**CS203 HW #9  
Spring 2012  
Choose Your Own Program Using Inheritance**

**Due Dates:** Part A: Wednesday, Apr 11, 11 am (design proposal) [note: this is Wed after Easter]  
                   Part B: Wednesday, Apr 18, 11 am (code and report)   
  
This assignment will give you practice using inheritance in Java. You may also choose to work with a partner on this assignment. Working alone is also fine. If you plan to work with a partner, this information should be included in the design proposal. If you work with a partner, the expectation is that both partners contribute to the project as equally as possible. Both partners will receive the same grade, except in extreme circumstances.

**Specification:**

For this assignment, you will create a Java program of your choice that takes advantage of inheritance. However, you must include the following in your program:

1. (6 pts) Your program must contain at least **two** classes (written by you) that each inherit from some other class you have written. For example, in the Ball demo from lecture the Balloon class inherits from Ball and the Chameleon class inherits from Balloon. In the Library example, the Book and Journal classes both inherit from CirculationItem. Thus, you need at least three classes in your inheritance hierarchy.
   1. 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. Do not ignore this requirement. Remember: factor out the common attributes and behavior and put them in the superclass (aka parent, base class).
   2. In at least one place, a class that inherits from a superclass should **override** one of its superclass’s methods.
   3. In at least one place in the code, you should use **super**(..) to call the constructor of the superclass.
   4. In at least one place in the code, you should use **super.methodName**(..) to call the superclass’s version of methodName. See the tick method in the Ball animation and getSummary in the ChildrensBook class of the library example.
2. (.5 pt) Have at least one place where you use the **instanceof** operator.
3. (.5 pt) Have at least one place where you create an array of objects, where the type of objects in the array is the type of parent class of the inheritance hierarchy. You should iterate through the objects in the array at least once. For example, in the library example, the collection is an array of CirculationItem objects and a loop might iterate through this array to call checkout on every object in the array.
4. Each class you write must be robust and have a clear reason for existing. Classes that contain very little content do not count. The entire program should have a purpose. It should not be a random collection of classes.
5. If you wish, you may write a portion of a larger program (similar to the GamePlayer class from HW7), but you need to show that you have tested the objects and methods in your classes.
6. If you wish, you may build upon code you have written from previous assignments. You can even use code written by others and the code that was demonstrated in lecture. 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.
7. The program can be text-based (printing text to the screen and scanning data from the keyboard) or graphical (using JFrame). If you want to do something graphical, your textbook contains information about using JFrame in the last section of the first few chapters. HW4, HW8, and the Ball animation use JFrame for your reference. Also, the on-line API for Java will be helpful: <http://docs.oracle.com/javase/6/docs/api/>

This assignment has deliverables in two parts:

**Part A (design proposal) [7 pts]:**

1. If you plan to work with a partner, include both names of the partnership here. You only need to upload one copy of the proposal for both students. If you are working alone, include just your name here.
2. (2 pts) Create a brief (one paragraph) description in English of what you intend to create for your program.
3. (5 pts) Provide a set of UML class diagrams indicating the instance variables and methods for each class. You should also indicate with arrows the “is a” relationship between classes. Arrows should point to the superclass. The “is a” relationship is as follows: In the library example, a Book “is a” CirculationItem since Book extends CirculationItem. See the UML diagram for the Library code in the coursepack for an example of what to produce for this part. Be sure to include the types of instance variables, the types of parameters for methods, and the return types for methods.
   1. You can draw this by hand and scan it in to your document.
   2. You can draw this using MS Word drawing tools.
   3. You can draw this with Visio and importing the figure into your MS Word document.
   4. You can install and use dia. <http://live.gnome.org/Dia>

**Part B (code and report) [20 points]:** This is the completed code and report for your assignment. It is okay if your final program does not match your diagram from Part A completely. In fact, some deviation is normal, since you may need to modify the design once you start implementation. However, the code you turn in should clearly be derived from your original design.

*Important***:** Be careful not to get carried away! Start with a *very small, very simple* programthat *works* and build upon it slowly and carefully. It’s okay to have big ideas, but make sure you have a plan to create something with a small scope first that meets the homework specifications before you do any “additional enrichment.”

*Other thoughts:*

* If you are uncertain about what meets the requirements, *ask for clarification*.
* If you are paralyzed for ideas about what to do, feel free to modify the Library or Ball animation examples we have discussed in class. Alternatively, consider enhancing a previous assignment to add functionality to it.
* You might think about writing a program that will be useful for you to use in the future or is related to another course you are taking.

**Logistics:**

* There is no starter code for this project. To create a new BlueJ project, go to Project->New Project. Name your project. From there, you can create new class files. BlueJ will create starter code for you when you create a class. Tammy usually deletes this and starts from scratch.
* Turning in part A: Submit your word document. Name it username1\_username2\_proposal.docx if you are working in a partnership. If working alone, name it username\_proposal.docx. Submit this file to Moodle.
* Turning in part B: Create a zip file containing your entire BlueJ project and your summary report. Name it username1\_username2\_HW8.zip if working in a partnership. If working alone, name it username\_HW8.zip. Submit this file to Moodle. Just one set of files is needed from one of partners if you worked in a partnership.

**Grading Guidelines:**

Part A will be graded as shown above (out of 7 points).

Your code for Part B will be graded on a scale of 0 to 7 in two categories:

* Code Quality: Design, Implementation Style, Comments, Clarity
* Code Functionality: Functionality of code and adherence to your design from part A; should meet the requirements in the specification above

Your summary report for Part B will be graded on a scale of 0 to 6 based on:

* Correct use of technical vocabulary
* Correct use of diagrams
* Clarity
* Organization / Headings included
* Evidence of Testing

**Report Guidelines and Format: (Use the template for HW1, but answer the questions below)  
  
1. Introduction and System Use:** Describe what your program does and how a user should interact with your program. This paragraph might be similar to your proposal.   
  
**2. System Description:** Include the UML class diagrams for the final program. You may hand-draw this and scan it or use electronic drawing tools (MS Word or Visio). Did any of your classes change from your design proposal? If so, what were the changes?

If you modified existing code, describe what parts you modified and created and what parts you used “as is”. Be sure to attribute the source for code that you did not write yourself.

**3. Testing and Evaluation:**

**3.1 Specification:** If your program does not meet the specifications, please note these differences. If it meets the specifications for all the features listed at the top of this assignment, write “Application meets all specifications.”

**3.2 User Test:** Have a friend use your program. It may be that the program simply runs the main method without a lot of user activity. If this is the case, explain to the user what is happening. If the program is interactive, let the user type/mouse click/etc to interact with your program.

What did the user like best?

Did the user have any suggestions for improvement?

Ask the user any other questions that you would like to know about, given that you designed the program.

**3.3 Results:** Include a screen shot of your program (if it is graphical) or the terminal window text (if the program is text-based).  
  
**4. Conclusion:** What did you learn by completing this assignment? If you worked with a partner, how did you break up the work? (Did you pair program? Did you each implement some of the classes?)

1. How long did you spend on this assignment (both parts A and B)? If you worked in a partnership, indicate the hours for each person. For example, you might write “Joe worked a total of 8 hours. Sue worked a total of 7 hours.”

2. By typing your name(s) here, you are acknowledging that the code and report you are submitting are your own.

**Appendix:** Copy and paste your code here – all classes that you wrote. Use Courier Font 8 pt, so the characters align correctly.