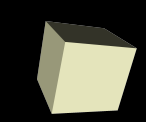
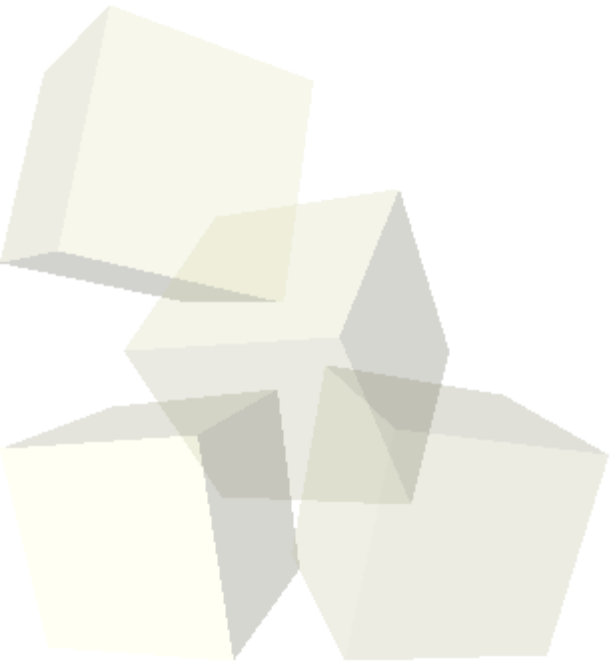


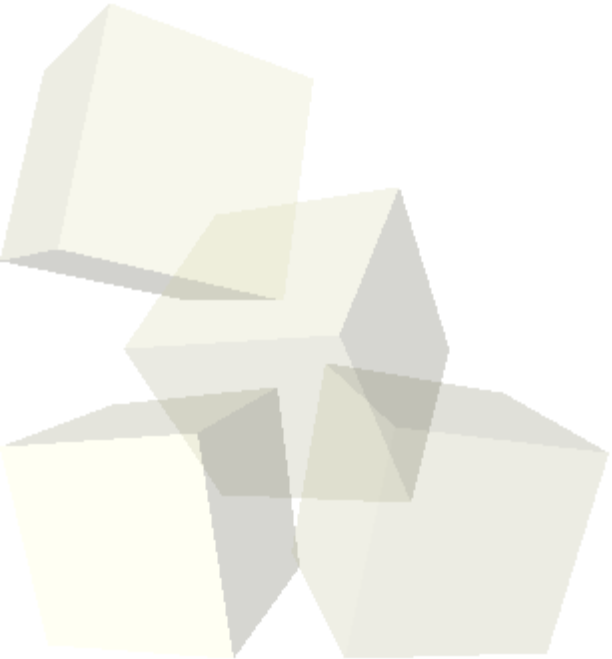
Bitwise Operators

11-28-2006



Opening Discussion

- Do you have any questions about the quiz?
- Do you have any questions about the assignment?
- What did we talk about last class? Let's go ahead and finish up the code we started last class.



Bitwise Operators

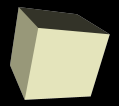
- Since the numbers on computers are stored in binary, it makes sense that there would be special operators that interact with the data in that format. These are called bitwise operators.
- Bitwise boolean operators (&, |, ^, ~) do boolean operations, but on the bits in an int instead of treating the whole int as a boolean.
- Bit shifting operators (<<, >>) do what the name implies. In decimal the equivalent would be multiplication and division by powers of 10. In binary it is multiplication and division by powers of 2.

Use of Bitwise Operators

- Bitwise operators are used for two primary purposes: flags and packing values into fewer bits.
- Those who have done the graphics options have seen three small integer values packed into the bytes of an int. This is very common in graphics.
- Packing used to be even more common when memory was more limited. It a value only had 4 possibilities you would only use 2 bits for it if you could find something to do with the other 6 bits in a byte.
- Many C functions use bitwise values for flags. This is basically packing booleans in single bits.

