Lab: Introduction to Parallel Programming with Java Threads

Objective:

The objective of this lab is to introduce students to parallel programming using Java threads. Students will implement a simple matrix multiplication program and parallelize the computation using threads to observe the impact on performance.

Prerequisites:

- Basic knowledge of Java programming.
- Understanding of matrix multiplication.

Lab Instructions:

1. Getting Started:

- Create a New Java Project:
- Start a new Java project in your preferred IDE (Eclipse, IntelliJ, etc.).

2. Task Overview:

2.1 Sequential Matrix Multiplication:

- Implement a Java program (MatrixMultiplier.java) that performs matrix multiplication sequentially without using threads.
- Generate random matrices, multiply them, and print the result.
- Measure and print the execution time.

2.2 Parallel Matrix Multiplication:

- Create a new Java class (ParallelMatrixMultiplier.java) to implement parallel matrix multiplication using Java threads.
- Divide the work among multiple threads to compute the result.
- Test the program with different numbers of threads (e.g., 2, 4, and 8).
- Measure and print the execution time for each thread count.

3. Code Implementation:

3.1 Sequential Matrix Multiplication:

- Implement the matrix multiplication algorithm without using threads.
- Use the standard matrix multiplication approach.
- You can refer to the sequential matrix multiplication in the provided example for guidance.

3.2 Parallel Matrix Multiplication:

- Implement the parallel matrix multiplication using Java threads.
- Divide the computation among multiple threads.
- Ensure proper synchronization and handling of thread-related issues.
- Experiment with different numbers of threads.

4. Performance Analysis:

- Compare the execution time of the sequential and parallel approaches.
- Discuss any performance improvements or trade-offs observed.
- Reflect on the impact of the number of threads on performance.

5. Report and Submission:

5.1 Lab Report:

- Prepare a lab report that includes the following:
- Introduction: Briefly explain the purpose of the lab.
- Code Explanation: Describe how matrix multiplication is implemented sequentially and in parallel.
- Your Implementation: Explain the design choices and challenges faced during parallel implementation.
- Performance Analysis: Discuss the observed performance differences between sequential and parallel approaches.
- Conclusion: Summarize the key learnings from the lab.

5.2 Submission:

Submit the code for both the sequential and parallel matrix multiplication classes (MatrixMultiplier.java and ParallelMatrixMultiplier.java) along with the lab report.