Precept Topics
- Comparators and Comparables
- Mergesort

Relevant Material
- Book chapters: 2.1, 2.2 and 2.5

A. RECAP: Elementary Sorts + Mergesort

Your preceptor will briefly review key points of this week's lectures, and introduce two new ones: stable sorts and the `Comparator` interface. Stay tuned!

B. EXERCISES: Comparable & Comparator

Solve the exercises in the "⚖️ Comparable & Comparator" Ed lesson, up to (and including) "Implementing Comparator". ("Stability Counterexample" is optional.)

C. EXERCISES: Three-way Mergesort

1) (Two-way) Mergesort is quite a simple algorithm to describe: to sort n elements, divide the array in half, (recursively) sort each then merge the two halves together. In this exercise, we will study a variant of it: in three-way Mergesort, we divide an array of length n into 3 subarrays of length $n/3$, sort each of them and then perform a 3-way merge.

(a) Given 3 **sorted** subarrays of size $\frac{n}{3}$, how many comparisons are needed (in the worst case) to *merge* them to a sorted array of size n? Provide your answer in tilde notation.

(b) What is the *order of growth* of the number of compares in 3-way Mergesort as a function of the array size n? (Here we're counting the total number, including all recursive calls.)

(c) Given a choice, would you choose 3-way or 2-way mergesort? Justify your answer.

2) Challenge Problem (optional): In an array `h` of n numbers, an *inversion* is a pair of elements that isn't sorted; that is, two indices `i` and `j` such that `i < j` and `h[i] > h[j]`.

Describe an algorithm to compute the total number of inversions of an array of length n in time $\Theta(n \log n)$. *Hint: think about how you can modify the merge sort algorithm to achieve this*.

D. ASSIGNMENT OVERVIEW: Autocomplete

Your preceptor will briefly go over the assignment due **in two weeks**. If you haven't yet and have some time to spare, take a look at the assignment page. Feel free to ask questions, but please leave implementation/debugging details for the lab TAs and office hours.