# Assignment: Complexity and Sorting - Marking Criteria

As a group, evaluate your work against each criterion and assign yourselves a mark for each row. If you would like, provide a short explanation for the self-assigned mark in the Notes column.

## Marking Criteria

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Needs Improvement (0–50%)** | **Good (51–75%)** | **Excellent (76–100%)** | **Group-Assigned Mark** | **Notes** |
| **Working Code** | * Command line does not work properly and does not generate the objects using reflection | * Command line works and generates the objects partially using reflection | * Command line works properly and generates the objects using reflection | **/10** | Mark: 100  We encountered challenges in understanding the data structure and found the command-line input format somewhat unclear, but we were able to make it work eventually. |
| **Style** | * Sorting algorithms don’t work on list of Comparables | * Sorting algorithms partially work on a list of Comparables | * Sorting algorithms work on a list of Comparables | **/5** | Mark: 100 The algorithms worked properly once we adapted them to the comparable and comparator interfaces, at the beginning we just created an algorithm and we needed to make some adjustments to make it work for accomplish the requirements of the assignment. |
| **Testing** | * Comparable/ Comparator not implemented correctly | * Comparable/ Comparator partially implemented | * Comparable/ Comparator implemented correctly | **/5** | Mark: 100  At first, we implemented the comparators as a single class. After receiving feedback from the instructor, we were able to modify it. Now, each comparator has its own class, and the height is sorted using the comparable interface. |
|  | * Didn’t use correct OOP modelling of objects | * Used somewhat correct OOP modelling of objects | * Used correct OOP modelling of objects | **/5** | Initially, we had limited knowledge about testing units, which led to implementing them towards the end. Nevertheless, they proved to be valuable for ensuring the correctness of our code. |
|  | * Incorrect naming and packaging of classes according to Java standards | * Partially correct naming and packaging of classes according to Java standards | * Correct naming and packaging of classes according to Java standards | **/5** | Mark: 100  We tried to follow the naming conventions reviewed in class. |
|  | * Bubble sort not implemented in utility class | * Bubble sort implemented partially in utility class | * Bubble sort implemented in utility class | **/5** | Mark: 100 |
|  | * Selection sort not implemented in utility class | * Selection sort implemented partially in utility class | * Selection sort implemented in utility class | **/5** | Mark: 100 |
|  | * Insertion sort not implemented in utility class | * Insertion sort implemented partially in utility class | * Insertion sort implemented in utility class | **/5** | Mark: 100 |
|  | * Merge sort not implemented in utility class | * Merge sort implemented partially in utility class | * Merge sort implemented in utility class | **/10** | Mark: 100 |
|  | * Quick sort not implemented in utility class | * Quick sort implemented partially in utility class | * Quick sort implemented in utility class | **/10** | Mark: 100 |
|  | * Student’s choice of sorting algorithm not implemented in utility class | * Student’s choice of sorting algorithm partially implemented in utility class | * Student’s choice of sorting algorithm correctly implemented in utility class | **/5** | Mark: 100  The instructions asked to choose an algorithm that was different from the ones outlined in the document. We chose heap because we were not familiar with binary trees and wanted to learn more about them. |
|  | * Description and complexity analysis are unclear | * Description and complexity analysis are partially clear | * Description and complexity analysis are clear | **/5** | Mark: 100 |
|  | * Sort Output is dissimilar to expected | * Sort Output is somewhat similar to expected | * Sort Output matches what is expected | **/5** | Mark: 100 |
|  | * Incorrect timing of sorts shown | * Somewhat correct timing of sorts shown | * Correct timing of sorts shown | **/5** | Mark: 100 |
|  | * No Readmes/Instructions or javadoc documentation submitted | * Some Readmes/Instructions and javadoc documentation submitted | * All Readmes/Instructions and javadoc documentation submitted correctly | **/10** | Mark: 100 |
| **Total** | | | | **/95** |