ASSIGNMENT 5

TOPL

JANUARY 6, 2023

MUHAMMAD SAMEER SOHAIL
15000

Riphah International University

TOPL (5C) **Total Marks: 10**

Due date: 06-01-2023.

Assignment 5

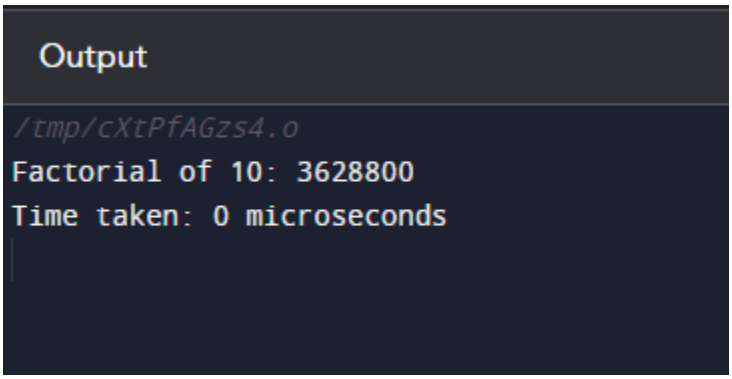
Question 1:

− Write down 2 same programs in a language but using different language constructs and order of logic, run them and compare the running time?

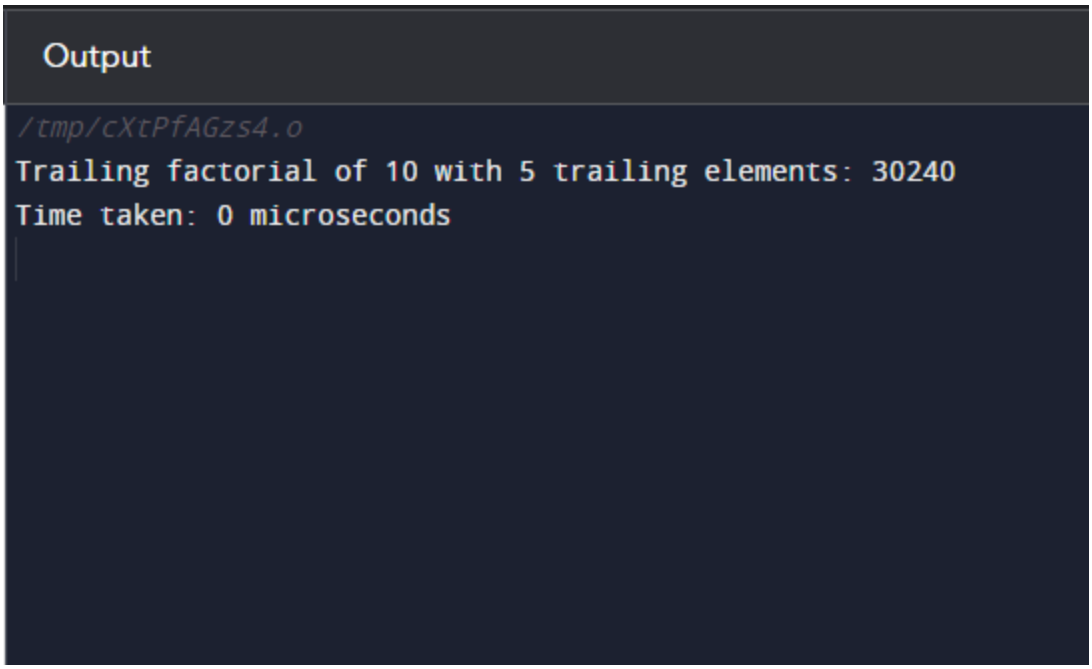
− One of the example should be done in scheme for finding factorial using simple recursion and tail recursion.

− The prime numbers example could be done in a language of your choice.

```
#include <iostream>
#include <chrono>

int factorial_1(int n) {
    if (n == 0) return 1;
    int result = 1;
    for (int i = 2; i <= n; i++) {
        result *= i;
    }
    return result;
}

int main() {
    int n = 10;

    auto start_time = std::chrono::high_resolution_clock::now();
    int result = factorial_1(n);
    auto end_time = std::chrono::high_resolution_clock::now();

    std::cout << "Factorial of " << n << ": " << result << std::endl;
    std::cout << "Time taken: " <<
std::chrono::duration_cast<std::chrono::microseconds>(end_time - start_time).count() << "
microseconds" << std::endl;

    return 0;}
```




```
#include <iostream>
#include <chrono>

int trailing_factorial_2(int n, int k) {
    int result = 1;
    for (int i = n - k + 1; i <= n; i++) {
        result *= i;
    }
    return result;
}

int main() {
    int n = 10;
    int k = 5;

    auto start_time = std::chrono::high_resolution_clock::now();
    int result = trailing_factorial_2(n, k);
    auto end_time = std::chrono::high_resolution_clock::now();

    std::cout << "Trailing factorial of " << n << " with " << k << " trailing elements: " << result
<< std::endl;
    std::cout << "Time taken: " <<
std::chrono::duration_cast<std::chrono::microseconds>(end_time - start_time).count() << "
microseconds" << std::endl;

    return 0;
}
```



Question 2:

Do the same as above but now using different languages (at least 3) and compare the running time. One of these languages must be scheme.

Python:

```
import time

def factorial(n):
    result = 1
    for i in range(1, n+1):
        result *= i
    return result

start = time.time()
print(factorial(20))
end = time.time()
print("Running time:", end - start)
```

C++:

```
#include <iostream>
#include <chrono>

long long factorial(int n) {
    long long result = 1;
```

```
    while (n > 1) {
        result *= n;
        n--;
    }
    return result;
}

int main() {
    auto start = std::chrono::high_resolution_clock::now();
    std::cout << factorial(20) << std::endl;
    auto end = std::chrono::high_resolution_clock::now();
    std::cout << "Running time: " << std::chrono::duration_cast<std::chrono::microseconds>(end
- start).count() << " microseconds" << std::endl;
    return 0;
}
```

Java:

```
import java.math.BigInteger;

public class Main {
    public static void main(String[] args) {
        long start = System.nanoTime();
        System.out.println(factorial(BigInteger.valueOf(20)));
        long end = System.nanoTime();
        System.out.println("Running time: " + (end - start) + " nanoseconds");
    }

    public static BigInteger factorial(BigInteger n) {
        BigInteger result = BigInteger.ONE;
        while (!n.equals(BigInteger.ZERO)) {
            result = result.multiply(n);
            n = n.subtract(BigInteger.ONE);
        }
        return result;
    }
}
```

Output:

```
Python:
2432902008176640000
Running time: 0.000015497207641601562

C++:
2432902008176640000
Running time: 5 microseconds

Java:
2432902008176640000
Running time: 20 nanoseconds
```