Assignment 3

Topics: Nested Classes, Interfaces, Abstract classes

NOTE: exercises with prefix * are not mandatory**

1. Please, read the tutorial at:
   http://docs.oracle.com/javase/tutorial/java/javaOO/nested.html
   http://docs.oracle.com/javase/tutorial/java/IandI/createinterface.html

2. Implement the classes *Student*, *PhDStudent* and/or *TAStudent* using interfaces and/or abstract classes. Discuss what are the pros and the cons, from an OOP framework perspective (thus not just pragmatically!), to implement these classes in the way you choose to.

3. Let's imagine we defined the following interface and class:

   ```
   public interface StudentInteface {
           // here my interface declaration
   }

   public class Student implements StudentInterface {
           // here my class implementation
   }
   ```

   Given the following variables

   ```
   StudentInterface sInt;
   Student student = new Student();
   ```

   discuss the following assignments:

   - *sInt = null;*
   - *sInt = student;*
   - *student = sInt;*
   - *sInt = new StudentInterface();*
   - *sInt.someMethodInStudentInterface();*

4. This task requires you to implement two classes, at first only using pen and paper (i.e. without any computer and IDE). The first class, called *ClassOne*, has just one variable *someVar* and a single constructor *public ClassOne(anotherVar)*
   The second class, called *ExtendClassOne* extends *ClassOne* and has no variables, no constructors and no methods. Does *ClassOne* compile? Does E*xtendClassOne* compile? After you write down your answers on a piece of paper, write these classes in NetBeans and check your answers. If any of your classes does not compile, read the error messages and fix the problems without removing any class (NOTE: removing *ExtendClassOne* would solve the problem; yet such an approach is rather a cheat than a real fix!).

.

5. Consider the following two classes

```
class Test{
                int a = 23;
                void display(){ System.out.println(this.a);  }
}

class ExtendTest extends Test{
                String a = "hello world";
}
```

   Given an instance *te* of *TestExtend* and one instance *t* of *Test*:

   - What is the outcome of *t.display()* and *te.display()*?
   - Create the getter for the variable a in both classes. What happens and why? Remember that a method can return an object of a subtype of the return type due to polymorphism.
   - Fix the problems in the previous item i.e. have a getter for the variable a in both *Test* and *TestExtend*.
   - Add the following method to the *Test* class

     ```
     void displayWithGet(){ System.out.println(getA()); }
     ```

     where *getA()* is your getter method. What is the outcome of

     t.displayWithGet() and *te.displayWithGet ()* and why?

6. Given the class *Student()* (the most simple one you have created) make it a nested class of class *OOPCourse* that manages the OOP course. Come up with a few methods for class *OOPCourse* and test them.

7. ***Discuss the differences between static nested classes and non-static nested classes and static and non-static methods within each of these two situations. Please, do some tests with your own class as developed in point 6