**Rajan Gautam**

**19BCP101**

**Div. II, CE 19**

**SOT, PDEU**

**Pandit Deendayal Energy University**

**School of Technology**

**Design & Analysis of Algorithm (20CP209P)**

**B. Tech - Computer Science & Engineering (Sem-IV)**

**Table of Contents**

[**Lab 4 Assignment: Implementation of Recursive/Non-Recursive Function and its Analysis 2**](#_Toc66027327)

[**AIM 1: To write a C/C++ Program to implement Recursive/Non-Recursive Function and its Analysis 2**](#_Toc66027328)

[**CODE: 2**](#_Toc66027329)

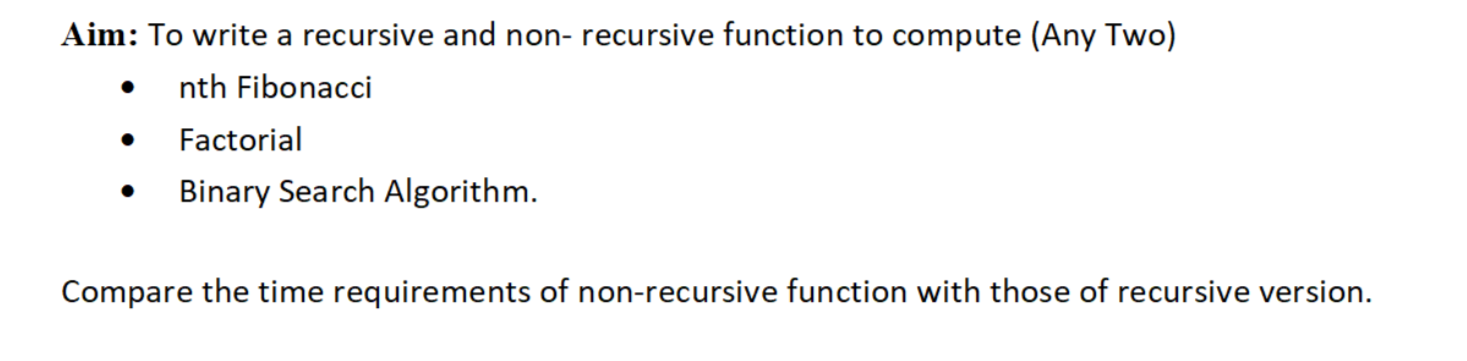
[**OUTPUT: 5**](#_Toc66027330)

[**Comparison for Factorial Function 5**](#_Toc66027331)

[**Comparison for Fibonacci Function 6**](#_Toc66027332)

# Lab 4 Assignment: Implementation of Recursive/Non-Recursive Function and its Analysis

## AIM 1: To write a C/C++ Program to implement Recursive/Non-Recursive Function and its Analysis



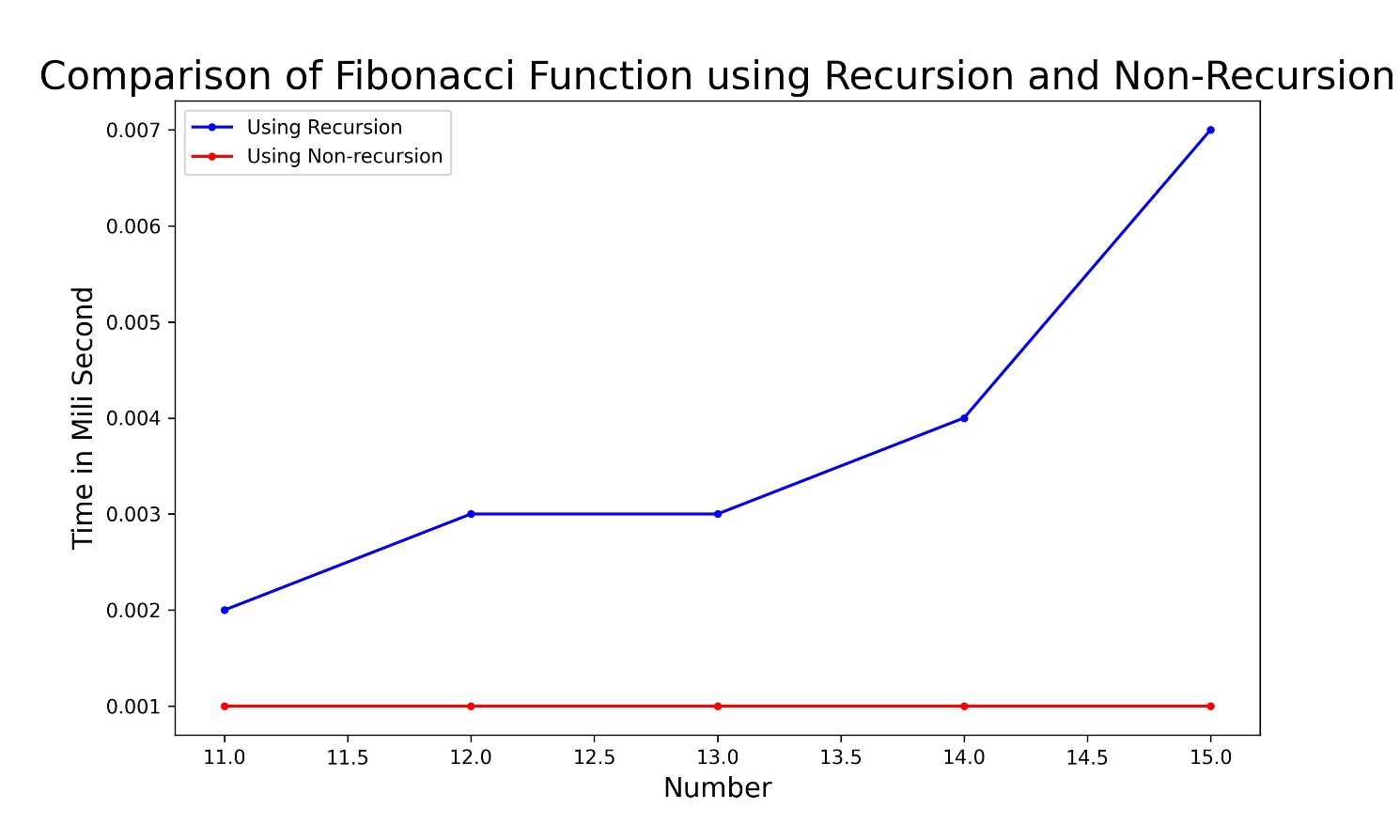
### CODE:

