lecture10: List Implementations

Sean Massung

Largely based on slides by Cinda Heeren
CS 225 UIUC

25th June, 2013
Announcements

- No lab today (but you should know that by now!)
- mt1 tonight! Covers all the C++ we’ve learned
- mp3.1 extra credit due Friday (6/28)
Inserting in the middle of a doubly-linked list

```cpp
template <class T>
void List<T>::insertAfter(size_t num, const T & data)
{
    if(num >= size) // make sure we’re in range
        return;

    // iterate to the position
    // we want to insert into

    // create new node

    // fix pointers
}
```
Three choices for the List ADT

1. Array
2. Singly-linked list (next pointers only)
3. Doubly-linked lists (next and prev)
List running times: linked lists vs arrays

<table>
<thead>
<tr>
<th>Function</th>
<th>Array</th>
<th>SLL</th>
<th>DLL</th>
</tr>
</thead>
<tbody>
<tr>
<td>insertFront</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insertBack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeFront</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeBack</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insertAtGiven</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeAtGiven</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>insertAtArbitrary</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>removeAtArbitrary</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
insert at the front and back
remove at the front and back
insert and remove at given locations

A given location is a cell index or pointer to a Node.
An arbitrary location is the $k^{th}$ element in the collection. For arrays, this is an index; for linked lists, this is $k$ elements past the head.