RaceHorse

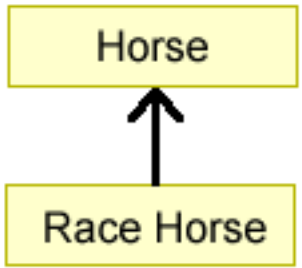
Directions

A Horse class is defined with the following attributes and behaviors. Copy the following code into a source file named **Horse.java**.

```
public class Horse
{
    // instance variables
    private String owner;  // owner of horse
    private int age;       // age of horse
    private double value;  // value of horse

    public Horse()
    {
        owner = "";
        age = 0;
        value = 0;
    }

    public Horse(String o, int a, double v)
    {
        owner = o;
        age = a;
        value = v;
    }

    public String toString()
    {
        String str;
        str = "Owner = " + owner + "\n" +
              "Age   = " + age   + "\n" +
              "Value = $" + value;
        return str;
    }
}
```

The method **toString** returns a string representation of the values stored in a horse object's instance variables.

Define a RaceHorse class with all of the attributes of the Horse class but with an additional attribute for the number of races a horse has **won**. Since a race horse **is-a** type of horse this relationship can be represented in Java using inheritance. The diagram below illustrates this relationship.



Inheritance Rules

- A subclass can add new private instance variables.
- A subclass can add new public or private methods.
- A subclass can override (redefine) inherited methods.
- A subclass must define its own constructors.
- A subclass cannot access the private members of its superclass.

Copy the following code into a source file named **RaceHorse.java**

```
public class RaceHorse extends Horse
{
    // instance variables



    // Default constructor
    // Use the keyword super to call the Horse class's default
    //   constructor so that the instance variables inherited
    //   from this class can be initialized.
    // Initialize the instance variable numRacesWon to zero.
    public RaceHorse()
    {

    }

    // Second constructor
    // Use the keyword super to call the Horse class's second
    //   constructor so that the instance variables inherited
    //   from this class can be initialized to values specified
    //   in the parameter list.
    // Initialize the instance variable numRacesWon to the value
    //   specified in the parameter list.
    public RaceHorse(String owner, int age, double value, int races)
    {

    }

    // Accessor Method - getRacesWon
    // Return the number of races won by this horse
    public int getRacesWon()
    {

    }

    // Mutator Method - wonRace
    // Increment the number of races won attribute
    public void wonRace()
    {

    }
```

```
    // toString method
    // Build and return a string representation of the instance
    // variables from both the Horse class and the RaceHorse class.
    // Use the keyword super to call the toString method of
    //   the Horse class then concatenate the numRacesWon attribute
    //   to the end to produce the output shown in the Sample
    //   Run.
    public String toString()
    {

    }

}
```

Modifications

Make the following additions and modifications to the RaceHorse class:

1. **Rule :** A subclass can add new private instance variables.
   - Add an instance variable of type int named **numRacesWon**.

2. **Rule :** A subclass can add new public or private methods.
   - Implement the accessor method **getRacesWon**.
   - Implement the mutator method **wonRace**.

3. **Rule :** A subclass can override (redefine) inherited methods.
   - Override the **toString** method so that it includes the numRacesWon attribute.

4. **Rule :** A subclass must define its own constructors.
   - Add a **default constructor**
   - Add a **second constructor** which includes 4 parameters to initialize the instance variables owner, age, value, and numRacesWon.

5. **Rule :** A subclass cannot access the private members of its superclass.