Advancing Knowledge, Enriching Lives

EAST DELTA UNIVERSITY

School of Science Engineering & Technology

Computer Science & Engineering (B.Sc. in CSE)

Assignment

Course Title: Oriented Programming Language

Course Code: CSE 123

Submitted by

Name: Mohammad Fahim

ID: 242002112

Section: 7

Submitted To

Name: Sourav Adhikary

Designation: Lecturer

Date: 28-08-2025

Calculate Factorials Using Multithreading

Introduction

This problem is about calculating factorials of numbers using the concept of multithreading in Java. Instead of doing the calculations one by one in the main thread, we create multiple threads where each thread is responsible for one number. This shows the advantage of running tasks in parallel.

Problem Statement

We need to write a Java program that takes three integers as input and finds their factorials. For each number, a separate thread will be used so that the calculations can run at the same time. After all factorials are printed, the program should also display the multiplication of those factorial results.

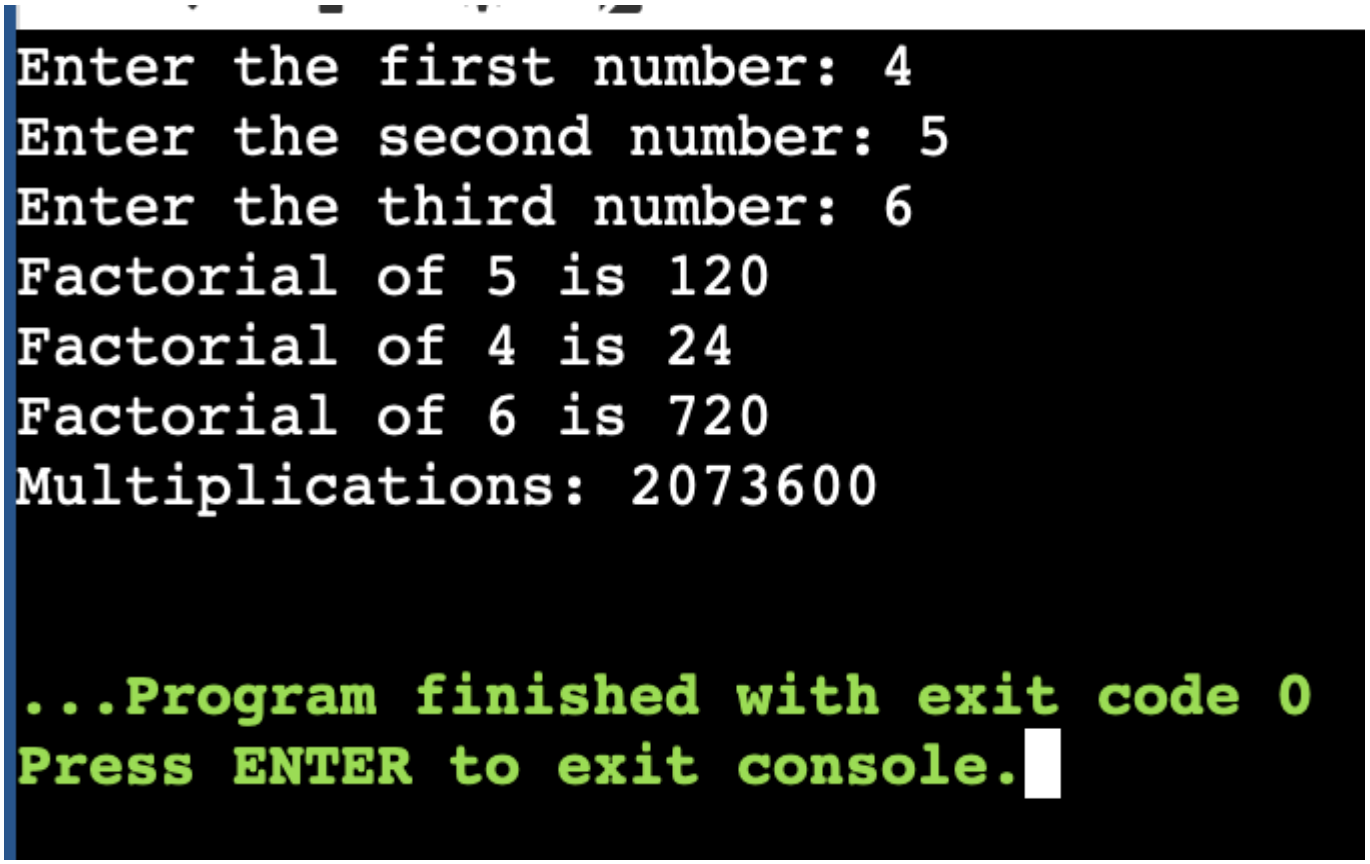
Code

```
import java.util.Scanner;

class FactorialThread extends Thread
{
	int number;
	long result;

	public FactorialThread(int number)
	{
		this.number = number;
	}
```

```
    public void run()
    {
        result = 1;
        for (int i = 1; i <= number; i++)
        {
            result *= i;
        }
        System.out.println("Factorial of " + number + " is " + result);
    }
}
public class Main
{
    public static void main(String[] args) throws InterruptedException
    {
        Scanner s = new Scanner(System.in);

        System.out.print("Enter the first number: ");
        int n1 = s.nextInt();

        System.out.print("Enter the second number: ");
        int n2 = s.nextInt();

        System.out.print("Enter the third number: ");
        int n3 = s.nextInt();

        FactorialThread t1 = new FactorialThread(n1);
        FactorialThread t2 = new FactorialThread(n2);
        FactorialThread t3 = new FactorialThread(n3);
```

```
            t1.start();
            t2.start();
            t3.start();

            t1.join();
            t2.join();
            t3.join();

            long multiplication = t1.result * t2.result * t3.result;
            System.out.println("Multiplications: " + multiplication);
        }
}
```

