**Practical - 2**

**Implement functions to print nth Fibonacci number using iteration and recursive method.Compare the performance of two methods by counting number of steps executed on various inputs.Also draw a comparative chart. (Fibonacci series 1, 1, 2, 3, 5, 8..... Here 8 is the 6th Fibonacci number)**

**Code:**

#include<stdio.h>

int count\_it=0;

int count\_rec=0;

void itfibonacci(int n)

{

intfirst=0,second=1,third,i; count\_it++;

for(i=2;i<=n;i++){

count\_it++; third=first+second; count\_it++;

first=second;

count\_it++;

second=third;

count\_it++;

} printf("%d",third); count\_it++;

}

int recfibonacci(intn){ if(n<=1){ count\_rec++; count\_rec++;

return n;

}

Else

{

count\_rec++;

return recfibonacci(n-1)+recfibonacci(n-2);

} }

void main()

{

printf("21012021003\_AMIT GOSWAMI\n");

int n,i;

printf("Enter the number of element : ");

scanf("%d",&n);

printf("\nFibonacci series using iteration");

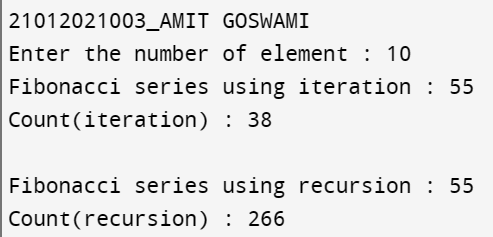
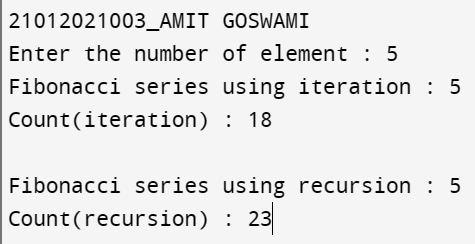
itfibonacci(n); printf("\nCount(iteration) : %d",count\_it);

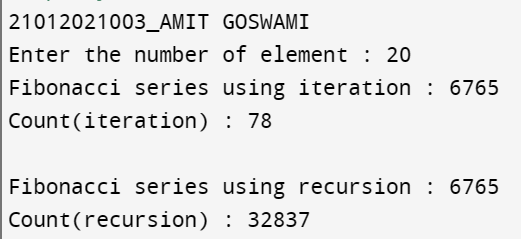
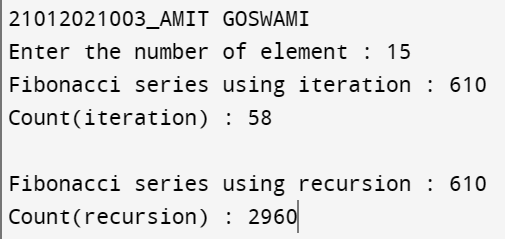
printf("\n\nFibonacci series using recursion : ");

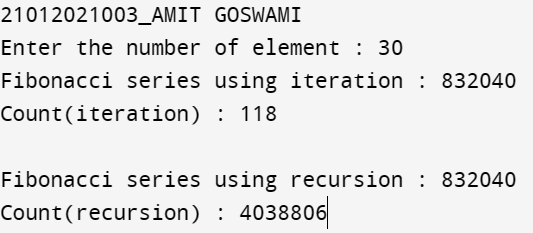
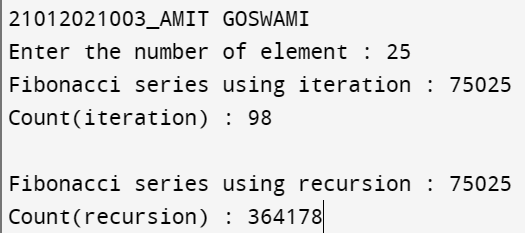
printf("%d",recfibonacci(n));

printf("\nCount(recursion) : %d",count\_rec); }

**Output:**



****



**Analysis:**

|  |  |  |
| --- | --- | --- |
| **Position of number** | **Count with recursive method** | **Count with iteration method** |
| 5 | 23 | 18 |
| 10 | 266 | 38 |
| 15 | 2960 | 58 |
| 20 | 32837 | 78 |
| 25 | 364178 | 98 |
| 30 | 4038806 | 118 |

0

500000

1000000

1500000

2000000

2500000

3000000

3500000

4000000

4500000

0

5

10

15

20

25

30

35

ANALYSIS OF FIBONACCI USING ITERATION

Count with recursive method

Count with iteration method

**Conclusion:**

From the above practical we got to know that recursion is a fast process for smaller numbers,but as we increase the numbers, it can be quite a lengthy method, and to avoid that iteration can be a more reliable method.