Assignment 3 (CAT 3) – Java Programming

Topic Focus: Exception Handling and Java I/O System
Weight: 50% of Continuous Assessment
Open Book | Time Allocation: 1 Week | Submission via Moodle

PART A: THEORY – 20 MARKS

Answer all questions. Use clear language and apply concepts to real-world examples where necessary.

1. Explain the importance of exception handling in large-scale systems such as NHIF claims processing or university admissions portals. What problems can arise without it? *(5 marks)*
2. Distinguish between checked and unchecked exceptions in Java. Give two examples of each, and explain when each would typically occur in systems like online banking or student information systems. *(5 marks)*
3. Describe the purpose and use of the finally block in Java exception handling. Explain how this would be useful in a library management system that writes logs to a file. *(5 marks)*
4. A file-based system must read student records from a .txt file and process them for graduation eligibility. What potential exceptions should a developer prepare for? How would they handle these gracefully in code? *(5 marks)*

PART B: PRACTICAL – 30 MARKS

Write and test your code using NetBeans. Submit working .java files and annotated screenshots.

TASK 1: Loan Repayment Logger – Kenya Women Microfinance Simulation (15 marks)

Scenario: A loan officer at Kenya Women Microfinance wants to record the names and loan balances of five clients into a text file. The file must be created, written to, and properly closed, handling any file-related exceptions.

Instructions:

- Create a program that stores names and balances in arrays.
- Write this data to a text file (loan_records.txt) using BufferedWriter.
- Include exception handling to manage file creation and write errors.
- Display a message:
  Data successfully written for 5 clients.
  Or catch and report errors such as IOException.

TASK 2: Student Marks Reader with Exception Handling (10 marks)

Scenario: A university department wants to read a file containing student names and marks to calculate their average and grade. The system must skip any malformed or missing entries without crashing.

Instructions:

- Create a file marks.txt with 5 entries (e.g., John 78, Mary 65, David 91).
- Read each line using BufferedReader.
- Split each line and convert the mark to int.
- Use try-catch to handle:
  - Missing values
  - Number format errors
- Print the average and total students processed.

TASK 3: Custom Exception for Fee Clearance (5 marks)

Scenario: A student should only register for graduation if their balance is zero. Create a custom exception called FeeBalanceException.

Instructions:

- Write a method checkClearance(double balance) that throws FeeBalanceException if the balance is greater than 0.
- In main(), simulate checking three students with varying balances.
- Display "Cleared for Graduation" or "FeeBalanceException: Pending fees" accordingly.

MOODLE SUBMISSION INSTRUCTIONS

Upload a .zip file named:

CAT3_JavaProgramming_YourRegNo.zip

Your submission must include:

1. A typed document (Word or PDF), titled Theory_CAT3_YourRegNo:
   - Responses to all 4 theory questions (Part A), clearly numbered
   - A brief reflection (optional, max 5 lines) on what you learned
2. All Java source code files (.java) for the practical section

    - Use clear and descriptive file names (e.g., LoanLogger.java, MarksReader.java, FeeClearance.java)

3. Screenshots of program output from NetBeans
    - Label each screenshot for the corresponding task