Announcements

- HW due tonight (written part=hand in now)
- Next HW & lab are posted
Recap of arrays

Limits

- not very flexible
  - size is fixed at creation
  - 1 kind of data
  - inserting + moving can be difficult

Q: How would we insert an element in the middle of an array?

ex: insert (20) in sorted order

```
2 5 6 11 25 26 31
```

pro: if \[26 \geq 3\]
**Singly Linked Lists**

A collection of nodes that together form a linear ordering.

```
head
```

```
LAX → STL → MSP → 0
```

```
delete this
```

```
Con: get element 263 is slow
```

```
Memory
```

```
<table>
<thead>
<tr>
<th>Node</th>
<th>LAX 263</th>
</tr>
</thead>
<tbody>
<tr>
<td>head</td>
<td>263</td>
</tr>
<tr>
<td></td>
<td>MSP 1036</td>
</tr>
<tr>
<td></td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>105</td>
</tr>
</tbody>
</table>
```
Copy Constructor:

other.head → current

head

myprevies

newnode

∅
Code

See SLinkedList.h & SLinkedList.tcc
Algorithm Analysis

How do we compare two programs?

SPEED

time to run
Speed

How fast an algorithm runs can be very dependent on variables in the system.

Examples:
- Architecture
- Language
- Low level (assembly)
- Inputs vary
Primitive Operations

As a way to compare algorithms in a generic way, we instead count primitive operations.

In addition, we (generally) only analyze the worst possible running time.

Why?
Comparing

OK, so we have the worst case # of operations - usually a function of \( n \).

How to compare?