Homework 2

Please submit your solutions via NYUClasses. Solutions to programming exercises should be submitted as plain text files. No exotic formats, please!

The deadline for Homework 2 is February 10, before class.

Problem 1 AMP, p.41, Exercise 11: Flaky Lock (12 Points)

Programmers at the Flaky Computer Cooperation designed the protocol shown in Fig. 1 to achieve n-thread mutual exclusion. For each question, either sketch a proof, or display an execution where it fails.

- Does this protocol satisfy mutual exclusion?
- Is this protocol starvation-free?
- Is this protocol deadlock-free?

```
class Flaky implements Lock {
  private int turn;
  private boolean busy = false;
  public void lock() {
    int me = ThreadID.get();
    do {
      do {
        turn = me;
      } while (busy);
      busy = true;
    } while (turn != me);
  }
  public void unlock() {
    busy = false;
  }
}
```

Figure 1: The Flaky lock

Problem 2 AMP, p.41, Exercise 14: ℓ-Exclusion (13 Points)

The ℓ-exclusion problem is a variant of the starvation-free mutual exclusion problem. We make two changes: as many as ℓ threads may be in the critical section at the same time, and fewer than ℓ threads might fail (by halting in the critical section).
An implementation must satisfy the following conditions:

- ℓ-Exclusion: At any time, at most ℓ threads are in the critical section.
- ℓ-Starvation-Freedom: As long as fewer than ℓ threads are in the critical section, then some thread that wants to enter the critical section will eventually succeed (even if some threads in the critical section have halted).

Modify the n-process `Filter` mutual exclusion algorithm to turn it into an ℓ-exclusion algorithm.