ABD HABAYEB (202011230)

Itreative methods time

Value	Time
5	0 microseconds
40	0 microseconds
300	0 microseconds
1000	1 microseconds
10000	11 microseconds

Recrsion methods time

Value	Time
5	0 microseconds
40	0 microseconds
300	5 microseconds
1000	17 microseconds
10000	250 microseconds

Code (itreative)

```
#include <iostream>

#include <chrono>

long long factorial_iterative(int n) {

   if (n < 0) {

      throw std::invalid_argument("Factorial is not defined for negative numbers.");}

       long long result = 1;

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

       result *= i;   }

       return result;}

int main() {

      try {

      int n = 10000;

      // Measure execution time

      auto start_time = std::chrono::high_resolution_clock::now();

      long long result = factorial_iterative(n);

      auto end_time = std::chrono::high_resolution_clock::now();

      auto duration =
std::chrono::duration_cast<std::chrono::microseconds>(end_time - start_time);

      std::cout << "The factorial of " << n << " is: " << result << std::endl;

      std::cout << "Execution time: " << duration.count() << " microseconds" <<
std::endl;

   } catch (const std::invalid_argument &e) {

      std::cerr << "Error: " << e.what() << std::endl;

   }

   return 0;}
```

Code(recursion)

```
#include <iostream>

#include <chrono>

long long factorial_recursive(int n) {

    if (n < 0) {

        throw std::invalid_argument("Factorial is not defined for negative numbers."); }

    // Base case: factorial of 0 is 1

    if (n == 0) {

        return 1;}

    // Recursive case: n! = n * (n-1)!

    return n * factorial_recursive(n - 1);

}

int main() {

    try {

        int n = 10000;

        // Measure execution time

        auto start_time = std::chrono::high_resolution_clock::now();

        long long result = factorial_recursive(n);

        auto end_time = std::chrono::high_resolution_clock::now();

        auto duration =
std::chrono::duration_cast<std::chrono::microseconds>(end_time - start_time);

        std::cout << "The factorial of " << n << " is: " << result << std::endl;

        std::cout << "Execution time: " << duration.count() << " microseconds" <<
std::endl;

    } catch (const std::invalid_argument &e) {

        std::cerr << "Error: " << e.what() << std::endl;

    }

    return 0;

}
```

Discuss	From we have in our hand we found the iterative method are the best way to solve this function the execution time is less from the other method.
Conclusion	We notice here the excution time incress when we incress the value and the rucrsion is not the best way to do this function because the function is simple and we can solve it dirictily without recursion.