CSC207.01 2014S, Class 06: Arrays in Java

Overview

- Preliminaries.
    - Admin.
    - Questions on the homework.
    - Questions on the reading.
- Writing Classes.
- ADT Design.
- Arrays as ADTs.
- Lab.

Preliminaries

Admin

- Note that if you use the "Source" version of an EBoard, you should see the stuff within a minute or two of me updating it.
- Christine and Hannah want to talk to you about the TC corps.
- The CS dept. is asking you to wait a bit before voting in the SEPC election (unless the deadline is right upon us).
- Sorry for not mentioning last night’s mentor session. We’ll have them every Monday night.
- When sending me questions, please put "HELP" or "QUESTION" early in the subject, such as "CSC 207: HELP on Assignment 2".
- Today’s writeup: problems 1b and 6.
    - Due Friday
    - Subject: CSC 207 Writeup 4: Arrays (YOUR NAME(S))
- Extra credit:
    - CS Extras, Thursday: Ushahidi, Android, and 207 by Spender, Daniel, and Lea.
    - CS Table Friday: The ACM Code of Ethics.
    - Convo Feb. 5. (I’ll give my "Why go to convo" lecture closer to the date.)

Questions on the homework

How would you test `reverseInts`?

Here’s one simple test.

```
int[] original = new int[] { 1, 2, 3, 4, 5 };
int[] expected = new int[] { 5, 4, 3, 2, 1 };

reverseInts (original);
assertArrayEquals ("onetwothreefourfive", expected, original);
```

But I'd probably use loops to build and fill the arrays so that I can get different size arrays.

Questions on the reading

How do I build a new array?

```
assertArrayEquals ("onetwothreefourfive", new int[] { 5, 4, 3, 2, 1} , original);
```

Detour: Writing Classes

- We'll take a quick look at how we might write the classes from yesterday's discussion.
- See examples/time/Time.java and Examples/time/TimeZone.java

ADT Design

Three basic questions for every ADT

- What's the overall philosophy?
  - Homogenous collections of values where we can access values by integer index
- How might we use these things?
  - I am a number, not a person
- What methods should we provide to the client?
  - Simple/basic/essential
    - Get an element of the array - useTheIndexToFindTheElement (int index) (maybe just getElement (int index) or get (int index))
    - Change an element of the array - set (int index, TYPE newvalue)
    - Get the length of the array
    - CREATE A NEW ARRAY
  - More complex
    - Compare
    - Sort the array - sort() or sort (Comparator order)
    - Join two arrays together - concat (ArrayOfSameType addme) Does not affect original arrays; instead, creates a new one
    - Split the array into two arrays
  - Taken from other data types
    - push and pop
    - prepend and append
    - ...
  - Our design goal: SMALL AND CONCISE, not BIG AND EXPANSIVE
    - Quick

        - More likely to be correct
        - Can focus on the efficiency of your set of operations
        - If you have the right set, you can write the big and expansive procedures
    - Why have a BIG AND EXPANSIVE set of methods?
        - If lots of your clients are going to write the same methods, you save overall time (and get more clients) by writing the common methods
        - Personal satisfaction of knowing that lots of people use your code
        - More likely to be correct

N basic questions for every data structure

- What's the basic approach?
    - Arrays are traditionally one big chunk of memory. How big?
        - length of the array x size of individual element
        - plus another chunk of memory for the size
- What fields are we likely to need?
- How do we implement the methods?
    - To get the ith element: start of the array + size*index
- How efficient is this approach?

Detailed questions that come later

- What is private and public?

Lab

- Yes, I need to find a better balance of recitation and lab. But it was important for you to go through those questions.