CS23304 JAVA PROGRAMMING

Course Instructor: V P Jayachitra

WEEK 8

Instruction:

- **Use meaningful variable names**
- **Consistent indentation**
- **Proper error handling**
- **Proper comment to follow the question requirement**

Mathematical Calculator with Runtime Exceptions

1. Create a MathCalculator class that includes:

- calculatePower(int base, int exponent) which throws:
    - IllegalArgumentException if base is 0 and exponent is negative
    - ArithmeticException if both base and exponent are negative
    - UnsupportedOperationException if both base and exponent are zero
    - Otherwise returns the result using Math.pow()
- factorial(int n) which throws:
    - IllegalArgumentException if n is negative
    - ArithmeticException if n > 20 (overflow)
    - Otherwise returns factorial calculated iteratively
- safeDivide(double dividend, double divisor) which throws:
    - ArithmeticException if divisor is 0.0
    - IllegalArgumentException if both dividend and divisor are 0.0
    - Otherwise returns division result

Test Cases:

- Normal: calculatePower r(2,3)→8.0, factorial(5)→120, safeDivide(10,2)→5.0
- Exceptions: calculatePower (0,-2), calculatePower(-3,-2), calculatePower (0,0), factorial(-5), factorial(25), safeDivide(5,0), safeDivide(0,0)

Banking System with Custom Exceptions

2. Design a banking system with:

- Custom exceptions:
    - InsufficientFundsException with extra info on balance and shortfall
    - InvalidAmountException for zero or negative amounts
- BankAccount class with:
    - Constructor validating accountId (non-null) and balance (non-negative)
    - deposit(double amount), withdraw(double amount), transfer(BankAccount target, double amount) methods
    - Appropriate exceptions thrown on invalid operations and handled safely

Test Cases:

- Normal operations: deposit(100), withdraw(50), transfer(account, 200)
- Exception scenarios: withdraw(5000) from balance 1000,

3. Create a file processing system showing checked exception propagation through method calls.
4. Build a calculator demonstrating unchecked exceptions (like NumberFormatException, ArithmeticException) propagating without declaring throws.
5. Write an example with method overriding illustrating rules when superclass method throws checked exceptions and overridden method throws unchecked exceptions.