**Scoring Rubric for Project 3 : BubbleSort**

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

|  |
| --- |
| Student Name: Nathan Carter |

|  |  |  |
| --- | --- | --- |
|  | **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 | 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 | 6 | **6** |
| Easily identifiable variable names | 4 | **4** |
| **Total (100 pts):** | 96.5 | **100** |

Notes:

You should label the axes on your plot.

For future reference, you should do your check of if argc doesn’t equal 3 before you try to read in the seed and length, or you will get a segmentation fault if the user doesn’t input the correct number of arguments. It’s also nice to print out a clear usage line as follows: cout << “Usage: ./BubbleSort <seed> <length>” << endl; You also want to call exit(1); to quit the program, since you don’t want it to keep running if you don’t have all the required arguments.

To free the allocated memory, you need delete [] a; a = nullptr; The first statement frees the memory that stores all the array values, and the second statement ensures that array will not be a dangling pointer.