Arab American University

# **Faculty of Information Technology and Engineering**

## **ALGORITHMS ANALYSIS AND DESIGN**

Factorial Calculation Report

**Introduction:** Factorial calculation is a fundamental mathematical operation that finds extensive application in various fields, including mathematics, computer science, and engineering. In this report, we explore the implementation of factorial computation in the C++ programming language. We discuss different approaches to implement factorial, analyze their efficiency, and provide insights into optimizing factorial calculations.

1.Recursive Approach:

long long recursive (int n)

{

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

return 1;

return n \* recursive(n-1);

}

2. Iterative Approach:

long long itrative (int n)

{

if (n==0)

return 1;

long long fct=1;

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

fct\*=i;

return fct;

}

Entire code:

#include<iostream>

#include <chrono>

using namespace std;

std::chrono::steady\_clock::time\_point getCurrentTime() {

return std::chrono::steady\_clock::now();

}

double getDuration(std::chrono::steady\_clock::time\_point start, std::chrono::steady\_clock::time\_point end) {

return std::chrono::duration\_cast<std::chrono::milliseconds>(end - start).count() / 1000.0;

}

long long itrative (int n)

{

if (n==0)

return 1;

long long fct=1;

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

fct\*=i;

return fct;

}

long long recursive (int n)

{

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

return 1;

return n \* recursive(n-1);

}

int main()

{

auto startTime = getCurrentTime();

//cout<<itrative(5)<<endl;

cout<<recursive(5)<<endl;

auto endTime = getCurrentTime();

double duration = getDuration(startTime, endTime);

std::cout << "Runtime: " << duration << " seconds" << std::endl;

return 0;

}

|  |  |  |
| --- | --- | --- |
| N | Recursive Time | Iterative Time |
| 5 | 0s | 0s |
| 10 | 0s | 0s |
| 20 | 0s | 0s |
| 40 | 0.001s | 0s |
| 45000 | Stack Overflow | 0.002s |
| Max Number can do using Long Long data type 20 | | |

Discussion :

In the begging we implement factorial code, what interesting When I calculate the factorial for n number, I realize when I Insert a small number in Recursive it take a less time than it take In Iterative .In some cases when my computer is compressed but in normal case there is no big difference until we insert a huge number for exp 43000 a slight difference appears in runtime .But we have a problem when we insert a big number in Recursive function stack overflow appears in contrary there is no stack overflow in Iterative .

Conclusion:

In this report, we explored various approaches to implement factorials in C++. We discussed the differences between two methods Recursive and Iterative in runtime and its relation with large numbers.

For small inputs, both recursive and iterative methods are suitable. In large number iterative method, the best choice.