CS180- Lists

Announcements

- HW due Monday
- No lab next week
- Review Thurs, test Friday
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

\[ \text{head} \quad \text{ORD} \quad \text{LAX} \quad \text{MSP} \quad \text{tail} \]

Insert (ORD) - where?

1 node allocation
4 pointer updates
Better: Circularly linked lists

Empty: Sentinel

Sentinel

Sentinel
Problem: Pointers!

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

- need pointer to a Node

Solution: Iterator class whose private data is simply one pointer. Hide all possibility of seg

fault!
Iterators

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

Compromise: Functionality versus info. encapsulation

Goal: protect data a user
STL functions
Usage:

```cpp
List<int> my_list;

List<int> :: iterator it;

it = my_list.begin();
it++;
// now at 2nd element in list
my_list. insert(it, 5);

for (it = my_list.begin(); it != my_list.end(); it++)
    cout << *it << " " << "\n";
```

```cpp
# operator*
```
Code:

2 internal classes
\[ A[\sum_{i=1}^{j} i] = \text{value}_j \]
\[ A[\sum_{i=1}^{j} i] = \text{value}_j \]