Advanced Programming (OOP) Lab

Assignment–6

Report Instructions

This document specifies the mandatory structure and assignment-specific questionnaires for the handwritten report corresponding to Assignment–6: Thread Management and Multithreading in Java. The report will be evaluated for understanding of concurrency concepts, synchronization issues, and correctness of concurrent program design.

General Submission Rules

- The report must be handwritten and submitted individually.
- Do not rewrite complete thread programs; include only selected and relevant code snippets.
- Explanations must focus on thread behavior, interaction, and safety.
- All implemented exercises and attempted bonus tasks must be reflected in the report.
- Superficial or copied explanations of threading concepts will be penalized.

Tip: *Use multiple colors for highlighting your explanations, references and annotations.*

Section A: Multithreading Motivation and Overview

Explain why multithreading is required in this assignment. Describe the problems that cannot be efficiently solved using a single-threaded approach. Identify the roles played by different threads in your implementation.

Section B: Thread Creation and Execution Model

Explain how threads are created and started in your program. Distinguish between implementing Runnable and extending Thread. Describe the lifecycle of a thread with reference to your code.

Section C: Shared Data and Synchronization Analysis

Using the shared counter or queue example:
- Identify the shared resources accessed by multiple threads
- Explain the race condition that may occur without synchronization
- Justify the use of synchronized methods or blocks

Section D: Timing, Sleeping, and Interruption

Explain the role of Thread.sleep() in your program. Describe what happens when a thread is interrupted during sleep. Explain why InterruptedException must be handled explicitly.

Section E: Executor Framework Reasoning

If ExecutorService was used:
- Explain how thread pools improve performance and resource management
- Describe task submission and execution flow
- Compare this approach with manual thread creation

Section F: Producer–Consumer or Advanced Synchronization

If you implemented Producer–Consumer, ReentrantLock, or wait/notify:
- Explain the coordination problem being solved
- Describe how deadlock and busy waiting are avoided
- Explain the correctness of the synchronization strategy

Section G: What-If Concurrency Analysis

Answer any two of the following:
1. What breaks if synchronization is removed?
2. How does performance change with increased thread count?
3. What issues arise if shared variables are not atomic?
4. How would behavior differ on a multi-core versus single-core processor?

Section H: Individual Extension Explanation

Describe one individual enhancement or variation you implemented, such as:
- Additional synchronization logic
- Visualization of thread states
- Use of Callable and Future

Explain how this extension affects thread interaction and correctness.

Section I: Debugging and Learning Reflection

Describe one concurrency-related issue you encountered. Explain how it was diagnosed and what deeper understanding of multithreading was gained.

Section J: AI Usage Disclosure

Declare whether AI or online tools were used. Clearly state:
- Which part of the concurrency logic was assisted
- How correctness and thread safety were verified