1. **#include<bits/stdc++.h>**
2. **#include <iostream>**
4. **using namespace std;**
6. **int recursive\_fibonacci(int n)**
7. **{**
8. **if (n <= 1)**
9. **{**
10. **return n;**
11. **}**
12. **else**
13. **{**
14. **return recursive\_fibonacci(n-1) + recursive\_fibonacci(n-2);**
15. **}**
16. **}**
18. **int non\_recursive\_fibonacci(int n)**
19. **{**
20. **int n1 = 0, n2 = 1, next;**
21. **for(int i = 0; i < n - 1; i++)**
22. **{**
23. **next = n1 + n2;**
24. **n1 = n2;**
25. **n2 = next;**
26. **}**
27. **return next;**
28. **}**
30. **int recursive\_factorial(int n)**
31. **{**
32. **if(n == 0)**
33. **{**
34. **return 1;**
35. **}**
36. **else**
37. **{**
38. **return n \* recursive\_factorial(n - 1);**
39. **}**
40. **}**
42. **int non\_recursive\_factorial(int n)**
43. **{**
44. **int factorial = 1;**
45. **for(int i = 1; i <= n; i++)**
46. **{**
47. **factorial = factorial \* i;**
48. **}**
49. **return factorial;**
50. **}**
52. **int main()**
53. **{**
54. **int m, n;**
55. **cout<<"\t Enter the value of Factorial and Fibonacci: ";**
56. **cin>>m;**
57. **cin>>n;**
59. **cout<<"\t Time Taken by the Recursive and Non-Recursive Factorial"<<endl;**
60. **cout<<"\t Using Recursion: "<< m << "! = "<< recursive\_factorial(m)<<endl;**
61. **cout<<"\t Using Non-Recursion: "<< m << "! = "<< non\_recursive\_factorial(m)<<endl;**
62. **cout<<"\tNumber \t Time Taken by Recursive \tTime Taken by Non-Recursive"<<endl;**
63. **while(m > 5)**
64. **{**
65. **clock\_t ft1 = clock();**
66. **recursive\_factorial(n);**
67. **clock\_t ft2 = clock() - ft1;**
69. **clock\_t nft1 = clock();**
70. **non\_recursive\_factorial(n);**
71. **clock\_t nft2 = clock() - nft1;**
73. **cout<<"\t"<<m<<" \t "<< fixed << setprecision(3)<< (float)ft2 / CLOCKS\_PER\_SEC \* 1000 << " ms "<<"\t\t\t"<<(float)nft2 / CLOCKS\_PER\_SEC \* 1000<< " ms\n";**
75. **m = m - 1;**
76. **}**
77. **cout<<"\n\t Time Taken by the Recursive and Non-Recursive Fibonacci Series"<<endl;**
78. **cout<<"\t nth Term Using Recursion: " <<recursive\_fibonacci(n)<<endl;**
79. **cout<<"\t nth Term Using Non-Recursion: " << non\_recursive\_fibonacci(n)<<endl;**
80. **cout<<"\tNumber \t Time Taken by Recursive \tTime Taken by Non-Recursive"<<endl;**
81. **while(n > 10)**
82. **{**
83. **clock\_t ft1 = clock();**
84. **recursive\_fibonacci(n);**
85. **clock\_t ft2 = clock() - ft1;**
87. **clock\_t nft1 = clock();**
88. **non\_recursive\_fibonacci(n);**
89. **clock\_t nft2 = clock() - nft1;**
91. **cout<<"\t"<<n<<" \t "<< fixed << setprecision(3)<< (float)ft2 / CLOCKS\_PER\_SEC \* 1000 << " ms "<<"\t\t\t"<<(float)nft2 / CLOCKS\_PER\_SEC \* 1000<< " ms\n";**
93. **n = n - 1;**
94. **}**
96. **return 0;**

## OUTPUT:

## Comparison for Factorial Function

## Comparison for Fibonacci Function

****

**Link:** <https://github.com/rgautam320/Design-and-Analysis-of-Algorithm-Lab/tree/master/Lab_4_Recursion>