# Practical - 2

**Aim:**

Implement functions to print nth Fibonacci number using iteration and recursive methods. Compare the performance of two methods by counting the 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_ite = 0;
int count_rec = 0;

int fib_ite_142(int n) {
    if (n <= 1) {
        count_ite++;
        return n;
    }
    else {
        int t1 = 0, t2 = 1, nextTerm;
        for (int i = 1; i <  n; ++i) {
            count_ite++;
            nextTerm = t1 + t2;
            t1 = t2;
            t2 = nextTerm;
        }
        return nextTerm;
    }
}

int fib_rec_142(int n) {
    if (n <= 1) {
        count_rec++;
```

```
        return n;
    }
    else {
        count_rec++;
        return fib_rec_142(n - 1) + fib_rec_142(n - 2);
    }
}

int main() {
    int n;
    printf("\nEnter Number to find Fibonacci Value: ");
    scanf("%d", &n);
    int result_ite = fib_ite_142(n);
    printf("\n------------------------------------------");
    printf("\nEnroll Number: 142");
    printf("\nLoop Method");
    printf("\nFibonacci of Number %d is: %d", n, result_ite);
    printf("\nCount of Step of Algorithm is : %d", count_ite);
    printf("\n------------------------------------------");
    int result_rec = fib_rec_142(n);
    printf("\n------------------------------------------");
    printf("\nEnroll Number: 142");
    printf("\nRecursive Method");
    printf("\nFibonacci of Number %d is: %d", n, result_rec);
    printf("\nCount of Step of Algorithm is : %d", count_rec);
    printf("\n------------------------------------------");
    return 0;
}
```

**Output:**

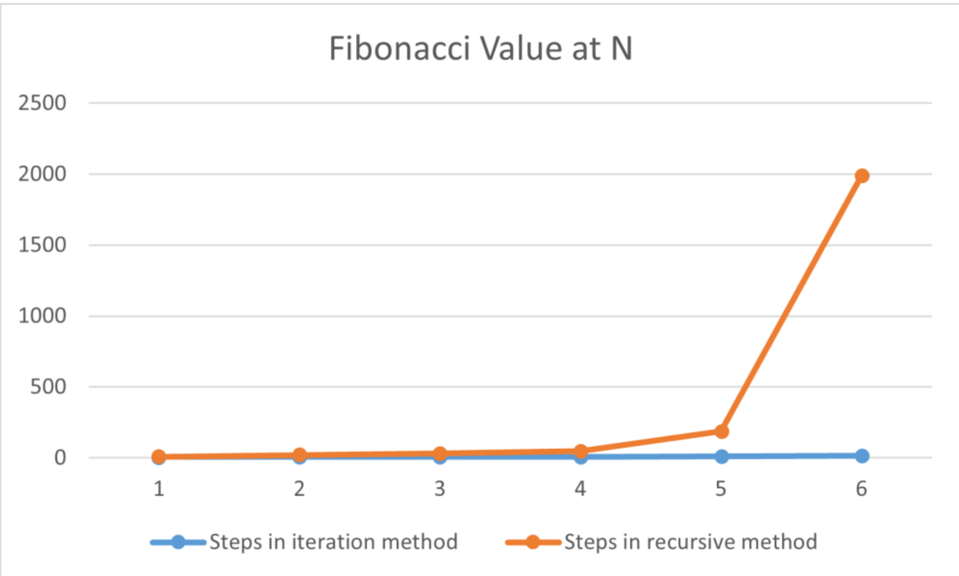
```
Vatsal  …\DAA\Prac2  main !?  v6.3.0  23:51  .\fibonacci.exe

Enter Number to find Fibonacci Value: 5

------------------------------------------
Enroll Number: 142
Loop Method
Fibonacci of Number 5 is: 5
Count of Step of Algorithm is : 4
------------------------------------------
------------------------------------------
Enroll Number: 142
Recursive Method
Fibonacci of Number 5 is: 5
Count of Step of Algorithm is : 15
------------------------------------------
```

**Analysis:**

| Value of N | Steps in iteration method | Steps in recursive method |
|---|---|---|
| 3 | 2 | 5 |
| 5 | 4 | 15 |
| 6 | 5 | 25 |
| 7 | 6 | 41 |
| 10 | 9 | 177 |
| 15 | 14 | 1973 |



**Conclusion:**

- The program contrasts iterative and naive recursive Fibonacci implementations.
- The iterative approach runs in O(n) time with O(1) space, updating two variables per step.
- The recursive version here has exponential time due to repeated subproblem calls, making it impractical for larger n.
- Recursive calls add overhead and can risk deep call stacks, while iteration avoids these costs.
- Step counts clearly show the iterative method scales predictably and efficiently.
- If recursion is desired, use memoization or dynamic programming to eliminate recomputation.