Assignment 10: Reading, Written Assignment and Programming
Project due 11/14

For this week you have to complete a Reading Assignment, Written Assignment based on the reading, and a small Programming Project, designed to reinforce the knowledge obtained through reading and written exercises and let you try your hand at Class and Objects in Java.

Reading Assignment

Read Chapter 4, *Defining Classes and Methods* completing self-check exercises. A big part of this chapter is devoted to defining instance methods. A lot of material there should already be familiar: parameter passing, local variables, return values, etc. Nevertheless, you should read the entire chapter carefully, paying attention to the new terminology and OO features of Java.

Review Handout 10, pay attention to notes and details of the code.

Written Assignment

Detach the next four pages of this handout, answer the questions that appear on these pages and submit to me by the beginning of class on Thursday. This assignment will be assigned credit based on the effort, however you should take it seriously. Similar questions will appear on the final exam.

Programming Assignment

Project *Employee* is described on page 6 of this handout.

For extra credit (6 points) write a program (or part of it) that is described in Programming Exercise 4, page 293. Submit that program by emailing it to me.
Answer questions based on the definition of the Species class from Species.java (from Chapter 4 of the textbook, also available electronically on the textbook’s CD) that follows.

```java
/***********************
*Class for data on endangered species.
***********************************/
public class Species
{
    private String name;
    private int population;
    private double growthRate;

    public void readInput()
    {
        System.out.println("What is the species name?");
        name = SavitchIn.readLine();
        System.out.println("What is the population of the species?");
        population = SavitchIn.readLineInt();
        while (population < 0)
        {
            System.out.println("Population cannot be negative.");
            System.out.println("Reenter population:");
            population = SavitchIn.readLineInt();
        }
        System.out.println("Enter growth rate (percent increase per year):");
        growthRate = SavitchIn.readLineDouble();
    }

    public void writeOutput()
    {
        System.out.println("Name = " + name);
        System.out.println("Population = " + population);
        System.out.println("Growth rate = " + growthRate + ")
    }

    //Precondition: years is a nonnegative number.
    //Returns the projected population of the calling object
    //after the specified number of years.
    *****************************/
    public int projectedPopulation(int years)
    {
        double populationAmount = population;
        int count = years;

        while ((count > 0) && (populationAmount > 0))
        {
            populationAmount = (populationAmount +
                                (growthRate/100) * populationAmount);
            count -= 1;
        }
        return (int) populationAmount;
    }
```
count--;  
}  
if (populationAmount > 0)  
   return (int)populationAmount;  
else  
   return 0;  
}  

public void set(String newName, int newPopulation, double newGrowthRate)  
{  
   name = newName;  
   if (newPopulation >= 0)  
      population = newPopulation;  
   else  
   {  
      System.out.println("ERROR: using a negative population.");  
      System.exit(0);  
   }  
   growthRate = newGrowthRate;  
}  

public String getName()  
{  
   return name;  
}  

public int getPopulation()  
{  
   return population;  
}  

public double getGrowthRate()  
{  
   return growthRate;  
}  

public boolean equals(Species otherObject)  
{  
   return ((name.equalsIgnoreCase(otherObject.name))  
      && (population == otherObject.population)  
      && (growthRate == otherObject.growthRate));  
}  

1. List the instance variables of the Species class  

2. List the accessor methods of the Species class
3. List the mutator methods of the `Species` class

4. Explain what’s wrong with the following code: (you may want to run this to find out the answer. To do that - create a project in JBuilder 6 like you did before, add files `Species.java` and `Exercise.java` to the project and compile it. To run this project - you’d have to select Exercise as the main class, since that class has method `main`.)

   ```java
   public class Exercise{
       public static void main(String[] args) {
           Species sp = new Species();
           sp.name = "Hobbit";
           sp.population = 200;
           sp.growthRate = 1.5;
           sp.writeOutput();
       }
   }
   ```

5. Would the program from item 4 work if instance variables of class species were declared as `public` rather than `private`? Why?

6. What’s wrong with the following code:

   ```java
   public class Exercise2{
       public static void main(String[] args) {
           Species sp;
   ```
sp.set("Hobbit", 200, 1.5)
sp.writeOutput();
}
}

7. Complete the program below by filling in the condition for checking if species object sp1 does not define the same species as sp2 (the condition should use the equals() method of Species class.)

```java
public class Exercise3{
    public static void main(String[] args) {
        Species sp1 = new Species();
        Species sp2 = new Species();
        sp1.readInput();
        sp2.readInput();
        if (________________________________________)
            System.out.println("different");
        else
            System.out.println("same");
    }
}
```

8. Consider the instance method `equals`. Explain what is being compared there.
Programming Assignment

Programming Project

Employee: extend Employee class and use it due 11:00 p.m. on Thursday, 11/7 worth 6 points

In this assignment you are asked to extend the Employee class and create an EmployeeDemo class that uses the new version of Employee to create an object of class Employee, assign its name, gender, pay rate and unpaid hours attributes according to the data entered by the user, and print out the Employee information using the modified version of PrintEmployeeInfo() method.

Part 1: modifying Employee class
The first modification you’re asked to make is to add an attribute to record the gender of each employee. To do that you need to add another instance variable to the Employee class that would store the gender information(make the variable private). The way you store the employee’s gender is completely up to you - the instance variable that records the gender could be of boolean, String, char or other type.

Furthermore, create an accessor and a mutator method for the new attribute (called getGender and setGender respectively).

Lastly, modify the PrintEmployeeInfo() method so that it uses pronouns him or her based on the gender of the employee to print the employee information in the following format

Jane Smith has worked 10.0 hours
At the pay rate of 9.5 the amount due to her is 95.0

or

George Clark has worked 12.0 hours
At the pay rate of 12.0 the amount due to him is 144.0

Preliminaries

Since in this project you are extending the Employee class definition at first download file Employee.java from the course webpage to the project folder for this assignment. Add Employee.java to the project.

Add a new file to the project, call it EmployeeDemo.java. The file should be created in the same project folder. That file will define the main method of the application in the same way as the EmployeeObjectsDemo.java from the lecture Handout 10.

After extending the Employee class in the specified way, define a main method in the EmployeeDemo class that

1. creates a new object of class Employee,
2. asks the user to specify the name of the employee,
3. asks the user to specify the pay rate of the employee,
4. asks the user to specify the gender - enter M or m for male, and F or f for female.
5. asks the user how many hours the employee has worked today
6. sets the attributes of the object to the values the user has provided,
7. calls the `PrintEmployInfo()` method to display the object data.

Here's a sample interaction

Greetings!
Please enter the name of the employee: Andrew Mills
Please enter the pay rate: 17.50
Please enter gender of this employee: m for male, f for female: m
Please enter how many hours the employee has worked today: 10

Andrew Mills has worked 10.0 hours
At the pay rate of 17.50 the amount due to him is 175.0

Had the employee been female - the last line should have read the amount due to her rather then him.