public class Fish {

// Instance variables

private String typeOfFish;

private int friendliness;

// Default constructor

public Fish() {

this.typeOfFish = "Unknown";

this.friendliness = 3; // Generic friendliness

}

// Parameterized constructor

public Fish(String t, int f) {

this.typeOfFish = t;

this.friendliness = f;

}

// Method to get friendliness

public int getFriendliness() {

return friendliness;

}

}

public class Fish {

// Existing code...

// Method to determine the nicest fish from a variable number of fish

public static Fish nicestFish(Fish... fishArray) {

if (fishArray.length == 0) {

return null; // No fish to compare

}

Fish temp = fishArray[0];

for (Fish fish : fishArray) {

if (fish.getFriendliness() > temp.getFriendliness()) {

temp = fish;

}

}

return temp;

}

}

8. public class Fish {

// Existing code...

// Method to determine the nicest fish from a variable number of fish

public static Fish nicestFish(Fish... fishArray) {

if (fishArray.length == 0) {

return null; // No fish to compare

}

Fish temp = fishArray[0];

for (Fish fish : fishArray) {

if (fish.getFriendliness() > temp.getFriendliness()) {

temp = fish;

}

}

return temp;

}

}

9. public class Main {

public static void main(String[] args) {

// Creating fish instances

Fish fish1 = new Fish("AngelFish", 5);

Fish fish2 = new Fish("Guppy", 3);

// Testing the nicestFish method

Fish nicest = Fish.nicestFish(fish1, fish2);

System.out.println("The nicest fish is of type: " + nicest.typeOfFish);

}

}

10. public class Employee {

private String name;

private String address;

private double salary;

private String phoneNumber;

// Getters and setters...

}