**Lab 6.1 Creating a Door class**

Object-oriented languages like Java are designed to make it easy for programmers to implement software versions of real-world objects. In learning Java, an important skill to master is the ability to represent an object in code. Objects that we model are described using Java classes, so we have chosen to begin this lab by modeling a very simple, everyday object: a door.  
  
Write the code to create a class that models a door object. Don’t worry about the internal details just yet. Just give the class a name and an empty body for the moment. We will add more to the class shortly.

**Lab 6.2 Adding Properties**

When modeling an object as a class, we also need to describe the properties it possesses. An everyday object that we wish to model always has one or more properties that describe it. For instance a door object might have a name like “Front” or “Side” to distinguish it from other doors. Another property that could describe a door is its state: “open” or “closed”. Properties of objects are described in code by using nouns like “state” or “name” to create instance variables that hold values.  
  
Add instance variables to your Door class for the name of the door and its state. Experience has shown that we almost always want to limit the visibility of instance variables inside the same class, so make the access modifiers of state and name private. And because the state and name properties have values like “open” or “front”, let the instance variables you create be of type String.

**Lab 6.3 Adding Methods**

Objects also have operations which can be invoked on the object and which may change the object in some way. For instance, the operations for a door object could be “open” and “close”. An operation on an object corresponds to a Java method and is described in code by using a verb like “open” or “close”. Invoking a method may change the value of an instance variable. For example, invoking close() would change the value of the state variable from “open” to “closed”.  
  
Declare methods for open and close. Because we usually want to allow free access to the methods a class contains, make the access modifier for each method public.

**Lab 6.4 Adding a Constructor**

Now that we have a Door class, we would like to be able to create some Door objects. Java constructors are components of a class whose purpose is to create objects of the given class and to initialize the object’s instance variables. Java provides a constructor with no arguments (the “default” constructor) for every class and the Door class is no exception. Unfortunately, the default constructor that Java provides initializes the state and name variables to null. This is unacceptable.  
  
Add a constructor for the Door class that receives two arguments: the name of the door and its initial state. Because we want to use the constructor outside of the class, make the access modifier for the constructor public.

**Lab 6.5 Adding an Accessor Method**

It is often convenient to have accessor methods that operate on a single instance variable. Here is an accessor method for the name variable:

public String getName()  
{  
 return name;  
}

The word String in front of getName() indicates that the method returns a String when it is invoked. The body is simple and just returns the value of name.  
  
Add this method to your class and write a similar accessor method for the instance variable state.

**Lab 6.6 Adding a Mutator Method**

Many instance variables in a class will have corresponding mutator methods that allow you to change the value of the variable. Here is a mutator method for the name variable:

public void setName(String newName)  
{  
 name = newName;  
}

The word void in front of setName() indicates that the method does not return a value when it is invoked. The body is simple and copies the value of the parameter variable newName to instance variable name.  
  
Add this method to the class and write a similar mutator method for the instance variable state.

**Lab 7 Unit Testing**

compile and run the code below:

/\*\*  
 A class to test the Door class.  
\*/  
public class DoorTester  
{  
 /\*\*  
 Tests the methods of the Door class  
 @param args not used  
 \*/  
 public static void main(String[] args)  
 {  
 Door frontDoor = new Door("Front", "open");  
 System.out.println("The front door is " + frontDoor.getState());  
 System.out.println("Expected: open");

}  
}

Create a second Door object called “backDoor” with the name property “Back” and an initial state of “closed”. Verify that the object was properly created. Use the mutator to change the state of object backDoor to “open”. Verify that the mutator is working.

Create a third Door object called “sideDoor” with the name property “Side” and an initial state of “open”. Verify that the object was properly created. Use the mutator to change the name of object sideDoor to “Front”. Verify that the mutator is working.