CSE 332: Data Structures and Parallelism

Section 5: Sorting

0. Different Sorts of Sorting

For each of the following sorting algorithms, indicate if they are in-place and stable. If the option is implementation-dependent, ensure your choice maintains the expected runtime. Justify your decision.

(a) $\mathcal{O}(n^2)$

- **Insertion Sort**
- **Selection Sort**

(b) $\mathcal{O}(nlogn)$

- **Merge Sort**
- **Quicksort**
- **Heapsort**

1. LibrarySort

Project Gutenberg has another task for you: They want to sort all their unique-length books by length.

(a) If the longest novel is (according to Wikipedia) 2,070,000 words, and you didn't care about space efficiency, how fast could you sort their books? Could you do it in faster than $\mathcal{O}(nlogn)$ time? How?

(b) Let's now just consider the numbers involved in the previous situation. If we did the same method as in part (a) to just sort the numbers, in Java, roughly how much space would we end up using? Is there any way we could reduce it further? Even get it down below half a megabyte?