Java Programming: Static Methods, Arrays
Announcements

• HW06 Grace period until 11:59:59pm tonight

• HW07 is available on the web
  – Image processing in Java
  – Due next Thursday, March 15\textsuperscript{th} at 11:59:59pm

• Have a good break!
Java arrays
Java Arrays: Indexing

- An array is a sequentially ordered collection of values that can be indexed in constant time.
- Index elements from 0

![Diagram showing array indexing]

- Basic array expression forms
  
  \[
  \begin{align*}
  a[i] & \quad \text{access element of array } a \text{ at index } i \\
  a[i] = e & \quad \text{assign } e \text{ to element of array } a \text{ at index } i \\
  a.length & \quad \text{get the number of elements in } a
  \end{align*}
  \]
Java Arrays: Dynamic Creation

- Create an array `a` of size `n` with elements of type `C`
  ```java
  C[] a = new C[n];
  ```
- Arrays are objects that live in the heap, values with array type are mutable references
  ```java
  int[] a = new int[4];
  a[2] = 7;
  ```

Array entries are mutable

Length is a `final` (immutable) field
Java Arrays: Static Initialization

```java
int[] myArray = { 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000};

String[] yourArray = { "foo", "bar", "baz" };

Point[] herArray = { new Point(1,3),
                    new Point(5,4) };

herArray = new Point[] { new Point(2,3),
                         new Point(6,5) };```

CIS120 / Spring 2012
Java Arrays: Aliasing

- Variables of array type are references and can be aliases

```java
int[] a = new int[4];
int[] b = a;
a[2] = 7;
System.out.println(b[2]);
```

Stack

Heap

CIS120 / Spring 2012
Array Iteration
For loops

```java
static double sum(double[] a) {
    double total = 0;
    for (int i = 0; i < a.length; i++) {
        total += a[i];
    }
    return total;
}
```

General pattern for computing info about an array
Multi-Dimensional Arrays

A 2-d array is just an array of arrays...

```java
String[][] names = {
    {"Mr. ", "Mrs. ", "Ms. "},
    {"Smith", "Jones"}};

System.out.println(names[0][0] + names[1][0]);
// --> Mr. Smith
System.out.println(names[0][2] + names[1][1]);
// --> Ms. Jones
```

String[][] just means (String[])[]

names[1][1] just means (names[1])[1]

More brackets → more dimensions
int[][] products = new int[5][];
for(int col = 0; col < 5; col++) {
    products[col] = new int[col+1];
    for(int row = 0; row <= col; row++) {
        products[col][row] = col * row;
    }
}
```java
int[][] products = new int[5][];
for(int col = 0; col < 5; col++) {
    products[col] = new int[col+1];
    for(int row = 0; row <= col; row++) {
        products[col][row] = col * row;
    }
}
```

**Note:** This heap picture is simplified – it omits the class identifiers and length fields for all 6 of the arrays depicted.

(Contrast with the array shown earlier.)
Demo

ArrayExamples.java