Rebekah Peterson
CS 2420
Last updated 9/7/16

1. This assignment is traditionally done with Pair Programming. Were there times you wish you had a partner to bounce ideas off of? We will ask you to compare your experiences with these first, "single" to assignments to the paired assignments later on in the semester. Please use your answer here for reference later.

There have been a few times in the past that I wished I had a partner to program with. Mostly this happens when I get stuck on understanding how a particular thing works, or how to implement something. On this assignment, I couldn't seem to understand how GregorianCalendar and due date worked together. It just didn't logically make sense to use a calendar for a single date. It took a lot of time and reading to get it, even though it was relatively simple compared to everything else. It also would have been nice to have a partner to keep more focused and thoroughly test. That being said, I was on vacation. I already am dealing with the drama that comes for finding a partner for one assignment. I'm glad I didn't have this one too!

2. Java's built-in classes Comparable and Comparator are both interfaces for doing comparisons among objects. What is the difference in the two interfaces? Give a situation when it is best to use each. Is it possible to change the extra features in LibraryGeneric such that the Comparable interface is used instead of Comparator? Why or why not?

Comparable and Comparator are both interfaces that have to do with making comparisons. Comparable has one method required to be implemented: compareTo(). It can only be implemented once. This interface should be used in conjunction with making comparisons that have to do with the natural ordering of an object. Examples include objects such as numbers, words, or time. Comparator has one method as well: compare(). Because a comparator is itself an object, you can have a variety of sorting options. This interface should be used in instances where there are many different ways to sort an object, like in fruit, people, or library books.

It may be "possible" to use Comparable for one version of comparing LibraryBookGenerics, but it definitely isn't a good design choice! Our

implementation is a classic example of why we would use a comparator. There isn't a clear, natural ordering.

3. Comment about the efficiency of your programming time. Did you utilize the time spent on this assignment effectively? How might it be improved?

Always looking back, I feel like I could have used my time more wisely because I know how to do everything. If I were to recode this it would take much less time. However, I think I did fairly good job at trying to solve problems on paper before typing haphazardly into eclipse. I definitely should have done better at writing tests before writing the program. This always makes the programming go smoother, and catches bugs in an organized fashion. It also makes testing less burdensome when you incorporate it into the programming, instead of putting it off until the end. Obviously I didn't do nearly enough testing, but I guess that's what happens when your laptop dies and you go on vacation.

4. *Reiterate why writing Generic code is important for this course. Phrase your answer in terms of Data Structures and Algorithms.*

Generic coding is important to improve efficiency. In terms of data structures and algorithms, we don't want to have to recode each algorithm for each possible reference type we run into or create. It is much more efficient to use generic programming to avoid the redundancy, and still allows us to be type specific.

5. How many hours did you spend on this assignment?

8 - 10 hours