# Advanced Programming (OOP) Lab

# Assignment–4

## Report Instructions

This document specifies the mandatory structure and assignment-specific questionnaires for the handwritten report corresponding to Assignment–4: Exception Handling in Java. The report must demonstrate conceptual clarity regarding Java's exception mechanism, correct use of built-in and custom exceptions, and sound error-handling design.

### General Submission Rules

- The report must be handwritten and submitted individually.
- Do not reproduce complete source code; include only relevant fragments where required.
- Explanations must focus on exception flow and design intent, not syntax alone.
- Both Part 1 and Part 2 of the assignment must be covered in the report.
- Generic or copied exception descriptions will attract penalties.

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

### Section A: Exception Handling Overview

Explain the purpose of exception handling in Java. Distinguish between compile-time and runtime exceptions, and justify why exceptions are preferred over error codes.

### Section B: Built-in Exception Demonstration Analysis

For any four built-in exceptions demonstrated in Part 1:
1. Describe the cause of the exception in this program.
2. Identify the exact statement that may throw the exception.
3. Explain how the corresponding catch block handles the exception.
4. State one real-world situation where this exception may occur.

### Section C: try–catch and Control Flow Reasoning

Explain the execution flow of a try–catch block when an exception occurs. Discuss what happens to subsequent statements inside the try block. Include a small annotated code fragment to support your explanation.

### Section D: Custom Exception Design Justification

Explain the need for the custom InsufficientFundsException class. Answer the following:
1. Why is this exception better represented as a separate class?
2. Why does it extend Exception instead of RuntimeException?
3. What information should a well-designed custom exception convey?

## Section E: BankAccount Method-Level Reasoning

For the deposit() and withdraw() methods:
- Explain how input validation is performed
- Describe when and why exceptions are thrown
- Explain how exception propagation affects the calling method

## Section F: What-If and Robustness Analysis

Answer any two of the following:
1. What changes if exceptions are not caught in the main method?
2. What problems arise if negative deposits are silently ignored?
3. How does retry logic using loops improve program robustness?
4. What design issues arise if exceptions are replaced with return codes?

## Section G: Extension or Exercise Explanation

If you implemented any exercises (transfer, logging, retry mechanism):
- Explain the exception-handling strategy used
- Describe how failures are detected and reported
- Discuss how program reliability is improved

## Section H: Error, Debugging, and Learning Reflection

Describe one difficulty encountered while implementing exception handling. Explain how the issue was diagnosed and what conceptual understanding improved as a result.

## Section I: AI Usage Disclosure

Declare whether AI or online tools were used. Specify:
- Which part of the exception logic was assisted
- How you verified correctness and adapted the solution