|  |  |
| --- | --- |
| Your name: (Person whose work is being evaluated) | Alexander Meyers |
| Names of other group members participating in the evaluation: | Raven Lickey, Alexander Meyers |
| Date: | 4/9/19 |

Instructions  
You should have already completed Assignment 1 and uploaded your source files to moodle. After you and another student (or students) have evaluated your work, you will submit this evaluation along with any revisions to your lab work to moodle. You will be graded on your revised lab work and the quality of this evaluation, but this evaluation will not determine your grade.

|  |  |
| --- | --- |
| **Criteria** | **Evaluation** |
| Does the program compile without errors or warnings? | *y* |
| Does the test program run with your class without crashing? | y |
| Is your class split into a header and a source file? | n |
| Does the class have default constructor, overloaded constructor, and destructor? | y |
| Do the constructors create an array using new and the destructor properly delete it? | y |
|  |  |
| Do the various tests run properly? | y |
| getSize returns the proper size of the array, not the number of elements? | y |
| setSize, resizes the array if necessary? | y |
|  |  |
| getAt and setAt properly access the array locations? |  |
| getAt and setAt correctly throw exceptions for invalid indices? |  |
|  |  |
| append properly adds at the next location? | y |
| If the array is full, append causes a resizing to twice as large? | y |
|  |  |
| insertAt causes an array resizing if the array is full | y |
| removeAt and insertAt throw exceptions if their indices are incorrect | *y* |
| insertAt properly moves other items out of the way | *y* |
| removeAt properly compresses the array when done | y |

General Comments: