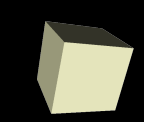
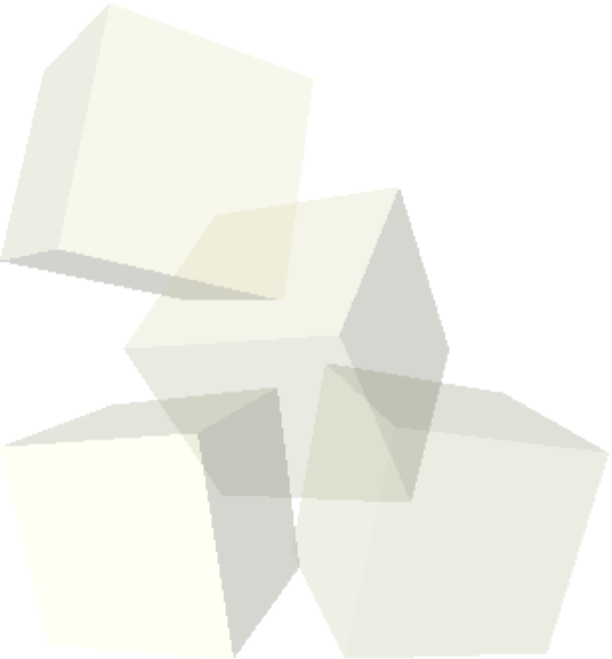


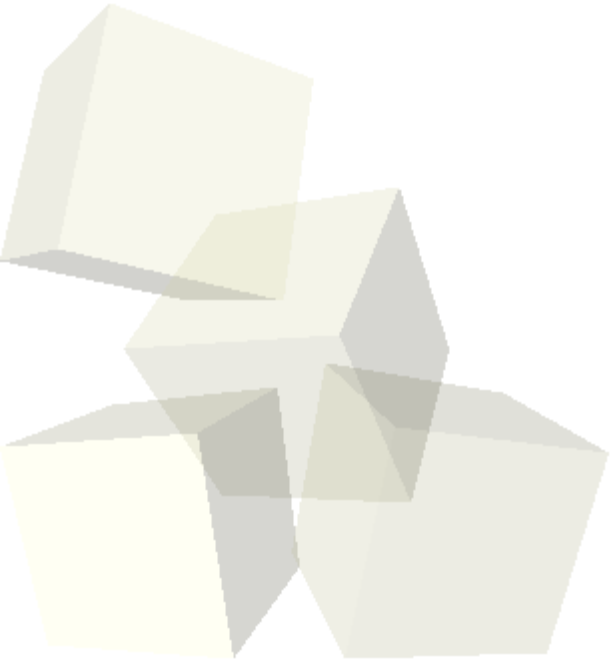
Stacks and Queues

2/24/2009



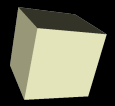
Opening Discussion

- Do you have any questions about the quiz?
- Let's look at solutions to the interclass problem.
- Do you have any questions about the reading?
- Do you have any questions about the assignment?
- Relevance of speed in modern computing.



Abstract Data Types (ADTs)

- Today we will be working with the simplest forms of abstract data types. These are things that hold data and specify how you can interact with it and what happens when data is added or removed.
- In Java an ADT is basically an interface for a container with comments giving details on what happens with each method.
- Note that it doesn't specify how things happen. That is why it would be an interface. ADTs can be implemented in many different ways.



Stacks and Queues

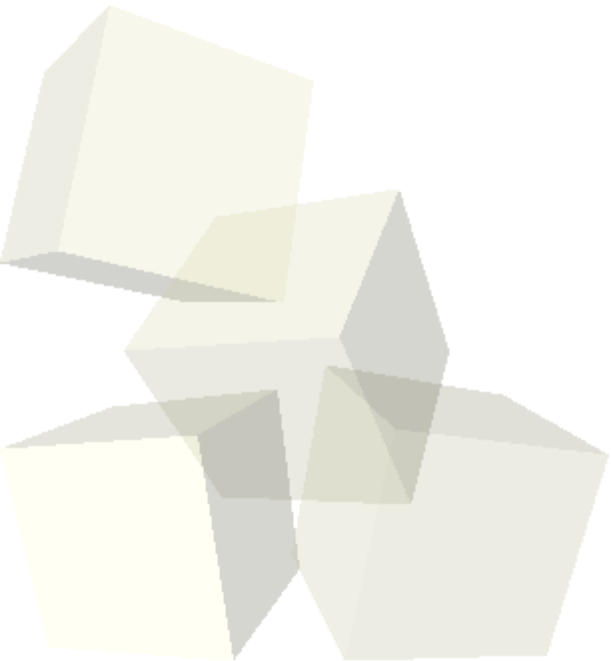
- The simplest forms of ADTs, they each require one method to add an element and one method to remove an element. For easy of use we typically also include two other methods.
- Methods of a stack
  - push
  - pop
  - peek
  - isEmpty
- Methods of a queue
  - enqueue
  - dequeue
  - peek
  - isEmpty

