Precept Outline
- Course introduction.
- Review of Lectures 1 and 2.
- Overview of Assignment 1 (Percolation).

Relevant Book Sections
- 1.4 (Analysis) and 1.5 (Union-Find)
- 1.1 and 1.2 (Java review)

A. Introduction

We're pretty psyched for you to see what we've got in store, but, before we let you loose, here are just a few words about the format of precept in this course:

Precepts will be a mix of review, problem solving and discussion. Each precept will start with a short review of the lecture contents, followed by solving a mix of exercises, some of which will be written in this handout and some will be in Ed lessons. A lot of the exercises follow the same format as the ones you will find in the midterm and final exams, so precepts will be good practice for those.

The exercises are meant to be done in pairs. We want to encourage you to talk about the details of algorithms and data structures with a peer so you can help fill in each other's blind spots.

These exercises are not graded. You don't have to hand in any solutions, we won't grade any of your precept work. So you are encouraged to ask questions about the problems. The solutions to each exercise will be released after all precepts are done.

You are not expected to complete all of the problems in each handout. These handouts and the accompanying Ed lessons are intended for practice, you don't have to solve all of them during or after precept. In fact, it's very unlikely you'll go through the whole handout in any precept. Some of the problems are marked as "optional", which means they are outside of the scope of the course and are intended to be bonus challenge problems.

Attendance is mandatory. Your preceptor will keep track of your attendance (except for this first week), and your attendance will be 2.5% of your grade.

B. Review: Analysis and Union-Find

Your preceptor will briefly review key points of this week's lectures.

C. Analysis

Part 1: Experimental Analysis

Suppose that you collect the following timing data for a program as a function of the input size n.

n	$T(n)$
125	0.03 sec
1,000	1.00 sec
8,000	32.00 sec
64,000	1,024.00 sec
512,000	32,768.00 sec

Estimate the running time of the program as a function of n and use tilde notation. (Remember that we assume the *power law hypothesis*.)

Part 2: Mathematical Analysis

Runtime analysis can be tricky business; even short and simple-looking code can be hard to analyze correctly. The key to mastering this skill is practice, practice, practice. That's what we'll do in this part.

Solve Exercises 1 to 7 of the "Analysis Drills" Ed lesson.

D. Union-Find

Solve all the exercises in the "Union-Find" Ed lesson.

Challenge Problem (optional): Suppose you are given a sequence of n positive integers. Define a "group" as a contiguous subsequence of elements. The length of the group is the number of elements in it, and its value is its smallest element. (E.g., the sequence $(5, 3, 4, 1)$ has a single group of length 4 and value 1; two groups of length 3, with values 3 and 1; etc.)

For each $l = 1, \ldots, n$, determine the maximum value among all groups of length l. Your algorithm's runtime should be smaller than $O(n^2)$.

E. Assignment Overview: Percolation

Your preceptor will introduce and give an overview of your first assignment. Please don't hesitate to ask questions!