Announcements

- Lab will be on Wednesday this week!
- HW due Tuesday by midnight
- Ch. 3 photo copies are at front of room
- Textbook is now available!
Recap of Arrays (Ch 3.1 of text)

- Limits:
  - Not at all flexible
  - Size is fixed
  - Inserting and moving is difficult

- I data type
- Access is unrestricted
Singly Linked Lists

A collection of nodes that together form a linear ordering.

head = head → next → next…

Memory

```
head
  | next
  | data
  |
LAX

head
  | next
  | data
  | 263
ORD

head
  | next
  | data
  | 128
MSP

head
  | next
  | data
  | 263
STL

head
  | next
  | data
  | 128
BDS

(head) (null)
```
Functions (very stripped down version)

What might we want to do?

- insert front
- delete front
- front — return a reference to — data
- Constructor
- Destructor
- empty
Implementation - Nodes

We want a node to store two things

- data
- pointer to next

template
Code: Need our node class

\[ E \text{ (in textbook)} \]

```cpp
template <typename Object>
class SLinkedList {

private:

class SNode {

private:

Object _elem;
SNode <Object> * _next;
};
};
```
Implementation - List

What private data do we need?

in private section:

struct Node <Object> * _head
Code

(switch to SlunckedList.h and SlunckedList.cpp)
Doubly Linked List
Implementation

What changes from Slunted list?