Polymorphism Lab

This lab contains exercises related to class design, abstract class, interface, inheritance and casting and array operation.

Start Eclipse IDE. Create a new Java project named **Week3-Project,** and complete the following tasks:

Task 1: Creating abstract class *Person*

Create an abstract class named ***Person*** which has three fields to hold the info of first name, last name and email address. The class also has an abstract method: "public abstract void doing();".

Task 2: Creating Interface *Talkable*

Create an Interface ***Talkable*** with a method "void say():".

Task 3: Creating classes: *Student, Professor* and *Staff*

Create classes ***Student***, ***Professor*** and ***Staff*** which are subclasses of the ***Person*** class and implement the interface ***Talkable***.

The three classes implement the "doing()" method by printing out message on console/terminal, correspondingly "`Studying as student!`", "`Teaching as professor!`" and "`Working as staff!`".

Add at least one field to each of the three classes, e.g. "program" for ***Student*** class, "office" for ***Professor*** class, and "title" for***Staff*** class. Each class should implement the say() method by printing messages including greeting, full name, and the info of added field above. For example, "`Hello! My name is John Smith, and I'm in CPD program.`".

Task 4: Creating class: *Meeting*

Create class ***Meeting*** which has only the main method which is as entry to run the program. Do the following in the main method:

Create two objects for each class of ***Student***, ***Professor***, and ***Staff*** - 6 objects in total.

Create an array of ***Person*** with 6 array elements:

```
Person[] working = new Person[6];
```

Add/assign the 6 objects to the array. Use **for-loop** to print out the info of each object in the array by calling toString() and doing() methods.

Here is the sample output:

```
test.Student [Program = CPD, [firstName=John, lastName=Smith,
email=jsmith@myseneca.ca]]
Studying as student!

test.Student [Program = CPD, [firstName=Colin, lastName=Thomas,
email=cthomas@myseneca.ca]]
Studying as student!

test.Professor [Office = T1034, [firstName=Jordan, lastName=Anastasiade,
email=jordan.anastasiade@senecacollege.ca]]
Teaching as professor!

test.Professor [Office = T2099, [firstName=Wei, lastName=Song,
email=wei.song@senecacollege.ca]]
Teaching as professor!

test.Staff [Title = Admin, [firstName=Jack, lastName=Brown,
email=jack.brown@senecacollege.ca]]
Working as staff!

test.Staff [Title = Technical Support, [firstName=Paul, lastName=Miller,
email=paul.miller@senecacollege.ca]]
Working as staff!
```

Create an array of ***Talkable*** with 6 array elements:

```
Talkable[] meeting = new Talkable[6];
```

Add/assign the 6 objects to the array. Use **enhanced-for-loop** to print out the info of each object in the array by calling the say() method. Here is the sample output:

```
Hello! My name is John Smith, and I'm in CPD program.
Hello! My name is Colin Thomas, and I'm in CPD program.
Hello! My name is Jordan Anastasiade, and I'm in T1034
Hello! My name is Wei Song, and I'm in T2099
Hello! My name is Jack Brown, and my title is Admin
Hello! My name is Paul Miller, and my title is Technical Support
```

OTHER REQUIREMENTS

- Each class, interface should be public and have its own source file.

- One (or more) package(s) should be used to organize your class.
- All class fields must be private and getters and setters for these fields should be implemented.
- For the type classes (Person, Student, Staff, and Professor), generate the constructor with the parameter of all fields and override the toString() method.