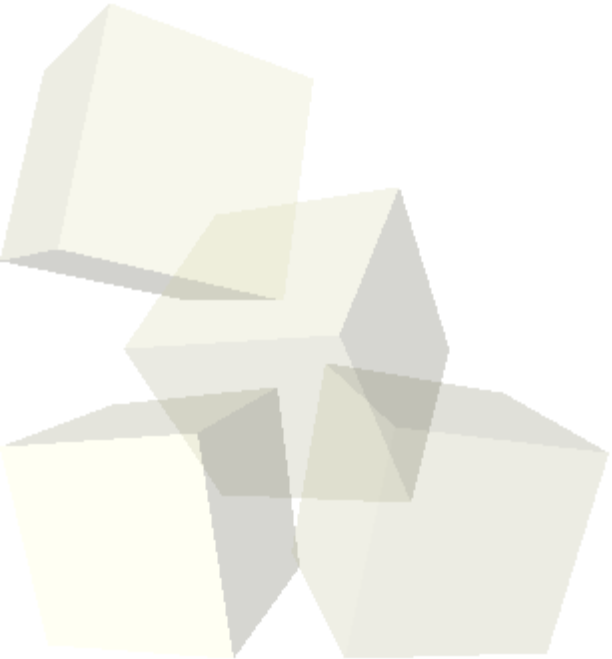
The List ADT

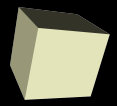
- ADT stands for Abstract Data Type. It is basically a collection of data with a set of functions that you are allowed to do on the data.
- The list ADT is just like your mental concept of an ordered list. You have the ability to add or remove things at different positions.
- In C we would implement this by writing a structure that stores the data for the list and a set of functions that operate on that structure. Note that outside code doesn't have to know what is actually in the structure.



Arrays as Lists

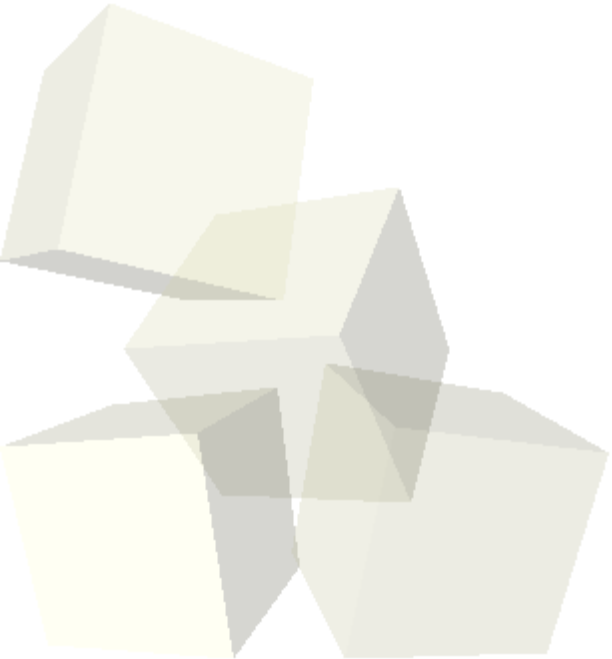
- Let's look at how we would implement a list using arrays. The methods we want to implement are add, insert, remove, and get.
- Note that we can't leave holes in the array of our list.





“Bad” Operations on Array Lists

- Hopefully it is clear that the code for the array based list had to do a significant amount of work in order to do the insert and remove operations.
- These operations basically require a large number of copies which is very slow.
- The number of copies scales as the number of things in the list. We refer to this as O(n) behavior.



Linked Lists

- A linked list is an implementation of a list that is intended to get around these problems for applications where we are frequently adding and removing data.
- While an array is a single chunk of memory and we know where all of it is, a linked list is make of little chunks of memory and each one knows where one or two other chunks are.
- We will focus on a singly linked list where each element knows about the the next element in line.
- This will be the topic of our next class. You should read up on it first.

Minute Essay

- What do & and | represent logically in C? When would you use each of them?