IST 256
Lab Week 10, Thursday, March 29, 2012

1. Writing a program that experiments with a class

In this program, we will write a Java application that contains a class for keeping pet information and recommending how much food to give a pet. The main program will experiment with creating and using objects of the pet class.

a. Start by **creating a Java application** and name it something like TestPet. As usual, NetBeans will bring up a program source code window where there is a class called Main and a main method where we will put our code to read a file.

b. Create a Pet class: In the left pane of NetBeans, find the TestPet project and right underneath the line that says "Source Packages", click on the line that says the package "testpet". In the menu, select New -> Java class. In the new class window:

give the **class name** as Pet
(the **package** of the class should already be testpet)

Note that in the left pane, there are now two programs listed under the Source Packages/testpet – Main.java and Pet.java.

c. Write the Pet class.

After the class header and inside the curly brackets for this class, we will add all the code for the class. First add the fields of the class. We will declare two variables as fields for the name of the pet and its weight in pounds. For the sake of demonstration, we make the fields public so that they can be used outside the class. Note that it is far more common to make these variables private.

```
// fields for this class are the name and weight (in pounds)
   public String name;
   public int weight;
```

After this code in the Pet class, we will add a Constructor method.

```
// constructor initializes both fields
   public Pet(String startname, int startweight)
   {
      name = startname;
      weight = startweight;
   }
```

Next, we will add a method to change the weight of the pet. Note that this method can just assign a new value to one of the fields.

```
// method to change the weight of the pet
   public void setWeight(int newweight)
   {
      weight = newweight;
   }
```

And finally, we had another method that can be used to find a recommended amount of food to feed this pet on any day. To find the amount of food, you must pass a string representing a level of activity of the pet on that day.

```
      // method to recommend how much pet food to give your pet
         //    depending on the day's activity level
         //    levels are "low", "normal", "high"
         public double recommendFood(String activityLevel)
         {
            // initialize food to 0 ounces
            double food = 0;
            if (activityLevel.equals("low"))
            {
               food = weight * 2.0;
            }
            if (activityLevel.equals("medium"))
            {
               food = weight * 2.2;
            }
            if (activityLevel.equals("high"))
            {
               food = weight * 2.4;
            }
            return food;
         }
```

Note that this method uses the weight variable from the fields of the class.

d. Write the main method

Double click on the Main.java program and add code to test the class. First, we create two instances of the class Pet, called myPet and yourPet, and we print out the fields of them both. Type the following code into the main method of the Main class, right after the TODO comment. But you may pick your own pet names and weights.

```
      // variables for two pets
            Pet myPet, yourPet;
            // amount of food
            double amount;

            // create an instance of Pet for my pet
```

```
        myPet = new Pet("Tiger", 20);

        // show values of my pet
        System.out.println("My pet name is " + myPet.name +
              " and weight is " + myPet.weight);

        //create another instance of Pet for your pet
        yourPet = new Pet("Fluffy", 10);

        // show values of your pet
        System.out.println("Your pet name is " + yourPet.name +
              " and weight is " + yourPet.weight);
        System.out.println();
```

Note the use of myPet.name and yourPet.name, for example, to show the values of the name fields in the two class instances. This is possible because the variables are public.

Run the program and observe its results in the output pane of the NetBeans Window.

Continue by using the methods setWeight and recommendFood. Here are some examples to try:

```
    // recommended food if my pet activity is medium
        amount = myPet.recommendFood("medium");
        System.out.println("With medium activity, pet food is " + amount);

        // My pet loses weight
        myPet.setWeight(18);
        // recommended food if my pet activity is medium
        amount = myPet.recommendFood("medium");
        System.out.println("With medium activity, pet food is " + amount);
```

Try using different values for the pet weight and activity level and view the results. Write here one test case, giving the call to setWeight and recommendFood, and showing the result.

2. Extending the Pet example

a. Add a field to the Pet class to hold the pet's type. This will be a String and can be used to simple hold a general string, like "cat" or "dog", or it could more specifically hold the breed, "Great Dane". Make this field private.

b. Add a method called setPetType to the Pet class that will set a value for the pet type field to the pet.

c. Add a method called toString() to the Pet class that returns a string with all the field values, name, weight and type.

d. In your main program,

- create a third pet with a name and a weight,
- call the setPetType method to add a type,
- call the toString method to get a String with all the fields,
- print that string using System.out.println

To submit this lab, print two things: your main program and your Pet class.
Submit this lab by Tuesday, April 3.