First Assignment

Intro :

in this assignment we will compare between iterative and recursive factorial functions to analyze performance and implementation differences.

Iterative function :

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| value | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
| time | 50 ms | 53 ms | 54 ms | 56 ms | 58 ms | 59 ms | 60 ms | 29 ms |

Recursive function:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| value | 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
| time | 58 ms | 60 ms | 64 ms | 65 ms | 66 ms | 60 ms | 54 ms | 29 ms |

Note :

Stack overflow errors can occur in both iterative and recursive factorial functions at n=40 due to either deep recursion or excessive stack memory usage.

Discuss :

In our findings, the iterative factorial function consistently increased in execution time with larger n values, reflecting its linear time complexity. However, the recursive approach initially showed increasing execution times, but beyond a certain point, it demonstrated a decrease.

In summary, iterative factorial functions maintain a steady increase in time with n growth, contrasting with recursive methods that initially rise but then decline , Efficiency considerations are paramount in choosing factorial computation strategies.

Iterative function code :

long long factorial(long long n)

{

long long result = 1;

for (int i = 1 ; i<=n ; i++)

{

result\*=i;

}

return result ;

}

Recursive function code :

long long factorial(long long n) {

if (n == 0 || n == 1) {

return 1;

}

return n \* factorial(n - 1);

}