Lab: Fantasy Creature Management System

Objective:

In this lab, you will create a simple system to manage various types of fantasy creatures. This lab will test your understanding of Java inheritance, method overriding, polymorphism, checked and unchecked exceptions, custom exceptions, substitution, the instanceof operator, and the .getClass() method. By the end of the lab, you should be comfortable applying these key OOP concepts to a project.

With a partner, create a new project named "comp2522 lab 2" in JetBrains IDEA:

1. File / New / Project
2. Java / Next / Next
3. Project name: "comp2522 lab 2"
4. Finish
5. File / Project Structure / Modules
6. Create new subfolder under src / code
7. Create new subfolder under src / tests
8. OK
9. Right-click the src/code folder / New / Java Class (then make the classes described below).

Step 1: Fantasy Creature Class Hierarchy

Superclass: Creature

Instance Variables:

- name: String (must not be null or empty).
- dateOfBirth: Date (must not be in the future).
- health: int (Must be 1 - 100)

Methods:

- Constructor Creature(String name, Date dateOfBirth, int health) (throw IllegalArgumentException if parameters are invalid).
- isAlive(): Returns true if health is greater than 0.Do not use magic numbers!
- takeDamage(int damage): Reduces health by damage. If health goes below 0, set it to 0. If damage is negative, throw an unchecked DamageException.
- heal(int healAmount): Increases health by healAmount but cannot exceed 100. If healing amount is negative, throw an unchecked HealingException.
- getAgeYears(): Calculates the creature's age in years based on its date of birth.
- getDetails(): Prints the creature's name, dateOfBirth, age in years, and health.

Subclass: Dragon (extends Creature)

Instance Variable:

- firePower: int (must be between 0 and 100).

Methods:

- Constructor Dragon(String name, Date dateOfBirth, int health, int firePower) (throw IllegalArgumentException if parameters are invalid).
- Override getDetails(): Add firePower to the output.
- breatheFire(): Reduces firePower by 10 and deals 20 damage to another creature. If firePower is less than 10, throw a checked LowFirePowerException.
- restoreFirePower(int amount): Increases firePower but cannot exceed 100.)

Subclass: Elf (extends Creature)

Instance Variables:

- mana: int (must be between 0 and 50).

Methods:

- Constructor Elf(String name, Date dateOfBirth, int health, int mana) (throw IllegalArgumentException if parameters are invalid).
- Override getDetails(): Add mana to the output.
- castSpell(): Reduces mana by 5 and deals 10 damage to a creature. If mana is less than 5, throw a checked LowManaException.
- restoreMana(int amount): Increases mana but cannot exceed 50.

Subclass: Orc (extends Creature)

Instance Variables:

- rage: int (must be between 0 and 30).

Methods:

- Constructor Orc(String name, Date dateOfBirth, int health, int rage) (throw IllegalArgumentException if parameters are invalid).
- Override getDetails(): Add rage to the output.
- berserk(): Increases rage by 5. If rage exceeds 20, deal double damage (30 damage) to a creature. If rage is below 5, throw an unchecked LowRageException.

Step 2: Exception Handling

Unchecked Exceptions:

- DamageException: Thrown when an invalid damage amount is applied.
- HealingException: Thrown when an invalid healing amount is applied.
- LowRageException: Thrown when an Orc cannot go berserk due to insufficient rage.

Checked Exceptions:

- LowFirePowerException: Thrown when a Dragon tries to breathe fire with insufficient firePower.
- LowManaException: Thrown when an Elf tries to cast a spell with insufficient mana.

Step 3: Testing with Polymorphism

Create a test class in the src/tests folder called CreatureTest.

In main(), create individual Creature objects that include:

- One Dragon.

- One Elf.

- One Orc.

Use substitution (polymorphism) to:

- Call getDetails() for each creature.

- Use the instanceof operator and .getClass() to determine the exact class of each object during runtime.

Use exception handling to ensure no unchecked exceptions crash the program. Catch and print a friendly message for each.

- Try to make the creatures fight each other using breatheFire(), castSpell(), and berserk().

Submission:

1. Make sure to include:

- Proper JavaDoc comments.

- Test cases demonstrating polymorphism, instanceof, and .getClass() usage.