Java Custom Exception Handling - Scenario Based Assignments

Assignment 1 – ATM PIN Validation

Scenario: You are building an ATM machine simulation. The user must enter a correct 4-digit PIN. If the PIN is invalid (wrong or not 4 digits), throw a InvalidPinException.

Requirements:

- Create InvalidPinException extending Exception.
- Create ATM class with a constant valid PIN and a validatePin(String enteredPin) method.
- Throw exception if entered PIN is incorrect or not exactly 4 digits.
- In Main class, ask user to enter PIN and handle exception with friendly message.

Expected Output:

- Enter PIN: 123 → Error: PIN must be exactly 4 digits.
- Enter PIN: 4321 → Error: Incorrect PIN.
- Enter PIN: 1234 → Access Granted. Welcome!

Learning Outcome:

- Create a checked custom exception.
- Validate user input and provide meaningful error messages.
- Understand exception flow in authentication scenarios.

Assignment 2 – Online Movie Ticket Booking

Scenario: In an online ticket booking system, a user cannot book more than 6 tickets at once. If they try to exceed this limit, throw a TicketLimitExceededException.

Requirements:

- Create TicketLimitExceededException extending RuntimeException.
- In TicketBooking class, create bookTickets(String movieName, int quantity) method.
- Throw exception if quantity > 6.
- In Main class, try booking different quantities and handle errors.

Expected Output:

- Booking 4 tickets for Avengers... → Booking successful!
- Booking 7 tickets for Oppenheimer... → Error: Cannot book more than 6 tickets at once.

Learning Outcome:

- Create an unchecked custom exception.
- Apply business rule validation in service logic.
- Understand when to use RuntimeException vs Exception.

Assignment 3 – File Upload Size Validation

Scenario: A file cannot be more than 25 MB. If it exceeds the limit, throw a FileTooLargeException.

Requirements:

- Create FileTooLargeException extending Exception.
- In FileUploader class, create uploadFile(String fileName, double fileSizeMB) method.
- Throw exception if file size > 25.
- In Main class, upload files and handle exceptions.

Expected Output:

- Uploading report.pdf (12 MB)... → File uploaded successfully.
- Uploading movie.mp4 (30 MB)... → Error: File size exceeds the 25 MB limit.

Learning Outcome:

- Implement size-based validations with exceptions.

- Throw exceptions from a business logic layer.
- Understand checked exceptions for predictable validation errors.

Assignment 4 – Ecommerce Order Processing

Scenario: A customer’s order must contain at least 1 item. If quantity is zero or less, throw InvalidOrderQuantityException.

Requirements:

- Create InvalidOrderQuantityException extending RuntimeException.
- In OrderService class, create placeOrder(String productName, int quantity) method.
- Throw exception if quantity <= 0.
- In Main class, place both valid and invalid orders.

Expected Output:

- Placing order for Laptop (Quantity: 1)... → Order placed successfully.
- Placing order for Phone (Quantity: 0)... → Error: Order quantity must be greater than zero.

Learning Outcome:

- Throw unchecked exceptions for invalid business inputs.
- Use exception messages effectively.
- Practice defensive programming in service methods.

Assignment 5 – University Exam Eligibility

Scenario: A student can take an exam only if attendance is >= 75%. If less, throw LowAttendanceException.

Requirements:

- Create LowAttendanceException extending Exception.
- In ExamEligibility class, create checkEligibility(String studentName, double attendancePercent) method.
- Throw exception if attendance < 75.
- In Main class, check eligibility for multiple students.

Expected Output:

- Checking eligibility for Alice (Attendance: 80%)... → Eligible for exam.
- Checking eligibility for Bob (Attendance: 65%)... → Error: Attendance below 75%. Not eligible for exam.

Learning Outcome:

- Implement rule-based validation using checked exceptions.
- Handle exceptions in decision-making scenarios.
- Understand domain-specific exception naming and usage.