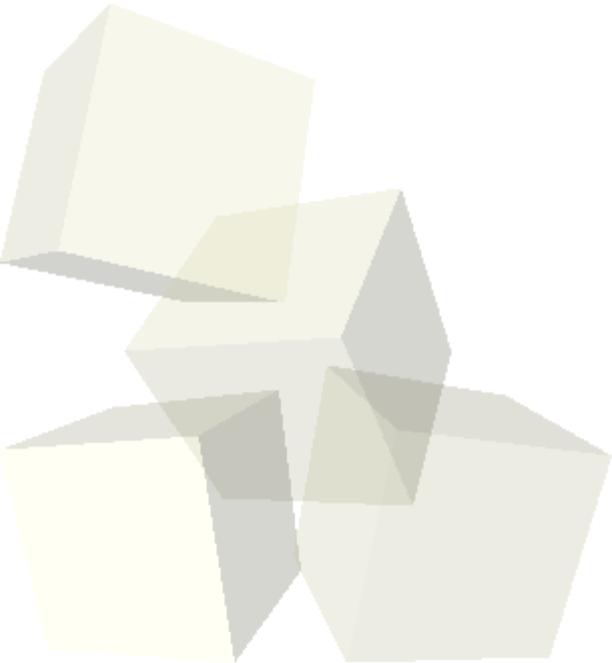
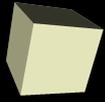




Linked Lists

2/21/2008

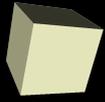




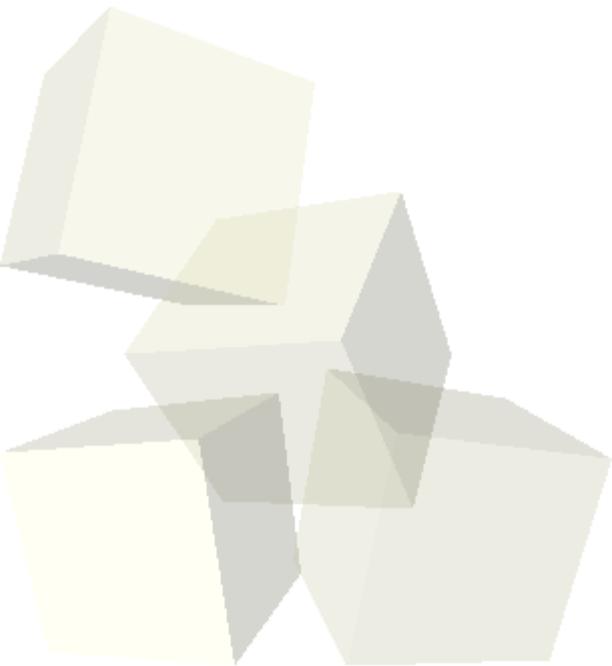
Opening Discussion

- Do you have any questions about the quiz?
- Let's look at some solutions to the interclass problem.
- Do you have any questions about the assignment or the readings?
- Let's finish our array based queue quickly.
- What is a list? What are the things that you can do with a list?





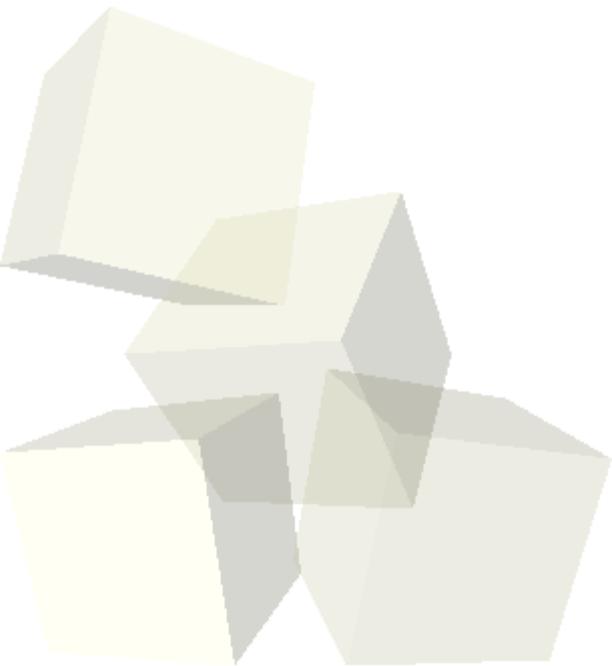
- The next step up the ADT ladder is the list. Basically a list provides general access so you can add, remove, or search for things at random locations in the list.
- Java has an interface called List in `java.util`. Let's go look at that.





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?



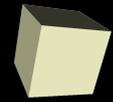


- There is an alternate method of implementing the list interface called the linked list. It is strong where an array list is weak, but weak where an array list is strong.
- A linked list is made of nodes and each node knows about one or two of its neighbors (has pointers to them).
- We move around linked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



