Announcements

- Program 4 will be up (Vector.h isn’t updated)
  due next Thursday

- No class Monday

- Reading assignment: read 6.2 (section on lists)
Lists:

Motivation: Insert in vectors is slow!
(Running time?) $O(n)$

Idea: If I know where the element should go, inserting should be easy.
Doubly Linked Lists

Sentinal nodes

Empty list:  \[
\begin{array}{c}
\text{head} \\
\text{null}
\end{array}
\]

Insert (ORD) - where?

\[ O(1) \] time

Node* mynode = new Node;
Problem: Pointers!

What do we need in order to know where we should insert?

- Node position (pointer)
- Nodes are private

Solution: "Wrap" Nodes inside a class.

In Iterations

Write functions which allow restricted pointer operations.
Iterators

An iterator will give the user a "pointer", but with a heavily controlled structure. (So they can't touch nodes directly.)

⇒ no seg faults

Compromise: Functionality versus info. encapsulation
STL functions

Mix of iterators & list functions.
Usage:

```cpp
list<int> my_list;

list<int> :: for_each(litj;
```
Code:

2 internal classes

private:
Node struct
  data:
  Node* _next
  Node* _prev
  Object _data

Iterator class (public)

private:
  Node* _current
Other ways circularly linked lists

head

Sentinel
tail