CS 367 - Introduction to Data Structures
Wednesday, June 17, 2015 - Lecture 3

Course website: http://pages.cs.wisc.edu/~cstapleton/367/
Piazza: https://piazza.com/wisc/summer2015/cs367/

Last Time
- Review SackADT
- Generics & the ListADT
- ListADT Concept & Design
- ListADT Use
- ListADT Implementation
- Complete HW0 (you should be done by now!)
- HW1 Assigned (due 11 PM on Monday, June 22 2015.)

Today
- ListADT with Arrays
- Iterators
- Primitives vs. References Review

Next Time
- Read: Exceptions
- HW1 due 11 PM on Monday, June 22 2015.
- P1 due 11PM on Monday, June 29 2015
- Error Handling
- Exceptions
- Week in Review
Implementation - ListADT using a Generic Array

Design - Iterators
What are they?

Concept

Operations
Providing iterators (implementer's hat)

Direct vs. indirect access

What are the benefits of indirect access?

What do we lose with indirect access?

Using iterators: Examples

Suppose L is an ArrayList<String> containing Strings. Write a code fragment that uses iterators to print each item in L.
Complete the method using iterators to determine if the list contains duplicates.

```java
public boolean hasDups(ArrayList<String> list)
{
}
```

Suppose `words` is a `SimpleArrayList` that implements the `Iterable` Interface. Write a code fragment that gets an iterator, named `itr`, from `words`.

What happens here?

```java
for (Iterator<String> it = alist.iterator(); it.hasNext(); ) {
    System.out.println(alist.get(i++));
}
```
Adding iterators to GSack

Primitives vs. References

Assignment

Primitives:

References:

Comparisons

Primitives:

References:
Parameter Passing

Primitives:

References:

Returning Values

Primitives:

References:

P1 Assigned