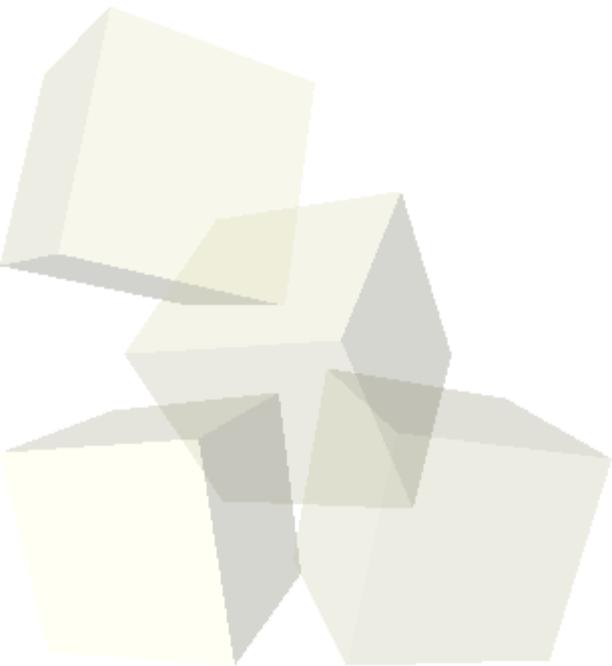
Types of Linked Lists

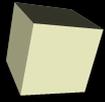
- Linked lists can be implemented in many ways. The basic characteristic is that we only keep a reference to one node and nodes then link to one another.
- The linking can be single or double. A doubly linked list has nodes that know about both the next and the previous elements.
- Linked lists can also be circular. In a circular linked list, the first element links around to the back one.
- For optimization purposes, lists can keep track of a head and a tail, but that isn't required.



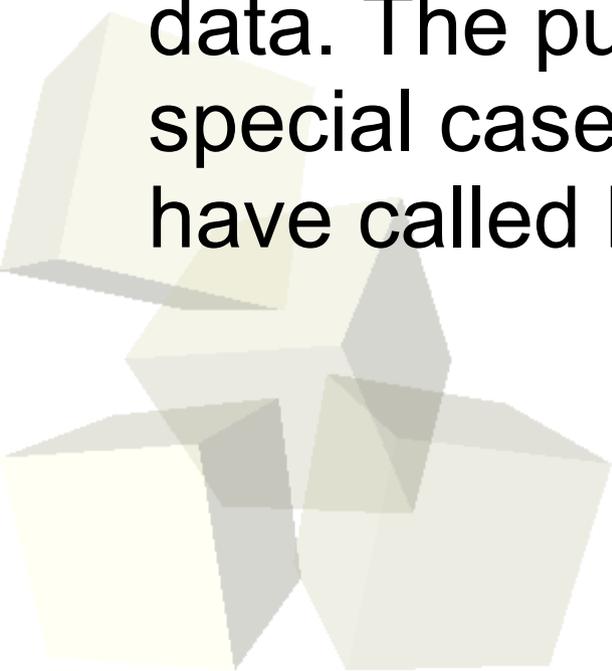
Implementing a Singly Linked List

- Let's work together to build an implementation of a singly linked list.
- We will implement the `java.util.List` interface, but we won't have time to implement all of the methods.



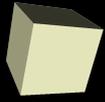


- Your book refers to an extra node placed at the beginning of a linked list as a dummy node. These are also called sentinels and your book understates how much they can improve your life. They also don't do them quite right.
- A sentinel is an extra node in the list that represents the “end” of the list and doesn't store data. The purpose of the sentinel is to remove special cases. The next of the sentinel is what we have called head.

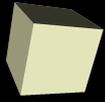


Implementing a Doubly Linked List

- Now let's implement `java.util.List` with a doubly linked list with a sentinel. The list will also be circular.
- You should notice that this implementation never has to check for null because no references in the list should ever be null. This simplifies the code significantly. We also implicitly get a head and a tail with no extra work. If you don't have a sentinel you will write a lot of extra checks for nulls and even more to include a tail.

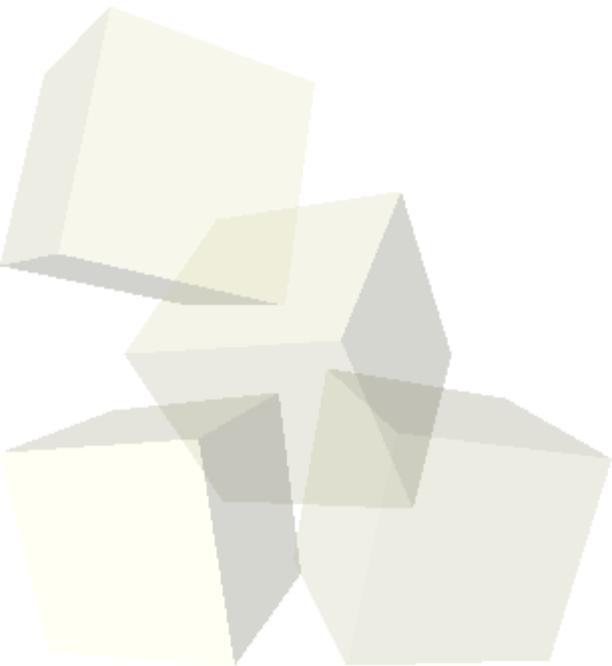


- We've already discussed that direct access on linked lists is very inefficient. How then should we walk through a list with outside code? Remember that the outside code doesn't have access to the nodes so it can't use the style of loop we have been doing internally.
- The concept of an Iterator is something that abstracts the process of walking through all of the elements in a container. Iterators can not only be efficient, they also make code more flexible because they don't depend on the implementation details of the containers.



Iterating Lists

- An iterator basically needs to encapsulate the information and functionality we would put into a standard method of going through a container.
- With this in mind, what do we need to put in an iterator for an array based list?
- What would we put in an iterator for a linked list?





- For one of your future assignments you will be forced to write your own linked list. What type of linked list do you think you will write?
- Remember that the design for assignment #3 is due on Tuesday.
- Interclass problem – Write a simple linked list that you can use for the memory function of our graphical RPC. Have a main that puts some numbers on your list then pulls them off.

