Project #7
CS 2210 – Fall 2021
Sam DeCook

I. Requirements: Restate the problem specification, and any detailed requirements

Develop an ArrayHeap class with implements the Heap interface and extends ArrayBinaryTree class. The heap must be expandable.

II. Design: How did you attack the problem? What choices did you make in your design, and why? Show class diagrams for more complex designs.

We just had to write the add() and removeRoot(). I made sure I understood the concept behind each method and then wrote out the logic. I separated out the bubble up and down procedures and also made a swap method to swap the elements.

III. Security Analysis: State the potential security vulnerabilities of your design. How could these vulnerabilities be exploited by an adversary? What would be the impact if the vulnerability is exploited?

I don't know if there are any potential security vulnerabilities. There isn't really any way that an attacker could gain access since this is just a data structure.

IV. Implementation: Outline any interesting implementation details.

I swapped the ArrayPositions in my swap method instead of the Item. This resulted in a (slightly) longer swap method, but it allowed me to use less dereferences and allowed recursion to be easier since I didn't have to pass a different ArrayPosition, I could pass through the one I had been working with.

V. Testing: Explain how you tested your program, enumerating the tests if possible. Explain why your test set was sufficient to believe that the software is working properly, i.e., what were the range of possibilities of errors that you were testing for.

I tested my program by running it. Since it is working and outputting the numbers in a non-decreasing fashion and not throwing any errors, I am assuming that it is working properly.

VI. Summary/Conclusion: Present your results. Did it work properly? Are there any limitations? If it is an analysis-type project, this section may be significantly longer than for a simple implementation-type project.

My code compiled and ran as properly and produced the expected output.

VII. Lessons Learned: List any lessons learned. What might you have done differently if you were going to attack this again.

I would have been more thorough when writing my methods. It's easier to consider all the possible scenarios than to figure out what an error message means.