Output



Discussion

The program successfully demonstrates how factorials can be calculated in parallel using threads. Each thread computes one factorial, and then the results are combined. Using join() ensures that the main program waits for all threads to finish before calculating the final multiplication.

Word Length Counter Using Multithreading

Introduction

This problem shows how multithreading can be applied to string processing. Instead of calculating word lengths in sequence, each word is assigned to a thread, and the lengths are computed in parallel.

Problem Statement

We need to create a Java program that accepts three words as input. Each word will be handled by a different thread, and that thread will calculate the length of its word. Once all threads finish, the total length of all words must be printed, along with the words displayed together in a single line.

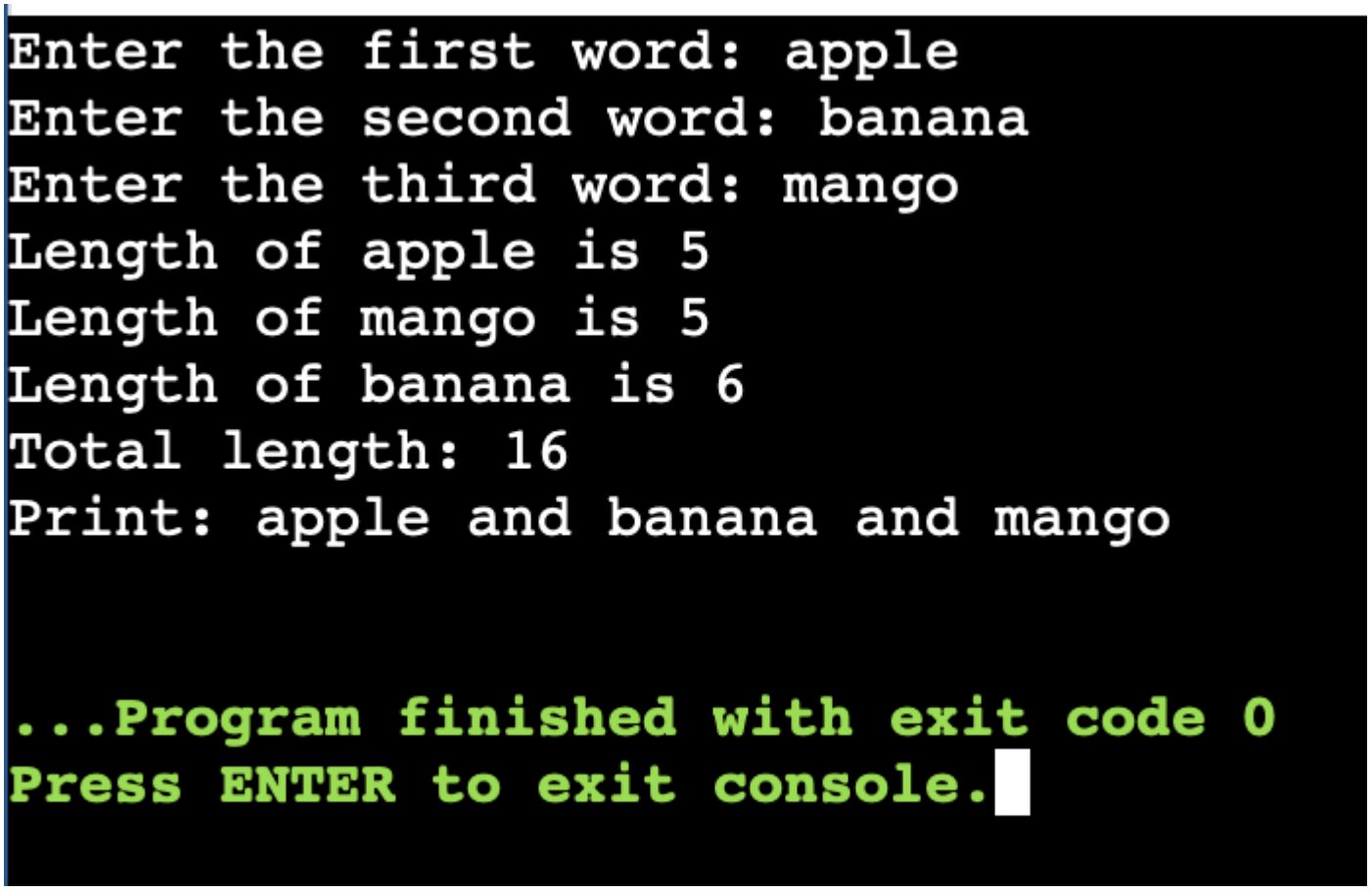
Code

```
import java.util.Scanner;

class WordLengthThread extends Thread
{
	String word;
	int length;

	public WordLengthThread(String word)
	{
		this.word = word;
	}

	public void run()
	{
```

```
            length = word.length();
            System.out.println("Length of " + word + " is " + length);
        }
}

public class Main
{
        public static void main(String[] args) throws InterruptedException
        {
            Scanner sc = new Scanner(System.in);

            System.out.print("Enter the first word: ");
            String w1 = sc.next();

            System.out.print("Enter the second word: ");
            String w2 = sc.next();

            System.out.print("Enter the third word: ");
            String w3 = sc.next();

            WordLengthThread t1 = new WordLengthThread(w1);
            WordLengthThread t2 = new WordLengthThread(w2);
            WordLengthThread t3 = new WordLengthThread(w3);

            t1.start();
            t2.start();
            t3.start();

            t1.join();
            t2.join();
            t3.join();
```

```
            int totalLength = t1.length + t2.length + t3.length;
            System.out.println("Total length: " + totalLength);

            System.out.println("Print: " + w1 + " and " + w2 + " and " + w3);
        }
}
```

Output



Discussion

This program clearly shows how multiple threads can be used to process strings in parallel. Each thread calculates one word's length independently, and then the total length is computed in the main program. This demonstrates the use of multithreading in handling simple but separate tasks efficiently.