# Discussion 1 handout

Learning to program requires more than just acquiring knowledge. You need to actively **explore the implications** of new concepts to build mental connections, and you need to **practice writing programs** just like athletes and musicians need to practice their skills. Programming assignments provide the latter, while discussion sections are meant to stimulate the former.

Discussion implies *interaction*—to thoroughly explore these concepts, you need to formulate your thoughts into words, and you need to accommodate ideas from others in your evolving mental model. Therefore, discussion sections are structured around **cooperative exercises** to be completed in groups and synthesized by the class as a whole. This is the first such exercise.

## Requirements

- Form a group of 2-4 classmates seated near enough to allow discussion.
- Record the group's responses to each activity on a sheet of paper.
- Register your group in CMS, then upload a photo or scan of your work, along with code if applicable. The submitted paper must include the names and NetIDs of all group members and your responses to each problem discussed during section.

## Objectives

- Get to know classmates and section instructors
- Identify declarations, expressions, and assignments in Java methods
- Debug test failures in the IntelliJ IDE

## Problem 1: Introductions

Share your name & NetID, year & college, and a fun fact about yourself with your groupmates. Record the names and NetIDs of all group members.

1.
2.
3.
4.

Discuss the following two questions with your group, recording a representative answer:

**Why do you want to learn object-oriented programming and data structures?** (is it relevant to your intended major or career? did you enjoy your experience in your prior CS class? is there something particular you hope to accomplish with the knowledge?)

**What was the most difficult concept in your previous programming class?** (recursion? sorting? objects? arrays?) Does someone in the group have an example that helped the concept click for them?

## Problem 2: Semantics of the assignment statement

Your TA will review vocabulary related to method declarations.

**Write the procedure** for how you would execute a Java assignment statement such as `sideLength = perim / 4;`, or more generally, "*<var>* = *<expr>*;". Assume any necessary prior code so that the statement compiles. Select steps from the following choices, re-ordering them as necessary:

- Store the value in the variable on the left-hand side.
- Create the variable on the left-hand side.
- Allocate memory for the type of the variable on the left-hand side.
- Evaluate the expression on the right-hand side.
- Check if the value can be stored in the variable on the left-hand side.

1.
2.
3. (?)
4. (?)
5. (?)

Would your procedure change if you were executing a similar statement in Python (or MATLAB) rather than Java?

# Problem 3: Open and debug a Java project

If you have your own laptop with IntelliJ already installed, try to follow along as your TA demonstrates how to open a Java project in the IDE. If no one in your group has a laptop, you can attempt to debug the projected code "by inspection."

1. Download the "dis01" project archive from the course website and extract its contents to a folder on your computer.
2. Open the "dis01" directory as a project in IntelliJ. See our setup page for step-by-step instructions.
3. Read the method specifications in "src/cs2110/Dis01.java".
4. Run the `Dis01Test` test suite and note which test cases are failing.
5. For each method, try to determine where the bug is. If you have trouble spotting it, try printing the results of intermediate calculations to see if they match your expectations.
6. When you think you have identified the issue, fix the bug and re-run the test suite to confirm.
7. Reformat "Dis01.java" by going to *Code* | *Reformat Code*. You are now ready for submission.

# Submission

1. Open the assignment page for "Discussion activity 1" in CMSX
2. [Recorder] Find the "Group Management" section and invite each group member
3. [Others] Refresh the page and accept your invitation
4. [Recorder] Take a picture of your work and save as either a JPEG or a PDF file named "discussion_responses". *After all invitations have been accepted*, upload your picture along with your code as your group's submission.
    - Recommended scanning apps: Microsoft Office Lens, Adobe Scan, Genius Scan, Evernote Scannable

Ensure that your group is formed and your work submitted before the Friday evening deadline.

# Tips and reminders

- Discussion is not a race. Focus on the current activity being facilitated by your TA and engage your whole group to propose and explain ideas.
- Elect a recorder to maintain the "master" copy of your work (others are still encouraged to jot down ideas on scratch paper). Rotate this position each week.
- It is a violation of academic integrity to credit someone for work if they were not present when the work was done, and the whole group is accountable. Your CMS group must only include classmates who attended section with you on that day. Remember that our participation policies accommodate occasional absences without penalty.
- It is your individual responsibility to ensure that you are grouped on CMS and that your group's work is submitted before the deadline. Log into CMS the day after section to ensure that everything is in order, and contact your groupmates if not. It would be prudent for multiple members to photograph the group's work.
- Only one group member (typically the recorder) needs to upload a submission. But their submission must not be uploaded until after all group members have confirmed their membership in CMS (contact your TA if you have trouble grouping in CMS).

---