Partially Filled Arrays
ArrayLists
Do-It-Yourself ArrayLists

Lecture 26

Readings
Next Week: Ch 8.3-8.8 and into Ch 9.1-9.3
(Ch 9.3-9.8 and Ch 11.1-11.3 in old 2nd ed)

(Reminder: Readings are absolutely vital for learning this stuff!)

Labs and Tutorials
Next Week: Lab #8 – A catch-up lab...

Learning Goals
By the end of class today you will be able to...
- Read and write programs that use partially-filled arrays (in the typical, idiomatic manner).
- Read and write programs that use Java's ArrayList class.
- Implement your own ArrayList class.
- (Ulterior Learning Goal: Get increasingly comfortable with arrays...)

Last Time: Teen Talk Barbie™ Reloaded
- A few lectures back, we wrote a program that learns some phrases and prints them back at random (inspired by Mattel's Teen Talk Barbie™ doll).
  - The goal was to have a fun example to introduce arrays.
  - But the programming style wasn't very good.
  - Better to create a TalkingDoll class...

Last Time: TalkingDoll Class
- Constructor that specifies name and maximum vocabulary size.
- Getters/Setters for the name.
- A method to add phrases to the doll.
- A method to get a random phrase back.
Arrays have a fixed size, but often we want to hold a variable number of items.

A Very Common Solution:
- Make an array bigger than you need.
- Have an int variable to keep track of what you are actually using.

A Very Common Solution:
- Your code must keep count variable up-to-date!
  - Good to keep these private, so your methods can always do the right thing.
  - (This is an example of a class or object invariant.)

\[aCount 2\]
\[a 3\]
\[a.length 4\]

Java's ArrayList Class
- What if you don't know (or want to force) a fixed maximum size in advance?
- Java has an ArrayList class for this case:
  - import java.util.ArrayList
  - Declare:
    \[ArrayList<String> phrases = new ArrayList<String>();\]

Methods:
- \[int size()\]
- \[void add(<type> newValue)\]
- \[<type> get(int index)\]
- \[void set(int index, <type> newValue)\]
- \[<type> can be any object type. See Ch 7.2.\]

TalkingDoll with ArrayLists
- Let's redo the TalkingDoll class using ArrayLists…

Do-It-Yourself ArrayLists
- ArrayLists are nothing magical!
  - (OK, the generic <type> stuff is kind of magic. See Ch 17.1 if you're interested (optional).)
- It’s just a class. If we fix the type of the elements (e.g., have an ArrayList of String), you know enough to write your own version.
- But how do you allow arrays to grow?
Real-Life Analogy: Moving Homes
- A house (or condo, apartment, etc.) has a fixed size. What happens when your family grows and you need more space?
- Answer: You buy a bigger place, and then you pack up and move all your stuff to the new place, and get rid of your old home.

Making Your Own ArrayList
- An array has a fixed size. What happens when your list grows and you need more space?
- Answer: You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.
Making Your Own ArrayList

- **Answer:** You allocate a bigger array, and then you pack up and move all your stuff to the new array, and get rid of your old array.

```
<table>
<thead>
<tr>
<th>aCount</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>a.length</td>
<td>4</td>
</tr>
<tr>
<td>newA</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>newA.length</td>
<td>8</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>aCount</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>a.length</td>
<td>8</td>
</tr>
<tr>
<td>newA</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>newA.length</td>
<td>8</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>aCount</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>a.length</td>
<td>8</td>
</tr>
<tr>
<td>newA</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>newA.length</td>
<td>8</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>aCount</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>(garbage collection)</td>
</tr>
<tr>
<td>a.length</td>
<td>8</td>
</tr>
<tr>
<td>newA</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>newA.length</td>
<td>8</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>aCount</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>(garbage collection)</td>
</tr>
<tr>
<td>a.length</td>
<td>8</td>
</tr>
<tr>
<td>newA</td>
<td>3 1 4 1</td>
</tr>
<tr>
<td>newA.length</td>
<td>8</td>
</tr>
</tbody>
</table>
```
MyStringArrayList

- Let's create a MyStringArrayList class.
- Methods:
  - int size()
  - void add(String newString)
  - String get(int index)