Reading

- This week:
  - Chap 7: 7.1, 7.5-7.7. Topics 7.3 and 7.4 (3rd ed)
  - Chap 8: 8.1, 8.5-8.7. Topics 6.3 and 6.4 (2nd ed)

Arrays

Lecture 21, Mon Mar 8 2010

borrowing from slides by Kurt Eiselt

http://www.cs.ubc.ca/~tmm/courses/111-10

DO NOT put size of array in brackets on the left

DO NOT

Cans of pop sold

Answer: Arrays

- use arrays: common programming language construct
  - grouping related data items together
  - meaningful organization such that each individual data item can be easily retrieved or updated

- use array name followed by pair of square brackets
- inside brackets, place index of array element we want to access
- Reference to array element allowed anywhere that variables can be used
- Example:
  ```java
  System.out.println(cansSold[4]);
  ```

- Prints value 209

- Each value stored at unique numbered position
- number called index of array element
- aka subscript
- cansSold name of this array
- holds 10 values

- Collection of variables has single name
- how do we access individual values?

- To access individual value in array
- set array name to number we want to access
- Example:
  ```java
  int[] cansSold = new int[10];
  ```

- Looks like variable declaration, except:
  - empty brackets on the left tell Java that cansSold is an array...
  - number in the brackets on the right tell Java that array should have room for 10 elements when it's created
  - DO NOT put size of array in brackets on the left

Array Declaration and Types

- Just like ordinary variable, must
- declare before we use it
- give a type
- Since cansSold contains integers
  - make integer array:
    ```java
    int[] cansSold = new int[10];
    ```
  - Looks like variable declaration, except:
    - empty brackets on the left tell Java that cansSold is an array...
    - the number in the brackets on the right tell Java that array should have room for 10 elements when it's created
    - DO NOT put size of array in brackets on the left

Departmental announcements
Array Declaration and Types

```java
public class ArrayTest1{
  public static void main(String[] args) {
    int cansSold[] = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
    System.out.println("Element 4 is "+ cansSold[4]);
  }
}
```

Tracing Arrays and Loops

```java
public class ArrayTest3{
  public static void main(String[] args) {
    int totalCans = 0;
    int cansSold[] = {185, 92, 370, 485, 209, 128, 84, 151, 32, 563};
    for (int i = 0; i < cansSold.length; i++)
      totalCans = totalCans + cansSold[i];
    System.out.println("We've sold "+ totalCans + " cans of pop");
  }
}
```
public class ArrayTest3{
    public static void main(String[] args) {
        int totalCans = 0;
        for (int i = 0; i < cansSold.length; i++) {
            totalCans += cansSold[i];
            System.out.println("We've sold " + totalCans + " cans of pop");
        }
        System.out.println("And so on...");
    }
}

int totalCans = 0;
for (int i = 0; i < cansSold.length; i++) {
    totalCans += cansSold[i];
    System.out.println("We've sold " + totalCans + " cans of pop");
}

And so on...
Tracing Arrays and Loops

```java
public class ArrayTest3 {
  public static void main(String[] args) {
    int totalCans = 0;
    for (int i = 0; i < cansSold.length; i++) {
      int totalCans = totalCans + cansSold[i];
      System.out.println("We've sold " + totalCans + " cans of pop");
    }
  }
}
```

And so on...

cansSold0 185 1 92 2 370 3 485 4 209 5 128 6 84 7 151 8 32 9 56 3

totalCans2299

cansSold.length10

Something To Remember

- Array cansSold created with 10 elements
- Indices (place of index) are 0 through 9
- In general, array of size n will have indices ranging from 0 through n-1
- When you number things, you're used to beginning with 1
- Computer folks begin with 0
- Leads to "off by one" errors, even among computer veterans

Averaging Loop Example

```java
import java.util.Scanner;
import java.util.Arrays;

public class ArrayTest3 {
  public static void main(String[] args) {
    int totalCans = 0;
    int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 131, 32, 963};
    for (int i = 0; i < cansSold.length; i++) {
      int totalCans = totalCans + cansSold[i];
      System.out.println("We've sold " + totalCans + " cans of pop");
    }
  }
}
```

What would happen if we made this little change?

Java.lang.ArrayIndexOutOfBoundsException: 10

Histogram Loop Example

```java
numbers
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
```

PrintMax Loop Example

```java
numbers
0 1 2 3 4 5 6 7 8 9
```

- Now instead of average, we want to find and print maximum value from some arbitrarily large array
- Similar loop, but with some extra tweaks.
- Will require loop
- Simple task for looping in the context of an array
- How will we make this happen?

Storing Different Data Types

```java
public class ArrayTest3 {
  public static void main(String[] args) {
    int totalCans = 0;
    int[] cansSold = {185, 92, 370, 485, 209, 128, 84, 131, 32, 963};
    for (int i = 0; i < cansSold.length; i++) {
      int totalCans = totalCans + cansSold[i];
      System.out.println("We've sold " + totalCans + " cans of pop");
    }
  }
}
```

Could use two arrays of same size but with different types

- Write program to compare what's been collected from each machine vs. how much should have been collected?
Storing Different Data Types

Arrays With Non-Primitive Types

Arrays of Objects

Each individual String object in array
of course has all String methods
available

For example, what would this return?
location[2].length()