Section 09: Sorting Algorithms

1. Sorting Algorithms Steps

Show the steps taken for each sort as we have learned in lecture.

(a) Insertion sort on `0, 4, 2, 7, 6, 1, 3, 5`.

(b) Selection sort on `0, 4, 2, 7, 6, 1, 3, 5`.

(c) Heapsort on `0, 6, 2, 7, 4`. (You may want to draw out the heap. Make sure the first step you do is the heapification step!)

(d) Merge sort on `0, 4, 2, 7, 6, 1, 3, 5`.

(e) Quicksort on `18, 7, 22, 34, 99, 18, 11, 4`. (Assume that we always choose first element as the pivot. Show the steps taken at each partitioning step.)

2. Sorting Decisions

For each of the following scenarios, say which sorting algorithm you think you would use and why. As with the design decision problems, there may be more than one right answer.

(a) Suppose we have an array where we expect the majority of elements to be sorted "almost in order". What would be a good sorting algorithm to use?

(b) You are writing code to run on the next Mars rover to sort the data gathered each night (Think about sorting with limited memory and computational power).

(c) You're writing the backend for the website `SortMyNumbers.com`, which sorts numbers given by users.

(d) Your artist friend says for a piece she wants to make a computer sort every possible ordering of the numbers $1, 2, \ldots, 15$. Your friend says something special will happen after the last ordering is sorted, and you'd like to see that ASAP.

3. Sorting Algorithm Analysis

(a) What are two techniques that can be used to reduce the probability of Quicksort taking the worst case running time?

(b) When choosing an appropriate algorithm, there are often several trade-offs that we need to consider. Complete the chart for the following sorting algorithms by writing down the best case and worst case runtimes in $\Theta()$ notation and whether or not the sort is stable. You may write any notes about the sort in the "Notes" column; this column will not be graded.

	Runtime (best)	Runtime (worst)	Stable? (Y/N)	Notes (not graded)
Selection Sort				
Insertion Sort				
Heapsort				
Merge Sort				
Quicksort (Naive)				