Announcements

- Math/CS club talk today
  - Talk by Boeing interns at 4:10 in RH lobby

- Boeing scholarship apps due soon

- Program due Sunday by 11:59 pm

- Lab tomorrow

- Graded exams back tomorrow
Vectors:

A note about our functions - all based on STL class.

(on website)
Last time:

private:

```cpp
int currentCapacity;
int numItems; 0

ItemType * data; // pointer to array
```

Coded - constructor
- operator[]
- destructor
Erase

- Takes an index and deletes that element

Ex: `myVec.erase(3);`

`myVec.erase(7) -> error`

num Items = 6

Data → [7, 3, 11, X15, 11]

0 1 2 3 4 5

num Items = 5
```c
void erase(int index) {
    if (index >= numItems)
        throw out_of_range("Index out of range");
    for (int i = index; i < numItems - 1; i++)
        data[i] = data[i + 1];
    numItems--;}
```
Insert: Examples

myVector.insert(2, 'c');

otherVector.insert(11, 'new');

anotherVector.insert(7, -25);

Alice, Bob, Dan, Edward, Franky

insert Carol at position 2
How to insert?

What if it is full?

- double capacity
- create new array + copy everything into it
- delete old array
Code for insert:

```c
void insert(int index, const ItemType &value) {
    // Error handling
    if (index > numItems) throw error;

    if (numItems == currentCapacity) {
        int oldCap = currentCapacity;
        currentCapacity = 2 * currentCapacity;
        ItemType *oldData = data;
        data = new ItemType[currentCapacity];
        for (int i = 0; i < oldCap; i++)
            data[i] = oldData[i];
    }
    delete oldData;
    data[index] = value;
    ++numItems;
}
```
I actually insert new guy
for (int i = numItems - 1; i > index; i--)
    data[i] = data[i - 1];
data[index] = value;

numItems++;
3
void push_back (const ItemType & value) {
    insert (numItems, value);
}
Vector<ItemType>& operator=(const Vector<ItemType>& other) {
    if (this != &other) {
        delete data;
        numItems = other.numItems;
        currentCapacity = other.currentCapacity;
        data = new ItemType[currentCapacity];
        for (int i = 0; i < numItems; i++)
            data[i] = other.data[i];
    }
    return *this;
}