The Difference?

- Push and enqueue add items while pop and dequeue remove items. The difference is what item gets removed.
- A stack is last in, first out (LIFO). Just think of how you interact with a stack.
- A queue is first in, first out (FIFO). If you were British you would use the term queue instead of line for what you stand in when waiting for something.

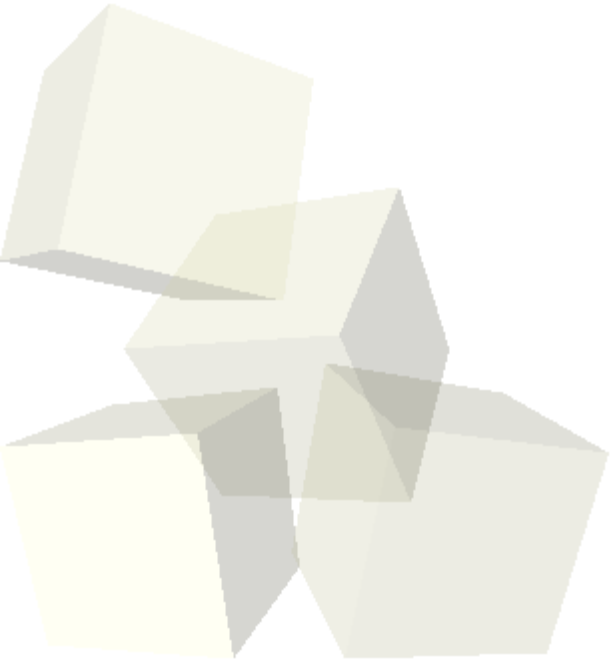
Array Based Stack

- Let's write an interface called MyStack with the methods we said should be in it. Make the interface generic so it can handle any type.
- Now let's 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.



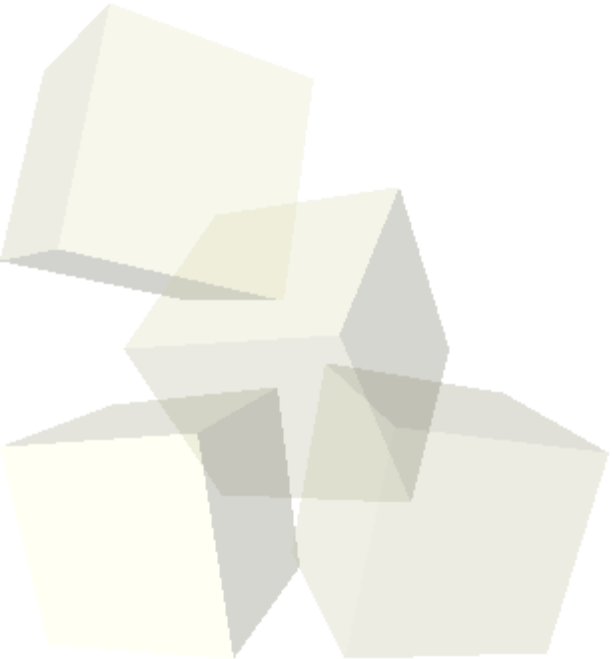
Array Based Queue

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



Other Code

- One of the most standard applications of a stack is a reverse polish calculator (RPC).
- Let's make a class for a RPC then make a command that will use it.



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?
- Remember that design #3 id sue on Thursday.
- Interclass problem – Edit the calculator GUI that you have made so that it implements an RPC calculator. You might want to use a TextArea to display the stack.