This recitation is about concurrency: what it is and how programmers handle it. This recitation is also about a pair of papers. The goal of these two papers is to eliminate a subset of the bugs that arise when programmers must handle concurrency.

OVERVIEW

1. Overview of concurrency, threads, and locking.
2. What problem is Eraser solving?
3. At a high-level, what is the Eraser solution?
4. What problem is RaceTrack solving?

REFLECT ON ERASER AND RACETRACK

The next set of questions asks you to reflect on the positive and negative points of Eraser and RaceTrack. What follows are specific aspects on which you may want to focus, but note that you may not be able to answer all of these for both papers because the entire papers were not assigned:

A. The algorithm. Does it find all bugs? Is everything it finds a bug?
B. The results. (Also known as "the evaluation"). Good? Bad? Convincing?
C. The quality of the writing. Good? Bad?

5. What are the good points about Eraser? Please be specific; give examples.
6. What are the bad points about Eraser? Please be specific; give examples.
7. What are the good points about RaceTrack? Please be specific; give examples.
8. What are the bad points about RaceTrack? Please be specific; give examples.

DISCUSSION AND LESSONS

TAKE-HOME POINTS

--Concurrency "hard". Prone to bugs. Be careful.

--For tools like Eraser and RaceTrack, we must read experimental evaluation to know how useful they are.

--There is a "do use threads" vs. "don’t use threads" debate in both research and industry. Each side presents trade-offs.