

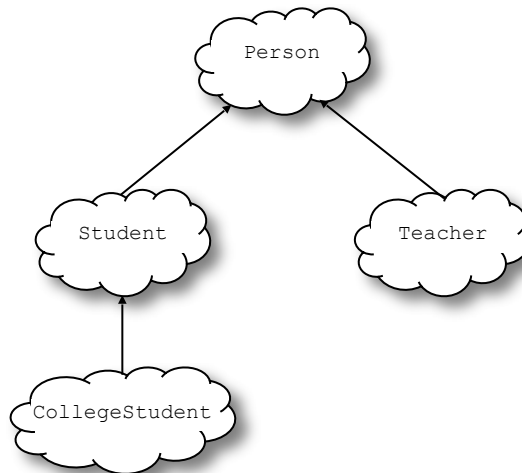
LAB EXERCISE

BackToSchool

Background:

The HighSchool application described in the lesson has two classes: the `Person` superclass and the `Student` subclass. Using inheritance, in this lab you will create two new classes, `Teacher` and `CollegeStudent`. A `Teacher` will be like `Person` but will have additional properties such as *salary* (the amount the teacher earns) and *subject* (e.g. "Computer Science", "Chemistry", "English", "Other"). The `CollegeStudent` class will extend the `Student` class by adding a *year* (current level in college) and *major* (e.g. "Electrical Engineering", "Communications", "Undeclared").

The inheritance hierarchy would appear as follows:



Here is the `Person` base class from the lesson to be used as a starting point for the `Teacher` class:

```
class Person
{
    protected String myName ;    // name of the person
    protected int myAge;         // person's age
    protected String myGender;    // "M" for male, "F" for female

    // constructor
    public Person(String name, int age, String gender)
    {
        myName = name; myAge = age ; myGender = gender;
    }

    public String toString()
    {
        return myName + ", age: " + myAge + ", gender: " + myGender;
    }
}
```

The `Student` class is derived from the `Person` class and used as a starting point for the `CollegeStudent` class:

```

class Student extends Person
{
    protected String myIdNum;    // Student Id Number
    protected double myGPA;      // grade point average

    // constructor
    public Student(String name, int age, String gender,
                   String idNum, double gpa)
    {
        // use the super class' constructor
        super(name, age, gender);

        // initialize what's new to Student
        myIdNum = idNum;
        myGPA = gpa;
    }
}

```

Assignment:

1. Add methods to “set” and “get” the instance variables in the `Person` class. These would consist of: `getName`, `getAge`, `getGender`, `setName`, `setAge`, and `setGender`.
2. Add methods to “set” and “get” the instance variables in the `Student` class. These would consist of: `getIdNum`, `getGPA`, `setIdNum`, and `setGPA`.
3. Write a `Teacher` class that extends the parent class `Person`.
 - a. Add instance variables to the class for *subject* (e.g. “Computer Science”, “Chemistry”, “English”, “Other”) and *salary* (the teachers annual salary). *Subject* should be of type `String` and *salary* of type `double`. Choose appropriate names for the instance variables.
 - b. Write a constructor for the `Teacher` class. The constructor will use five parameters to initialize `myName`, `myAge`, `myGender`, *subject*, and *salary*. Use the `super` reference to use the constructor in the `Person` superclass to initialize the inherited values.
 - c. Write “setter” and “getter” methods for all of the class variables. For the `Teacher` class they would be: `getSubject`, `getSalary`, `setSubject`, and `setSalary`.
 - d. Write the `toString()` method for the `Teacher` class. Use a `super` reference to do the things already done by the superclass.

4. Write a `CollegeStudent` subclass that extends the `Student` class.
 - a. Add instance variables to the class for *major* (e.g. “Electrical Engineering”, “Communications”, “Undeclared”) and *year* (e.g. FROSH = 1, SOPH = 2, ...). *Major* should be of type `String` and *year* of type `int`. Choose appropriate names for the instance variables.
 - b. Write a constructor for the `CollegeStudent` class. The constructor will use seven parameters to initialize *myName*, *myAge*, *myGender*, *myIdNum*, *myGPA*, *year*, and *major*. Use the **super** reference to use the constructor in the `Student` superclass to initialize the inherited values.
 - c. Write “setter” and “getter” methods for all of the class variables. For the `CollegeStudent` class they would be: `getYear`, `getMajor`, `setYear`, and `setMajor`.
 - d. Write the `toString()` method for the `CollegeStudent` class. Use a **super** reference to do the things already done by the superclass.
5. Write a testing class with a `main()` that constructs all of the classes (`Person`, `Student`, `Teacher`, and `CollegeStudent`) and calls their `toString()` method. Sample usage would be:

```
Person bob = new Person("Coach Bob", 27, "M");
System.out.println(bob);

Student lynne = new Student("Lynne Brooke", 16, "F", "HS95129", 3.5);
System.out.println(lynne);

Teacher mrJava = new Teacher("Duke Java", 34, "M", "Computer Science", 50000);
System.out.println(mrJava);

CollegeStudent ima = new CollegeStudent("Ima Frosh", 18, "F", "UCB123",
                                         4.0, 1, "English");
System.out.println(ima);
```

A sample run of the program would give:

```
Coach Bob, age: 27, gender: M
Lynne Brooke, age: 16, gender: F, student id: HS95129, gpa: 3.5
Duke Java, age: 34, gender: M, subject: Computer Science, salary: 50000.0
Ima Frosh, age: 18, gender: F, student id: UCB123, gpa: 4.0, year: 1, major:
English
```

6. There should be one source file for each class: *Person.java* for the `Person` class, *Student.java* for the `Student` class, *Teacher.java* for the `Teacher` class, *CollegeStudent.java* for the `CollegeStudent` class, and *BackToSchool.java* for the `BackToSchool` testing class. Run the testing class and call your instructor to your workstation for scoring.