**Scoring Rubric for Project 3 : BubbleSort**

*Due 10/03/2019 @ 3:30 pm*

|  |
| --- |
| Student Name: Henry Evans |

|  |  |  |
| --- | --- | --- |
|  | **Score** | **Maximum** |
| **Execution (50 pts):** | | |
| Program compiles without errors (warnings are okay) | 50 | **50** |
| **Implementation (40 pts):** | | |
| Uses function declarations as provided | 5 | **5** |
| Main function includes at least one unit test for Swap (can use assert or printed output) | 5 | **5** |
| BubbleSort works for input size of 42 and 47 (all or nothing) | 5 | **5** |
| Use a dynamically allocated array for BubbleSort | 2.5 | **5** |
| Free the allocated array at the end of Main function | 2.5 | **5** |
| Complete the BubbleSort unit test | 5 | **5** |
| Use command line arguments to read the array size and the seed | 5 | **5** |
| Measure the execution times of MergeSort and BubbleSort and plot them on a graph | 4 | **5** |
| **Style (10 pts):** | | |
| The driver and functions are easy to follow based on the use of comments | 3 | **6** |
| Easily identifiable variable names | 4 | **4** |
| **Total (100 pts):** | 91 | **100** |

Notes:

When you dynamically allocate your array, you should be creating it with the variable length that the user specified, not with 200. The point of dynamic allocation is that the program doesn’t know how much memory to allocate until runtime. This would also cause problems with some of your for-loops since you are indexing array based on v.size(), but if length is greater than 200, you will get a segmentation fault.

Your memory deallocation statements should be the other way around. First, you want to free all the memory that array points to, and then you can delete the pointer itself. If you delete the pointer first, you will not be able to find the memory it was pointing to anymore, so it will not be freed. So, it will be left as a dangling pointer.

In the future, it is good practice to check that the number of command line arguments that are inputted by the user are the same as the number the program is expecting. For example, if a user didn’t type in the length, you wouldn’t want the program to run. You can check for this with some code like follows:

if (argc != 3) {

cout << “Usage: ./BubbleSort <seed> <length>” << endl;

exit(1);

}

You added some comments to main, but you should have some descriptive comments in the functions bubbleSort and swap as well.

You should also label the y-axis on your graph.