Lecture 02 Ticket

Name: _____

Consider the problem of sorting an array a of n numerical values. Suppose you can access and modify a through two methods:

- $\mathrm{compare}(a, i, j)$ returns true if $a[i] > a[j]$ and false otherwise
- $\mathrm{swap}(a, i, j)$ swaps the values $a[i]$ and $a[j]$ in the array. That is, if before calling swap we had $a[i] = x$ and $a[j] = y$, then after performing $\mathrm{swap}(a, i, j)$, we would have $a[i] = y$ and $a[j] = x$, and the other values in a would be unaffected.

A natural strategy for sorting a is the following. Scan through the array to find the index i_1 of the smallest value in a, then swap $a[i_1]$ with $a[1]$ so that the smallest value is at index 1 in the array after the the swap. Then do the same for the second smallest value in the array: find the index i_2 storing the second smallest value, and swap it with index 2 so that the second smallest value is stored at index 2 after the swap. Continue in this way until the array is sorted in increasing order.

1. Express the sorting procedure described above as a method $\mathrm{sort}(a)$ in pseudocode.
2. If the array a has size n, how many $\mathrm{compare}$ operations does your procedure use in the worst case? How many swap operations does your procedure use in the worst case? (Give as precise expressions as you can.)