3070:~$ █
```

Analysis of Time and Space Complexity

Recursive Fibonacci

- **Time Complexity**: O(2^n)
  - Each call to `recursiveFibonacci` results in two more calls until reaching the base case.
  - This creates a binary tree of calls with height `n`, resulting in an exponential number of calls.
- **Space Complexity**: O(n)
  - The space complexity is determined by the depth of the call stack, which can go up to `n`.

Non-Recursive Fibonacci

- **Time Complexity**: O(n)

    - The iterative approach involves a single loop running n times.
- **Space Complexity**: O(1)
    - Only a constant amount of extra space is used for variables a, b, and c.

The non-recursive (iterative) approach is significantly more efficient in terms of both time and space complexity compared to the recursive approach. The recursive approach, while simple to implement, is not practical for large values of n due to its exponential time complexity and linear space complexity. The iterative approach, on the other hand, can handle large values efficiently with linear time complexity and constant space complexity.