**CS 203 HW #9**

**Due Dates: Part A –** Friday, April 12 at 11:00pm

**Part B –** Wednesday, April 17 at 11:00pm

This assignment will give you practice using inheritance in Java and practice in software design.

### Overall Specification

For this assignment, you will create a Java program of your choice that takes advantage of inheritance. This assignment is subject to the following requirements:

1. Your program must contain at least four classes you have written for this assignment.
2. Each of your classes must have a clear reason for existing; that is, a class should model one “thing.” Its attributes should describe or capture some aspect of the object being modeled, and the methods should perform actions that pertain to the object and make use of the object’s instance variables in some way.
3. Each of your classes should have a toString method that behaves properly, i.e., it produces a String representation of the object that is more helpful than the toString method inherited from the Object class.
4. At least two classes you have written must inherit from some other class. For example, in the college record-keeping program we discussed in class, the Student class inherits from Person and the Undergraduate class inherits from Student.
5. The classes you define that inherit from others must *clearly benefit* from that inheritance. Your use of inheritance should not be contrived to meet the requirements of the assignment.
6. In at least one place, a class that inherits from a parent class should override one of its parent’s methods. The constructor and toString methods don’t count for the purposes of this requirement.
7. Your program should have an obvious purpose or point. It should not be just a random collection of classes. Each class you write must contribute significantly to the program's overall functionality.
8. If you wish, you may build upon code you’ve written from previous assignments. You can even use code written by others. However, code written by others must be clearly identified. If you start with existing code, you still must write additional code specifically for this assignment that meets the requirements above. The additional classes you create must be a non-trivial extension of the program.

**Examples of trivial extensions that do not meet the specifications for this homework:**

1. Adding additional faces to the Faces lab for CS273
2. Adding more filters to the Digital Camera homework

**Examples of acceptable extensions:**

* + - 1. A bookstore that sells more than just books
      2. Re-writing Mini-Sudoku to use multiple classes

**Other ideas:**

A store of some kind i.e. grocery, furniture, electronics, clothing

Coffee shop or restaurant

An apartment complex or dorm

I’m sure you can come up with many more creative ideas!

Note: If you are unsure about the scope of the project - too small? too large? please check with me before you start.

### Part A: Design Your Program (20% of overall grade)

**Overview** (4%)

Create a clear and concise description in English of what you intend to create that is one or two paragraphs in length. Your description should include which classes you propose to implement and what your class with a main() method in it will do with those classes.

**UML Diagram** (16%)

Draw a UML class diagram for the classes you will implement. Each class specification

should show the attributes (instance variables) and actions (methods) that each class will have, including their public/private/protected status. The diagram should also show the “is-a” and “has-a” associations between the classes. You can draw this diagram by hand or use a tool[[1]](#footnote-1) for the purpose. The UML diagram you turn in must be in one of the following image or document formats: .png, .jpg, .pdf, .docx, or .odt. Other file formats are not acceptable.

Part of your grade for this diagram will be based upon whether your design meets the requirements listed in the Overall Specification section above.

### Hints for Part A

* **Important:** Be careful not to get carried away! Design a program that you know you can build. It’s okay to have big ideas, but make sure you have a plan to create something of small scope first that meets the specifications. Then you can expand it as time allows.
* If you are uncertain about what meets the requirements, *ask for clarification*. Don’t assume.
* Proposed exceptions or adjustments to the specifications will be considered on a case-by-case basis. Ask first! Do not assume a change is okay.

### Logistics for Part A

There is no starter code for this assignment.

Follow these steps to turn in part A of your assignment:

* Make sure your name is in both of your files, including your UML diagram and your program description document.
* Create a folder that contains:
  + your program description
  + your UML diagram
* Compress the entire folder into a .zip.
* Submit your zip file through the Homework #9A link on Moodle.

### Part B: Write your Program

Create a complete and working program based on the design you created in Part A. It is okay if your final program does not match your design exactly. In fact, some deviation is normal. However, the code you turn in should clearly be derived from your original design.

Your program should make clear use of Java's built-in support for inheritance where it makes sense rather than working around it. For example:

* If extending the functionality of a parent method, you should use the super reserved word.
* If you have a class with multiple constructors that have content in common, use the this reserved word to call a fundamental constructor.
* If you need to identify the type of an object, use the instanceof operator rather than an id variable.

**Grading**

Your grade for part B will be based upon the following criteria:

* [60%] Functionality – Does your program compile and run correctly? Does it meet the Overall Requirements for this assignment?
* [15%] Code Quality – Is your code well written and well commented? Each class and method should have appropriate comment headers using the Javadoc comment conventions. You can check this by loading your starter code into BlueJ and selecting Tools > Project Documentation to view your API.
* [5%] Adherence to Design – Was your design for Part A sufficient to guide your creation of part B. Is your program clearly derived from your original design?

### Logistics for Part B

Follow these steps to turn in part B of your assignment:

* Make sure that exactly one class in your program contains a properly defined main method: public static void main(String[] args). This is the method that the grader will use to run your program. This class counts as one of the four classes you must write.
* In the class that contains the main method, add the following information to the comment header at the top of the file:
  + A short description of what your program does
  + Major changes, if any, you made to your design since part A.
  + Known errors or shortcomings in your program.
  + Any information the grader will need to know to run and use your program
  + Where in the code that the grader can see that you are overriding one or more methods
* Compress your entire BlueJ project into a single .zip archive. **Make sure your UML diagram and program description are included again.**
* Submit your zip file through the Homework #9B link on Moodle.

1. Dia and MS Visio are two recommended tools for this. If you complete your specification in Dia, it can generate starter code for you though you'll have to edit it to meet the homework specification. Dia and Visio are included in the engineering build that is installed on the computers in the Shiley computer labs and classrooms. You can also download a copy of Dia for your personal computer: [http://dia-installer.de](http://dia-installer.de/) [↑](#footnote-ref-1)