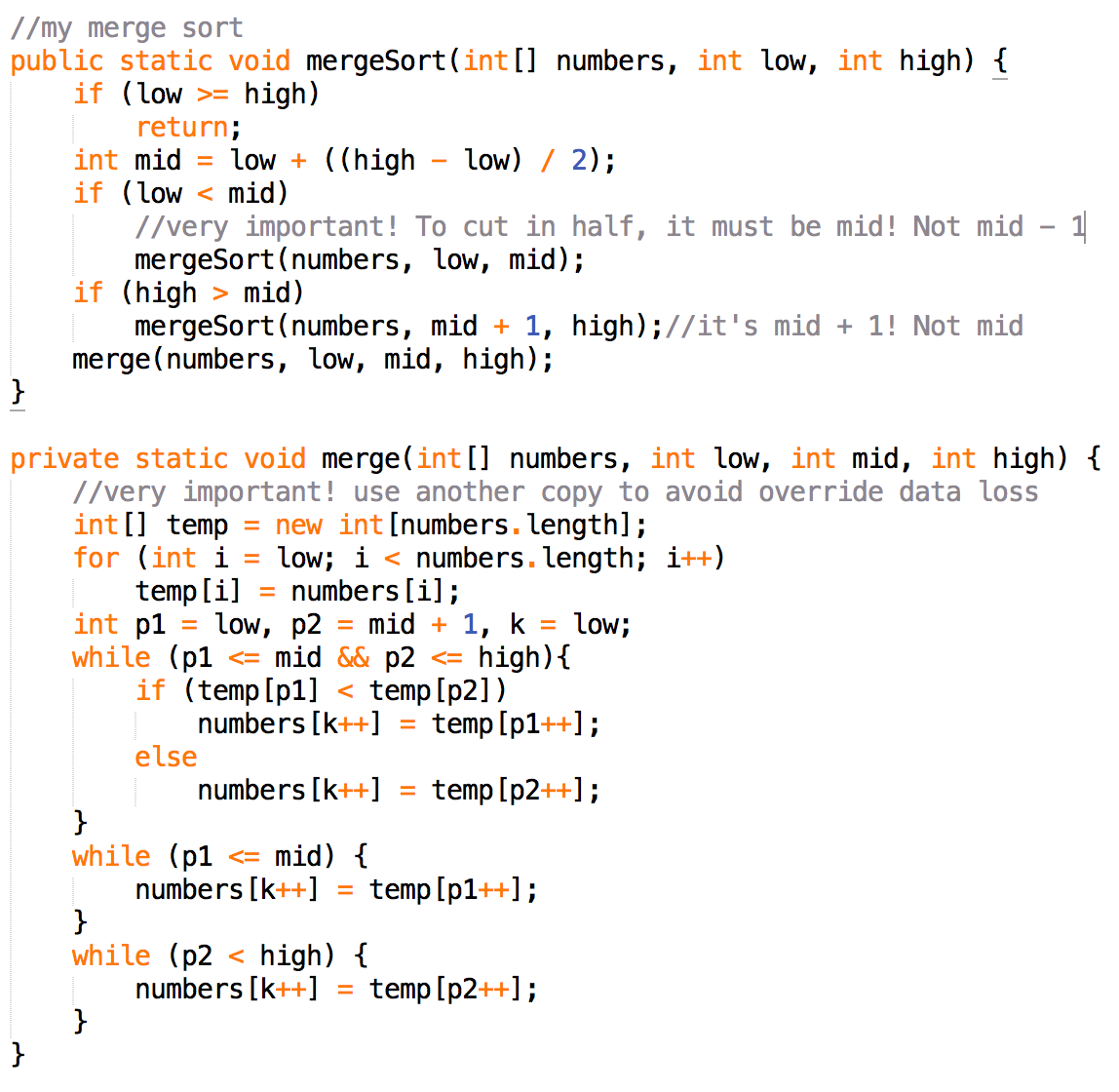
**I. Sorting**

1. **Sorting Algorithm**

* **Complexity of Sorting Methods**



* **Comparison**
* **Quick sort**: Best average run time (O(nlogn)); don’t have to be stable
* **Merge sort:** stable; fast (all O(nlogn))
* **Heap sort:** care more about worst case (O(nlogn)); space limited (O(k)); don’t have to be stable
* **Insertion sort:** N is guaranteed to be small (O(N^2)), stable.
* **Non-comparison sorts:**
* **Counting sort:** When you are sorting integers with a limited range.
* **Radix sort:** When log(N) is significantly larger than the number of radix digits.
* Bucket sort: When you can guarantee that your input is approximately uniformly distributed. O(N+C)，C is in-bucket quick sort complexity, When M=N -> O(N), but space complexity is O(N+M)
* **Merge Sort**

****

* **Quicksort**



* **External Sort**
* required when the data being sorted can not fit into the main memory
* External sorting typically uses a hybrid sort-merge strategy.
* In the sorting phase, chunks of data small enough to fit in main memory are read, sorted, **and written** out to a temporary file.
* In the merge phase, the sorted sub-files are combined into a single larger file.
* <http://faculty.simpson.edu/lydia.sinapova/www/cmsc250/LN250_Weiss/L17-ExternalSortEX1.htm>
* Number of passes: log(N/M) (m = memory capacity)