ArrayList

CSE 114 INTRODUCTION TO OBJECT-ORIENTED PROGRAMMING

1

Announcements

Midterm 3:
- - Review tomorrow
- - Exam Thurs 23-Nov

Course Evals => Please fill this out! Feedback is important to me!
- => Start date is 21-Nov. End date is 14-Dec. Notice will follow in email

Topic: ArrayList

Reading:
- Java Doc page: https://docs.oracle.com/en/java/javase/14/docs/api/index.html
  - On ArrayList: https://docs.oracle.com/en/java/javase/14/docs/api/java.base/java/util/ArrayList.html
- Additional: https://www.tutorialspoint.com/java/java_arraylist_class.htm

2

2

ArrayLists: Motivation

"Smart" array!

Arrays are fast and useful

But not very convenient
- Need to know the size to create one and can't resize it

What if you don't know the size up front? For example:
- Data could be 'streaming' in
  - Twitter feed
  - Remote sensors
  - etc.

Would be nice to have an array-like 'thing' (collection?) that can
- Dynamically resize as needed
- Add an element in the middle and shifting happens automatically
- Fill the 'hole' automatically if an element is removed in the middle somewhere
- Maintain the elements in the order that they come in

3

3

Collections

A collection is an object that contains other objects as elements

Some collections maintain an ordering, some allow duplicates

Typical operations: add, remove, contains, clear, size

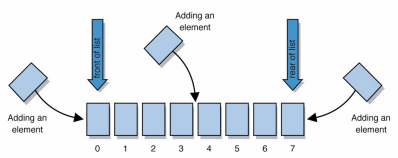
Examples found in Java class libraries (java.util.*):
- ArrayList, LinkedList, TreeSet, HashSet, TreeMap, HashMap, PriorityQueue
- We will study these in detail in CSE 214

4

4

Lists

List: a collection storing an ordered sequence of elements
- Each element is accessible by a 0-based index
- A list has a size (in addition to an unknown capacity)
- Elements can be added to the front, back, or elsewhere
- In Java, a list can be represented as an ArrayList object



5

5

Idea of a list ('listness' property)

Rather than creating an array, create an object that represents a "list" of items, initially an empty list, [ ]

You can add items to the list
- The default behavior is to add to the end of the list

[one, hi, two, there, sure, okay, apple]

The list object keeps track of the elements that have been added, their order, indices, the number of elements (size), etc.
- Think of an 'array list' as an automatically resizing array object
- Internally, the list is implemented using a regular array

ArrayList is one such example

6

6

ArrayList methods

add(value)	appends value at the end of the list
add(index, value)	inserts given value just before the given index, shifting subsequent values to the right
clear()	removes all elements of the list
indexOf(value)	returns first index where given value is found in list (-1 if not found)
get(index)	returns the value at given index
remove(index)	removes/returns value at given index, shifting subsequent values to the left
set(index, value)	replaces value at given index with given value
size()	returns the number of elements in list
toString()	returns a string representation of the list such as "[3, 42, -7, 15]"

- Focus on the orange methods for now

7

7

ArrayList methods (cont.)

addAll(list) addAll(index, list)	adds all elements from the given list to this list (at the end of the list, or inserts them at the given index)
contains(value)	returns true if given value is found somewhere in this list
containsAll(list)	returns true if this list contains every element from given list
equals(list)	returns true if given other list contains the same elements
iterator() listIterator()	returns an object used to examine the contents of the list (seen later)
lastIndexOf(value)	returns last index value is found in list (-1 if not found)
remove(value)	finds and removes the given value from this list
removeAll(list)	removes any elements found in the given list from this list
retainAll(list)	removes any elements not found in given list from this list
subList(from, to)	returns the sub-portion of the list between indexes from (inclusive) and to (exclusive)
toArray()	returns the elements in this list as an array

8

8

ArrayList vs. array

Construction

```
String[] names = new Strings[10];
ArrayList<String> ns = new ArrayList<String>();
```

Storing a value

```
names[0] = "Jennie";
ns.add("Jennie");
```

Retrieving a value

```
String s = names[0];
String s = ns.get(0);
```

9

9

ArrayList vs. array (cont.)

Doing something to each value that starts with "B"

```
for (int i = 0; i < names.length; i++) {
    if (names[i].startsWith("B")) {...}
}
for (int i=0; i < ns.size(); i++) {
    if (ns.get(i).startsWith("B")) {...}
}
```

Seeing whether the value "Billy" is found

```
for (int i = 0; i < names.length; i++) {
    if (names[i].equals("Billy")) {...}
}
if (ns.contains("Billy")) {...}
```

10

10

Looping through an ArrayList

Use a for-each loop

```
for (Student s : students) {
    // so something with s
}
```

Use an iterator object (don't worry about this one for now)

```
// students is an ArrayList<Student>
Iterator<Student> itr = students.iterator();
while (itr.hasNext()) {
    Student s = itr.next();
    // do something with s
}
```

11

11

Example

See UseArrayList.java

12

12