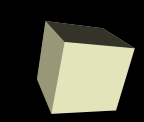
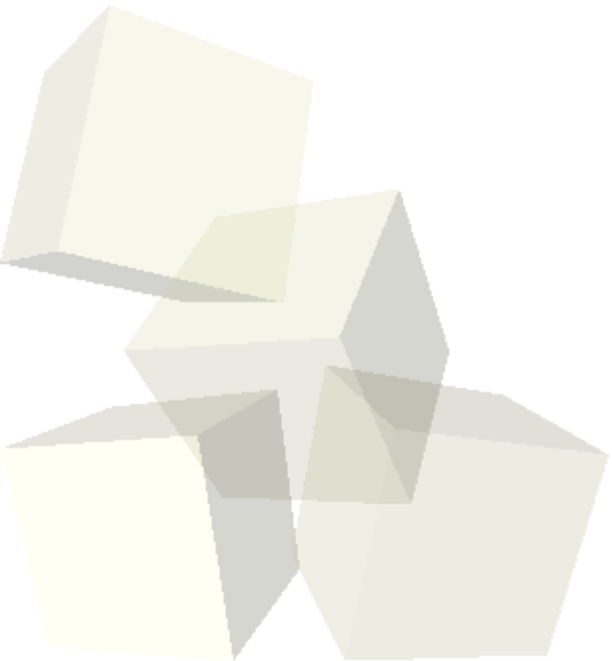


Linked Structures

1-17-2006

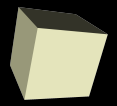


Opening Discussion

- Do you have any questions about the reading?
- Do you have any questions about the assignment?
- Do you have any questions about the test?

Linked Structures

- This term might conjure an image of a linked list into your head, but it is far more broad than that. A linked list is basically the simplest form of linked structure. The full class includes trees and graphs as well. Basically, it is all data structures that have different linked components.
- A flat array is not a link structure, but you can emulate linked structures with arrays. This is actually very helpful in some languages or if you want to dump data structures to file.



Two Types of Data Structures

- There are two main categories of data structures that we need to consider. Which one want to produce is very significant when you are making your design.
- The real question is whether you have control over where items go in the structure. If you get to place things by an index that is fine. If you can't, you typically are dealing with a map or keyed data structure where items are stored based on some key value.
- Every data structure you talk about can be made as a map though the ones you use most typically don't make much sense that way.

Linked List Review

- You should all remember the primary terminology for linked lists and the types we might make.
  - singly linked, doubly linked, circular
- The list interface can be implemented using arrays as well.
- What are examples of when a list is being used as a mapped data structure? (The hint here is that you can't specify where things are placed. Notice that is a very fundamental change in the interface of the data structure.)
- Sentinals can make life much nicer.

Trees

- Trees are a subset of directed graphs where one node/vertex, the root, has 0 incoming edges and all other nodes/vertices have 1 incoming edge. In a tree there are no cycles and there is exactly one path from the root to any other node in the tree.
- Trees are probably the favorite data structure of CS because they are remarkably flexible, yet have enough limitations to be manageable. Tree type structures also provide the hierarchical arrangement of data that humans seem to naturally gravitate towards.

Binary Search Trees

- In the case of data structures, the tree type we care about the most if the BST. This is a mapped data structure sorted by some key value with lower keys going to the left and greater keys going to the right.
- As with all linked data structures, adding and removing is relatively fast. The binary nature of a BST also allows for logarithmic time algorithms for searching out certain keys.
- BSTs are well behaved on random data. They degrade to linked lists on ordered data.

Balanced Binary Trees

- To keep thing efficient, we need to keep the trees well balanced. Two ways of doing this are with AVL trees and red-black trees.
- In both of these we occasionally rotate data around to preserve some type of balance condition.
- Let's talk about single and double rotations and when we would do them in an AVL tree.

Reminders

- The test is due when you come to class for next class.