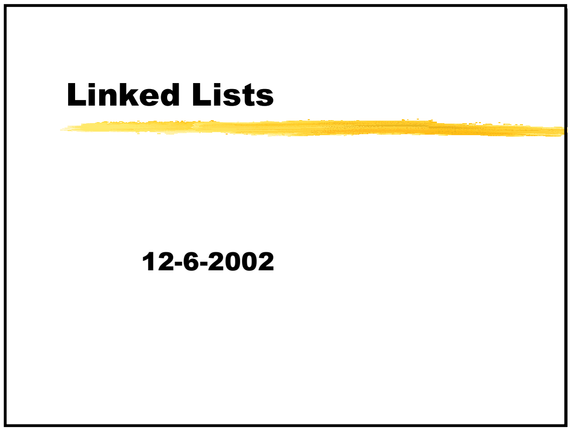
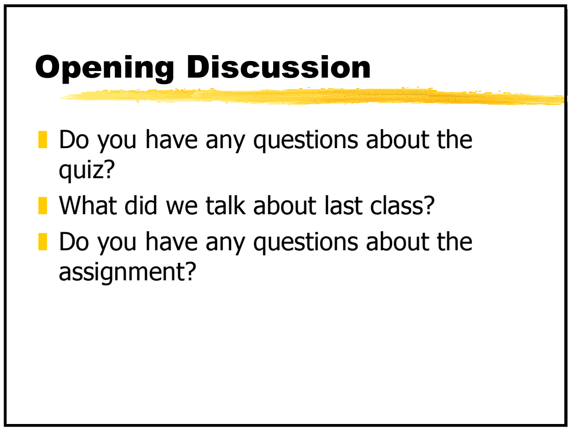


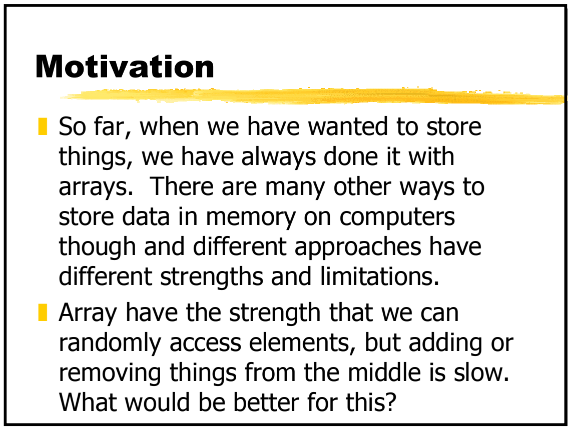
Linked Lists

12-6-2002



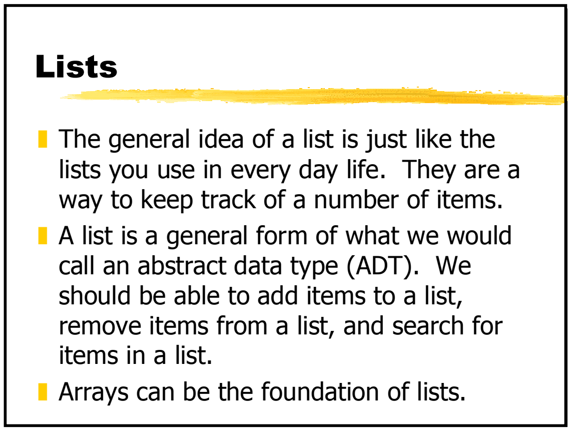
Opening Discussion

- Do you have any questions about the quiz?
- What did we talk about last class?
- Do you have any questions about the assignment?



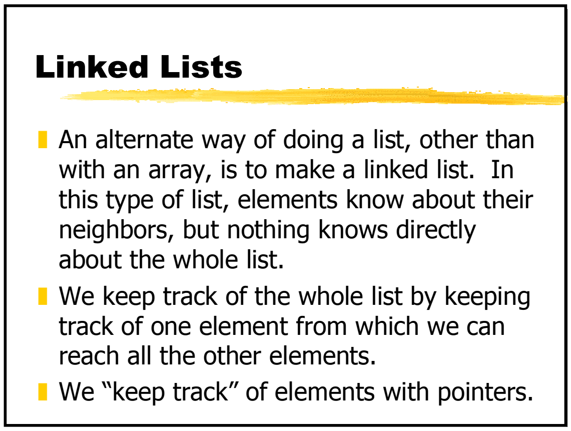
Motivation

- So far, when we have wanted to store things, we have always done it with arrays. There are many other ways to store data in memory on computers though and different approaches have different strengths and limitations.
- Array have the strength that we can randomly access elements, but adding or removing things from the middle is slow. What would be better for this?



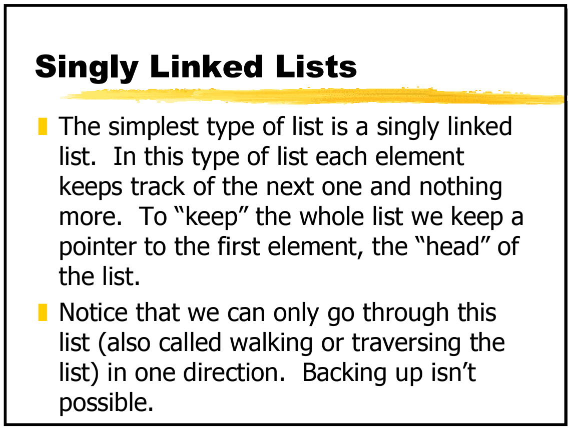
Lists

- The general idea of a list is just like the lists you use in every day life. They are a way to keep track of a number of items.
- A list is a general form of what we would call an abstract data type (ADT). We should be able to add items to a list, remove items from a list, and search for items in a list.
- Arrays can be the foundation of lists.



