CS 367 - Introduction to Data Structures
Tuesday, January 26, 2016

We assume that you are proficient at object-oriented programming in Java.

Websites
- **Lec 3 & 1:** http://pages.cs.wisc.edu/~cs367-1/
- **Lec 2:** http://pages.cs.wisc.edu/~cs367-2/

See syllabus page for online readings and lecture outlines (no textbook).

Waitlisted? Continue attending. Some seats might open by Thursday.

Homework h1 tentatively due 10 pm Friday, January 29th (if so, handin info covered Thurs)
Program p1 assigned tonight
Assignment questions?
Post it on Piazza. Invitations to join were sent yesterday to your UW email.
Consult with a TA in 1366 CS. See course page for scheduled hours.

Last Time

Implementing the Bag ADT
- casting when using Object
- using Java generics for generality

List ADT
- designing the ListADT
- coding the ListADT as a Java interface

Today

Lists
- using lists via the ListADT
- implementing the ListADT using an array (SimpleArrayList)

Java API Lists
Iterators Concept

Next Time

Read: finish *Lists*
Iterators
- iterators and the Java API
- using iterators
- options for implementing iterators
- making a class iterable

Recall the List ADT

Concept

A List is a general, position-oriented container that stores a contiguous collection of items where duplicates are allowed. It maintains relative ordering and uses zero-based indexing.

Operations

```
void add(E item);
void add(int pos, E item);
E get(int pos);
E remove(int pos);
boolean contains(E item);
int size();
boolean isEmpty();
```

Issues

Null item – detect then signal with `IllegalArgumentException`
Bad position – detect then signal with `IndexOutOfBoundsException`
Empty list – handle as a bad position

Use - `ListADT`

→ Assume **`myList`** is a ListADT. **What does the following code fragment do in general?**

```
for (int i = 0; i < myList.size(); i++) {
   myList.remove(i);
}
```

Use - `ListADT`

→ Assume `myList` is a ListADT. **Write a code fragment** to reverse the contents of `myList` without using any additional ListADTs or other data structures (e.g., array).

Implementation - `ListADT` using a Generic Array

```
public class SimpleArrayList<E> implements ListADT<E> {

    private E[] items;      //the items in the List
    private int numItems;   //the # of items in the List

    public SimpleArrayList() {









   }

  //*** required ListADT methods ***

    public void add(E item) { ... }

    public void add(int pos, E item) { ... }
    public E remove(int pos) { ... }
    public E get (int pos) { ... }

    public boolean contains (E item) { ... }

    public int size() { ... }
    public boolean isEmpty() { ... }

  //*** additional optional array list methods ***

}
```

Implementing `contains`

→ **Complete the method below** so that it returns true iff the given `item` is in the list.

```
public boolean contains(E item) {
```

Implementing `add` at end

→ What problem might occur with the following implementation:

```
public void add(E item) {

   items[numItems] = item;
   numItems++;

}
```

Java API Lists

Design - Iterators

What are they?

Concept

Operations