Guide to Cheating and Academic Integrity: see link here outlining the academic integrity guidelines for this class

Each midterm typically has an end-user license agreement (EULA) that you must agree to. You will save time if you review this ahead of time and ask any questions before exam day. (It’s better to be safe than sorry!) Typically, the EULA asks you to verify that you will not distribute your work or any material/questions within the exam document to anyone.

Quick cheat sheet for academic integrity:
- Asking other students/TAs/Prof Donald general questions about the homework (or concepts covered in the HW) that clear up misunderstandings (✔)
- Asking other students for their code or copying off another student’s paper (🚫)
- Asking clarifying questions about what will be on the exam (✔)
- Attempting to find out (through any means) ahead of time what specific types of questions will be on the exam (🚫)

Office Hours Tip: It is a good idea to review problem sets/SAs ahead of time and come to office hours with a list of detailed questions. It also helps TAs help you more effectively (they don’t have to guess your question for you). Please avoid asking the TAs to solve the problem for you — it delays your ability to learn!
What do I actually have to know for Midterm 1?
- Substitution Model aka S.M. (be able to answer ‘what will Scheme return?’)
- Important operations such as:
  - map
  - filter
  - accumulate / reduce
  - append
  - cons
  - cond
  (Be familiar with the above operations; make sure you can use S.M. to understand what they will return, even if you are given a slightly different procedure from these)
- How to write a proof by induction
  - Claim
  - Base case (with substitution model, S.M.)
  - Induction Hypothesis (I.H.)
  - Inductive Step (I.S.) - must utilize I.H. and (often) S.M.!
- Implementing a function in Scheme and analyzing its run-time (simple Big-O, recurrence relations)
  - How to write code on paper, by hand
  - How to write a recurrence relation based on a function, and solve it for overall run-time

Any tips for each of these topics?
- Have a deep understanding of the S.M. as well as the important operations listed above:
  - Memorize and deeply understand the source code for the important operations (and make sure you can recall them from memory so you can use them when implementing your own functions)
    - Example: Start with a blank sheet of paper. Write the function for accumulate in Scheme. Then run this function on a list of numbers to check that it works.
- Proof by induction
  - You can simplify the induction process by writing $P(n)$, the property of induction:
    - Example: I.H. becomes ‘Assume $P$ holds for all numbers $k$ less than $n$’
      I.S. becomes ‘N.O.S. $P$ holds for argument $n$’
  - Determine in which situations it is easier to use strong induction vs. weak induction
- Implementing a function in Scheme and analyzing its run-time
  - Think of each operation (filter, map, etc.) as tools in your tool box that you can use to write more complicated functions. In which situation should you use accumulate? What about filter? Consider this carefully.

How should I study for the exam?
- DO reread the lecture notes and handouts (redo any example problems you encounter!)
- DO review the SAs and problems from recitation.
- DO review the recommended SICP problems.
- *DO* ask for the blessing of the lambda gods the night before the exam.