**Walkthrough Guide for Developing the Java Program: AnimalArena**

In this walkthrough guide, we will explain how to develop the **AnimalArena** Java program step by step. This program simulates a fight between two animals with randomly assigned strengths and health. It includes the **Animal** class and methods to create and simulate a fight between animals.

**Table of Contents**

1. Understanding the Program Structure
2. Creating the Animal Class
3. Creating Animals in the **AnimalArena** Class
4. Implementing the Fight Simulation
5. Running the Program

**1. Understanding the Program Structure**

The **AnimalArena** program consists of two classes: **AnimalArena** and **Animal**. The **Animal** class represents an animal with attributes such as type, strength, and health. The **AnimalArena** class contains the **main** method and methods for creating animals and simulating fights between them.

**2. Creating the Animal Class**

The **Animal** class is a simple class that has private fields for type, strength, and health, along with getters and setters for these fields. Additionally, it overrides the **toString** method to provide a meaningful string representation of an animal object.

**3. Creating Animals in the AnimalArena Class**

Create an Animal using Default Constructor and Mutators

The **CreateAnimalUsingDefaultConstructorAndMutators** method in the **AnimalArena** class creates an animal using the default constructor and mutator methods to set its properties (type, strength, and health) randomly.

Create an Animal using Parameterized Constructor

The **CreateAnimalWithParameterizedConstructor** method creates an animal using a parameterized constructor, where you can specify the type, strength, and health of the animal.

**4. Implementing the Fight Simulation**

The **Fight** method simulates a fight between two animals until one of them has zero or less health. It uses the **animalAttack** method to calculate and display damage inflicted during each attack.

**animalAttack** Method

The **animalAttack** method calculates a random attack value, updates the defender's health, and displays the attack details.

**pause** Method

The **pause** method is a helper method that introduces a delay in the program to control the timing of console output.

**5. Running the Program**

The **main** method in the **AnimalArena** class demonstrates how to create two animals and simulate a fight between them. You can run the program to observe the outcome of the battle.

Now, let's move on to the code explanations and code segments for each part of the program.

**1. Creating the Animal Class**

public class Animal { private String type; private int strength; private int health; // Constructors, getters, setters, and toString method are defined here }

**2. Creating Animals in the AnimalArena Class**

public class AnimalArena {

public static Animal CreateAnimalUsingDefaultConstructorAndMutators(){

// Implementation of creating an animal using default constructor and mutators

}

public static Animal CreateAnimalWithParameterizedConstructor(){

// Implementation of creating an animal using parameterized constructor }

}

**3. Implementing the Fight Simulation**

public class AnimalArena {

public static void Fight(Animal a1, Animal a2){

// Implementation of simulating the fight between two animals

}

public static void animalAttack(Animal attacker, Animal defender){

// Implementation of calculating and displaying attack details

}

public static void pause(int millis){

// Implementation of the pause method to control timing

}

}

**4. Running the Program**

public class AnimalArena { public static void main(String[] args) {

Animal liz = CreateAnimalUsingDefaultConstructorAndMutators();

Animal dog = CreateAnimalWithParameterizedConstructor(); System.out.println(liz); System.out.println(dog); pause(2000); Fight(liz, dog); }

}

You can compile and run the **AnimalArena** program to see how it works. It will create two animals and simulate a fight between them, displaying the outcome in the console.

Top of Form