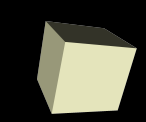
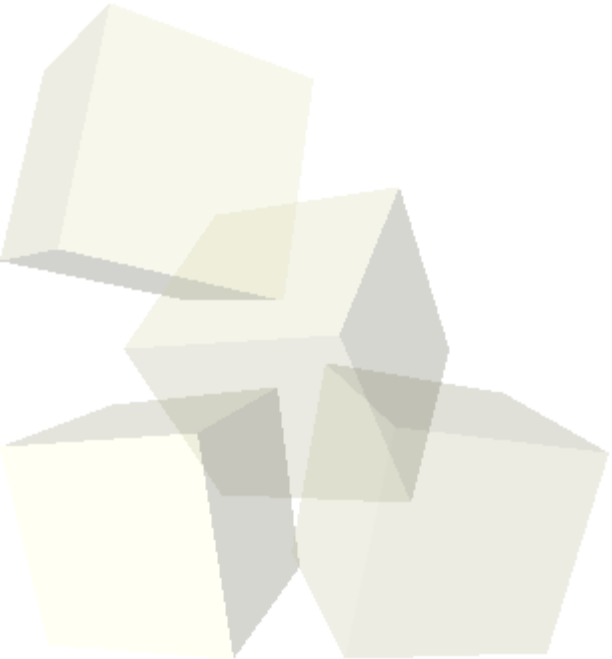


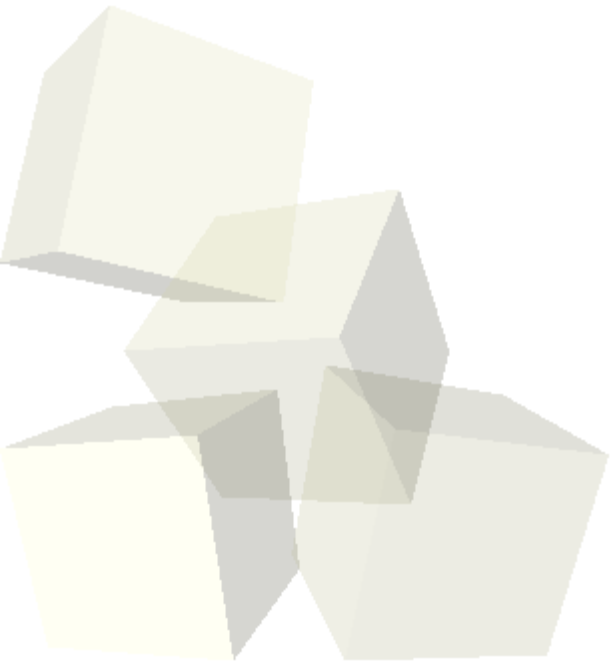
Stacks and Queues

2/13/2007



Opening Discussion

- Do you have any questions about the quiz?
- Let's look at why the threading code we did last time had been slow. (It's because we were using the same comparator that counted comparisons.)
- Do you have any questions about the reading?
- Do you have any questions about the assignment?

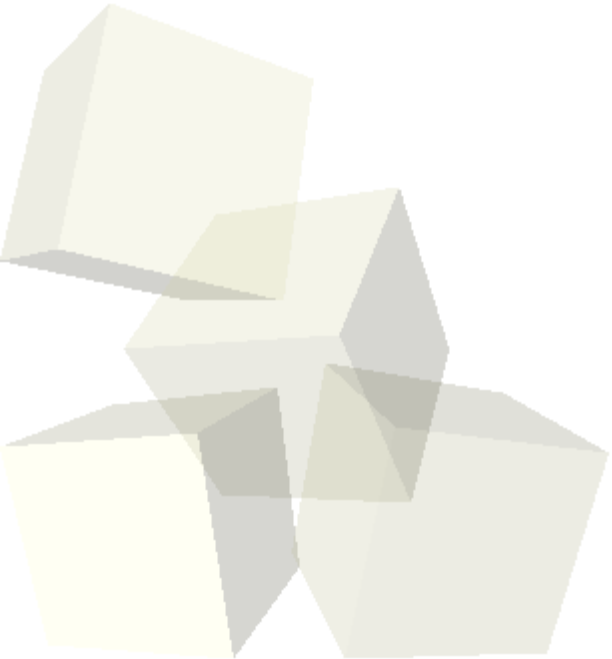


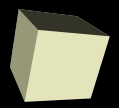
Array Based Stack

- I want you to write an interface called MyStack with the methods we said should be in it last class. Make the interface generic so it can handle any type.
- Then write a class called ArrayStack and make it implement MyStack. Fill in the code for ArrayStack and add a main method to test that they work.

