

More Project Info

- Congrats on living through your first design document!
 - o I will get them back to you ASAP.
- Next up:
 - o Design review signups will go online today, on my section homepage.
 - Design reviews will be held Monday and Tuesday.
 - Location TBD
 - Each design review will be 20 minutes.
 - You will present your solution to me on the board.
 - I will answer questions and make sure you're on the right path.
 - These are graded! Criteria:
 - o EVERYONE MUST ATTEND.
 - Unless someone submits a satisfactory reason for absence, everyone loses points!
 - o EVERYONE MUST UNDERSTAND THE PROBLEMS AT HAND
 - I reserve the right to ask questions of anyone on anything (project related)
 - o Meet these criteria, and full credit regardless of how many things we discuss. ☺
 - I hope to return your design document to you after your review.
 - o Monday Feb 23rd – Tuesday Feb 24th: Design reviews
 - o Thursday, March 4th: Project 1 code due
 - Submitted online
 - o Friday, March 5th: Project 1 final design documents due
 - Submitted online
 - These will consist of your REVISED design document and answers to questions in the project handout.
- **Unsolicited advice: Don't use your slip days right off the bat!**

Today's Menu:

- Working in groups
- Synchronization problems
- Discuss answers to synchronization problems

Working in Groups

- How to divide up work? Many ways to do this. Just remember:
 - o Sections NOT of equal difficulty.
 - o One problem might touch multiple areas of Nachos.
 - o Divide and conquer is useful only if you consider how things fit back together.

- Two people working on the same problem may not write code twice as fast, but having someone as a sounding board catches more problems.
- How to keep from holding your group members back if their work depends on yours?
 - Consider writing function prototypes (“stub functions”) that return a known value so that they can develop alongside you
 - Then fill in the prototypes.

Testing

- Unit testing
 - Include a testing function with each class that tests module functionality.
 - Can save you serious time when playing ‘hunt the bug’!
- Integration testing
 - MAKE SURE, when putting everything together, that you leave time to test the integration!
 - Even better: integrate on a rolling basis, instead of all at once.
 - Test as you go.
- Glass Box Testing
 - A glass box test is one which is written with some knowledge of the implementation of a piece of code.
- Black Box Testing
 - A block box test is one which only tests the given interface to some functionality.

Testing Examples

- PRODUCER/CONSUMER

```
Lock lock;
Condition wantToAdd = new Condition(lock);
Condition wantToTake = new Condition(lock);
```

```
Producer() {
    lock.acquire();
    while (numCokes == MAXCOKES) {
        wantToAdd.sleep();
    }
    numCokes++;
    wantToTake.wake();
    lock.release();
}
```

```
Consumer() {
    lock.acquire();
    while (numCokes == 0) {
        wantToTake.sleep();
    }
}
```

```

    }
    numCokes--;
    wantToAdd.wake();
    lock.release();
}

```

- **QUESTION:** If implementing this code under Nachos, write down some glass box tests.
- **ANSWER:** Several, but one would be to set the Alarm timeout to 1 unit so that yield is called after `_every_` instruction.

- **QUESTION:** Write down some black box tests for the producer/consumer code.
- **ANSWER:** Several, but here's one: create varying number of producers and consumers and verify that the correct number of consumptions happen and correct number of thread blockings happen.
 - o For example, create x producers and x consumers and all x threads should finish. Create $x - 2$ producers and x consumers and $x-2$ consumers and all producers should finish. etc.

- **QUESTION:** What are the corner cases for the producer/consumer code? How can you devise a test to test the corner cases?
- **ANSWER:** No consumers. Create a test with multiple producers and no consumers. No producers. Create a test with multiple producers and no consumers.

Quiz!

- (in groups this time—have them move their chairs)

Discuss Results