Alperen UGUS
CWID: 10864101

11/25/19

Software Engineering C25A

Team members: I did this assignment solo.

Why the code violates OCP:

The problem is that PetStore class has all those functions (sounds(), treats(), fun()) implemented. They should be abstract and there should be 2 additional classes named Cat and Dog which implements those functions since if anyone wants to add more animals, they need to change PetStore class which we don't want a class to be changed after testing it.

Modified Code:

Pet Class

```java
public abstract class Pet {
    public String petName;

    public Pet(String petName) {
        this.petName = petName;
    }

    public abstract void sounds();
    public abstract void treats();
    public abstract void fun();
}
```

For the Pet class, I made it abstract. Since we will add a new class for each animal, we do not need to have a petType class variable for each object. I made sounds(), treats() and fun() abstract because each Pet should implement it inside its own class.

PetStore Class

```java
import java.util.ArrayList;

public class PetStore {
    public ArrayList<Pet> pets = new ArrayList<Pet>();

    public void sounds() {
        for (Pet pet : pets) {
            pet.sounds();
        }
    }

    public void treats() {
        for (Pet pet : pets) {
            pet.treats();
        }
    }

    public void fun() {
        for (Pet pet : pets) {
            pet.fun();
        }
    }

    public static void main(String[] args) {

    }

}
```

In this class, we do not need to check the petType every time we call those functions since all of the objects from different classes implements their own methods.

Cat Class

```
public class Cat extends Pet{

    public Cat(String petName) {
        super(petName);

    }

    public void sounds() {
        System.out.println("Meow");
    }

    public void treats() {
        System.out.println("Give " + petName + " some catnip");
    }

    public void fun() {
        System.out.println("Watch " + petName + " sleep");
    }

}
```

This class extends Pet class and implements the methods for itself according to its requirements.

Dog Class

```
public class Dog extends Pet{

    public Dog(String petName) {
        super(petName);
    }

    public void sounds() {
        System.out.println("Woof");
    }

    public void treats() {
        System.out.println("Give " + petName + " a bone");
    }

    public void fun() {
        System.out.println("Throw a frisbee to " + petName);
    }

}
```

This class is the same as Cat class.

How my revised code adheres to OCP:

Previously, if one wanted to add an animal to the program, he/she needs to change PetStore class which would have been tested so according to OCP, we do not want any tested class to be changed and re-tested again and again. Creating a new class for each variable prevents this. So, now we do not need to change PetStore and test it again but instead, we can test the new Cat and Dog classes.