Array Based Queue

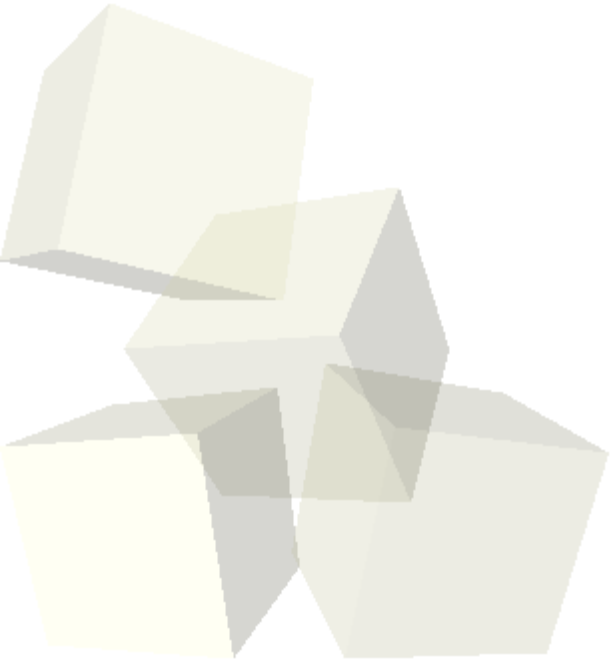
- Now we will do the same thing for a queue. Make a MyQueue interface and an ArrayQueue class.





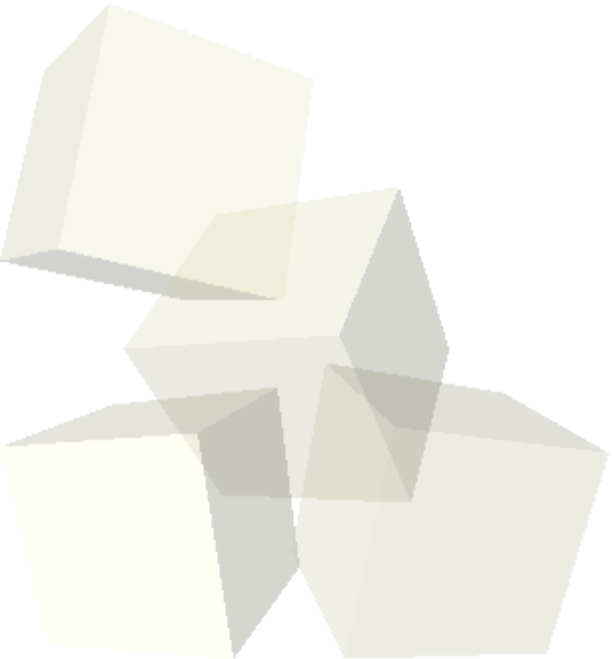
The List ADT

- The next step up the ADT ladder is the list. Basically a list provides general access so you can add, remove, or search for things at random locations in the list.
- Java has an interface called List in java.util. Let's go look at that.



An Array Based List

- So how could we implement the list interface using an array?
- What methods of that implementation would be “fast”? Which ones would be “slow”?
- What do the terms fast and slow mean here in O terms and what operations are being considered for that?



Linked Lists

- There is an alternate method of implementing the list interface called the linked list. It is strong where an array list is weak, but weak where an array list is strong.
- A linked list is made of nodes and each node knows about one or two of its neighbors (has pointers to them).
- We move around linked lists by “walking” from node to node.
- Adding and removing can be very fast and always require very few memory writes.

Minute Essay

- We will re-implement the MyStack interface latter on using a linked list for the implementation. Can you describe how we might do that?
- There is nothing due Thursday. The design for assignment #3 is due in a week.