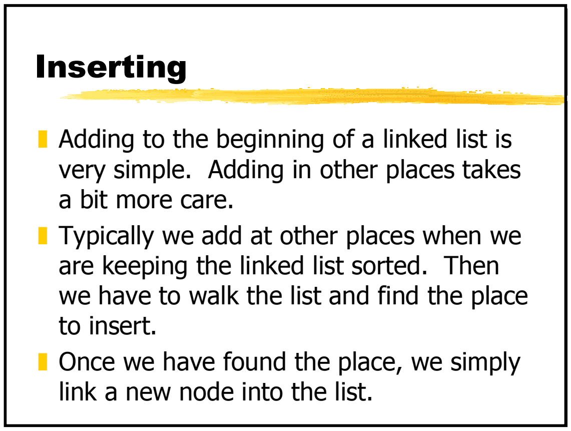
Linked Lists

- An alternate way of doing a list, other than with an array, is to make a linked list. In this type of list, elements know about their neighbors, but nothing knows directly about the whole list.
- We keep track of the whole list by keeping track of one element from which we can reach all the other elements.
- We "keep track" of elements with pointers.



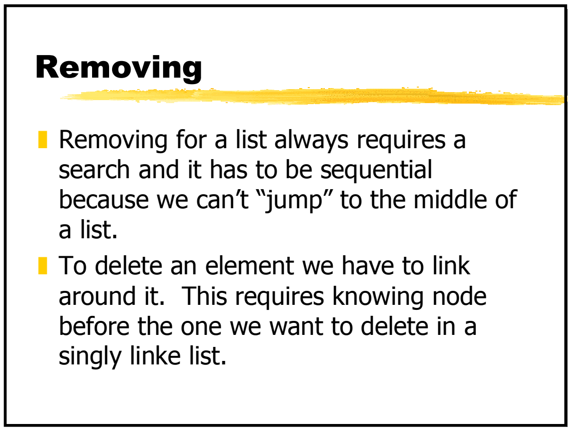
Singly Linked Lists

- The simplest type of list is a singly linked list. In this type of list each element keeps track of the next one and nothing more. To "keep" the whole list we keep a pointer to the first element, the "head" of the list.
- Notice that we can only go through this list (also called walking or traversing the list) in one direction. Backing up isn't possible.



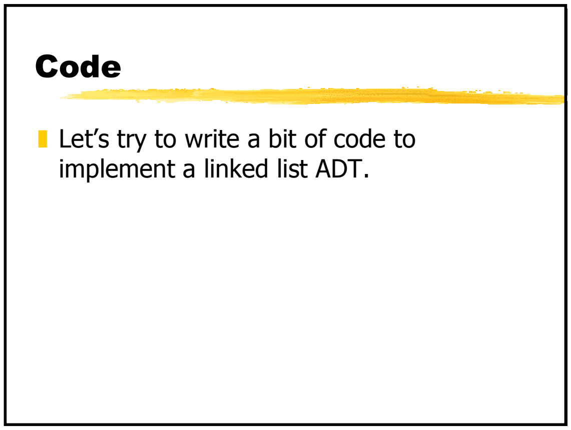
Inserting

- Adding to the beginning of a linked list is very simple. Adding in other places takes a bit more care.
- Typically we add at other places when we are keeping the linked list sorted. Then we have to walk the list and find the place to insert.
- Once we have found the place, we simply link a new node into the list.



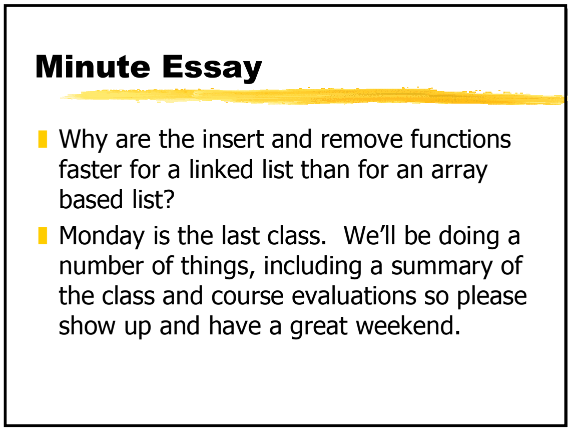
Removing

- Removing for a list always requires a search and it has to be sequential because we can't "jump" to the middle of a list.
- To delete an element we have to link around it. This requires knowing node before the one we want to delete in a singly linke list.



Code

- Let's try to write a bit of code to implement a linked list ADT.



Minute Essay

- Why are the insert and remove functions faster for a linked list than for an array based list?
- Monday is the last class. We'll be doing a number of things, including a summary of the class and course evaluations so please show up and have a great weekend.