CSE 2102: Introduction to Software Engineering
Lab #4: Feb. 22, 2022
Objects & Classes

Note: Lab assignments are intended for practice. These will not be graded, and need not be submitted.

A. Creating Objects & Classes

Objects in a program can represent either objects in the real word— like automobiles, houses, and employee records—or abstractions like colors, shapes, and words. A class is the definition of a kind of object. It is like a plan or a blueprint for constructing specific objects. Consider a Pet class, with three attributes:

- Name of the pet
- Age of the pet
- Weight of the pet

As we learned in class, these three attributes represent private data of an object that is created from a class. Hence, these three variables must be declared as "private" in the definition of the Pet class. Corresponding to each variable that is declared private, there must be get and set methods. The get method retrieves the value currently stored in the variable, and the set method modifies or changes the value stored in the variable.

Once we have defined a class, the next question is how to create objects of the class. The "new" operator (which we have seen so far for system-defined classes) is used for this purpose. A constructor is a special method that is called when we use the "new" operator to create an object. The constructor initializes the instance variables to default values, or those supplied by the user. A user may provide values for some or all of the instance variables. When the value of a variable is supplied by the user, the constructor may check if that value is valid.

A class may also contain methods to compute additional metrics of interest, the Pet class includes a method to compute the age of the pet in human years based on the age of the pet in calendar years. Another method that may be included to print the contents of all the instance variables and additional computations as necessary in a single method invocation.

The following code snippet defines the Pet class.

```java
public class Pet
{
    private String name;
    private int age; //in years
    private double weight;//in pounds
    // Constructors
    public Pet()     // Default constructor
    {
        name = "No name yet.";
        age = 0;
        weight = 0;
    }
    Public Pet(String initialName, int initialAge, double initialWeight) // Includes initial values for all the variables.
    {
        name = initialName;
        if ((initialAge < 0) || (initialWeight < 0)) // Check validity
        {
            System.out.println("Error: Negative age or weight.");
            System.exit( status: 0);
        }
        else
        {
            age = initialAge;
            weight = initialWeight;
        }
    }

    public Pet(String initialName) // Just one instance variable
    {
        name = initialName;       // Similar constructors can be
        age = 0;                  // defined with just age & weight
        weight = 0;
    }

    public String getName()      // Get and Set methods for name.
    {                            // Similar methods can be defined for
        return name;            // age and weight
    }
    public void setName(String newName)
    {
        name = newName;
    }
```

```
    public int getAgeInHumanYears() // Compute the age in human years.
    {
        int humanAge = 0;
        if (age <= 2)
        {
            humanAge = age * 11;
        }
        else
        {
            humanAge = 22 + ((age-2) * 5);
        }
        return humanAge;
    }

    public void writeOutput()  // Write the values of all instance
    {                          // variables and additional computations.
        System.out.println("Name: " + name);
        System.out.println("Age: " + age + " years");
        System.out.println("Weight: " + weight + " pounds");
        System.out.println("Age in human years: " + getAgeInHumanYears() + " years");
    }
}
```

When we write a class, it is necessary and common practice to include a test client or a test driver to test the methods of the class. The following code snippet, `PetTest.java` shows some sample test code, along with the output.

```
import java.util.Scanner;
public class PetTest
{
    public static void main (String[] args)
    {
        Pet myPet = new Pet( initialName: "Fido",  initialAge: 0 , initialWeight: 150);
        myPet.writeOutput();
        System.out.println("\n");
        Pet yourPet = new Pet( initialName: "Balto",  initialAge: 3,  initialWeight: 180);
        yourPet.writeOutput();
        yourPet.setName("Blaze");
        System.out.println("Your pet has a new name:  " +yourPet.getName());
    }
}
```

The output of the above test code is as follows:

```
Name: Fido
Age: 0 years
Weight: 150.0 pounds
Age in human years: 0 years


Name: Balto
Age: 3 years
Weight: 180.0 pounds
Age in human years: 27 years
Your pet has a new name:  Blaze
```