Part I: Documenting a Pet Class

☐ Create a New Project named Lab8, and a New Class called Pet.

☐ You are the new (replacement) programmer for a local pet store, and your predecessor had written some code, but it was unfinished. Download the two files provided for this lab from the course page: Pet.java and PetsTester.java.

☐ Open up the Pet.java file first.

☐ Sigh... you can already see why your predecessor was fired. He didn’t care to document the instance variables or methods! For each instance variable and method (including the constructors), provide Javadoc-style comments (see lecture notes for syntax). I’ll get a couple started for you:

```java
private String name;    /** The Pet’s given name */

// (code omitted)

/**
 * Constructor. Creates a Pet object given its name only.
 * Initializes age and weight both to 0.
 * @param name The given name for the Pet
 */
public Pet(String name)
{
    this.name = name;
    this.age = 0;
    this.weight = 0;
}
```

☐ Your documentation should be descriptive of every method’s behavior. Also make sure you describe each of the method’s parameters using @param and its return value, using @return.

☐ Fill in my documentation, and then click on selection box on the top-right hand corner of BlueJ’s source editor screen and change it to read “Documentation.” Make sure that the documentation has been updated.
Part II: Exploring the Pet Class

- After properly documenting, compile the Pet class.
- Return to BlueJ’s Project screen.
- Right-click on Pet to bring up its menu. Notice that the only methods available are the constructors.

![Image of Pet Menu](image1)

Figure 1: Opening the Pet Menu

- Select the new Pet() constructor, and BlueJ will ask you for a name for the instance. This instance name is equivalent to the variable name you would give it in code. Leave it as pet1 and click OK.
- You will now see that a Pet object has been created as a red box on the bottom left of your screen. Right click on this object to bring up its instance methods menu.

![Image of Pet Object Methods](image2)

Figure 2: Opening the Pet Object Menu

- Open up the README.txt file from BlueJ and answer the following questions.
  - **Question 0**: Call the toString() method on pet1. What is its output?
  - **Question 1**: Use the mutator methods to set the Pet’s name to Fluffy, its weight to 10, and its age to 2. What does toString() return now?
- **Question 2**: Create another Pet object, called pet2 by using the proper constructor. You know the pet’s name is Fido, its age is 10, and its weight is 20. This time, double-click on pet2’s object box on the bottom left. Explain what is displayed. Remove the pet1 and pet2 objects by right clicking them and selecting Remove.

- **Question 3**: Now go into the Pet class code and remove the this keyword from the setName() method:

```java
public void setName(String name)
{
    name = name;
}
```

Go back into the Project screen, and create another Pet object with the name “Fluffy,” age 5, and weighs 13 pounds. Now use the setName() method you just modified to rename this pet to “Hoppy.” Inspect the object by double-clicking on it (or by calling toString()). What is the value of the name field. Describe why you think this behavior occurred. Revert the code back to the original state.

- **Question 4**: Open up the PetsTester class. The only method in here in main(), which creates a Pet, who is named Furdi, is 4 years old, and 12 pounds. Try to set or print out this pet’s fields directly:

```java
Pet furdi = new Pet("Furdi", 4, 12);
furdi.age = furdi.age + 1;  // furdi now 5 years old?
System.out.println(furdi.name + " is " + furdi.age + " years old!");  //←
    what is furdi age now?
System.out.println(furdi.name + " weighs " + furdi.weight + " lbs!");  //←
    what does furdi weigh?
```

Try compiling this code. Write down what happens.

- **Question 5**: Now go back into the Pet code, and change the keyword private to public for all three fields:

```java
public String name;
public int age;
public double weight;
```

Re-run the program you created for Question 4. What is output now?

- **Question 6** From the previous two questions, what can you now infer about the uses of access specifiers public versus private? Do some Googling: why is it good practice to set all fields private, and only use accessor and mutator methods to get and set the values?

- **Question 7**: Revert all three fields back to private. Fix the code given in Question 4 to use the accessor and mutator methods for age, name, and weight. Your code should now print:

```
Furdi is 5 years old!
Furdi weighs 12 lbs!
```
Part III: Adding to the Pet Class

- For each of the following, provide the Javadoc comments!

- Add a new private instance variable called `type` to the `Pet` class, that indicates whether the Pet is either a cat or a dog.

- You learned in class that for cases like this, we might want to introduce some integer `constants` to denote cat and dog types, respectively. Do not use a string for the `type`.

- Modify all constructors. They all now need to include the pet’s `type` (i.e., cat or dog). The no-args constructor no longer makes sense, because every pet must now have a type, and it’s nonsensical to assign a pet some default type! Change it so that the no-args constructor now takes a single parameter, the pet’s type, and there will no longer be a no-args constructor.

- Add an accessor method `getType()`. You do not need to add a mutator method `setType()`, because it is impossible for a Pet to change from a cat to a dog after it has been created!

- The `toString()` method should also return the pet’s type.

- Add a method `benadryl()` that returns the dosage amount for the antihistamine, Benadryl. For cats, the dosage is 0.5mg per pound, and for dogs, the dosage is 2.0mg per pound.

- Add a method `amoxicillin()` that returns the dosage amount for the antibiotic, Amoxicillin. For cats, the dosage is 5mg per pound, and for dogs, the dosage is 10mg per pound.

- Back in the `PetsTester` class, modify the `main()` method to include tests of these new methods.

```java
Pet furdi = new Pet("Furdi", 4, 12, Pet.CAT);

//how much to dose Furdi?
System.out.println("Amoxicillin Dosage: "+furdi.amoxicillin());
System.out.println("Benadryl Dosage: " + furdi.benadryl());
```

The output from the code above should be:

Amoxicillin Dosage: 60.0
Benadryl Dosage: 6.0

Part IV: Submitting Your Lab

After you have completed the lab, please do the following to submit your work.

- Zip up the README.txt and the updated .java files.
- Rename the zip file to `LastName1_LastName2_Lab8.zip`, where the LastNames are the respective last names of the authors.
- Submit it on Moodle. You may submit as often as you’d like before the deadline. I will grade the most recent copy.