Arrays (see 8.1-8.3)

Listen to the following (short, insightful) PLive lectures on loops:
1. The 3 lectures on Lesson page 7-6 and the whole page.
2. The 4 lectures in Lesson page 7-5.

A5 due tonight. Today, use the TA office hours in Upson 328B if you are really behind and need involved one-on-one help; for small questions see the consultants in the ACCEL lab green room.

Prelim 2. Tuesday, 10 November, 7:30PM
If you have a conflict, and if you haven’t been emailed about it, please email Maria Witlox mwitlox@cs.cornell.edu by Friday!!!

On (computational) simplicity
We are trying to teach not just Java, but how to think about problem solving.
Most of us don’t write perfect essays in one pass, and coding is the same: writing requires revising; programming requires revising.
If you are writing too much code — it gets longer and longer, with no end in sight: stop and look for a better way. If your code is getting convoluted and you have trouble understanding it: stop and look for a better way.
Learn to keep things simple, to solve problems in simple ways. This sometimes requires a different way of thinking.

A key point is to break a problem up into several pieces and do each piece in isolation, without thinking about the rest of them. Our methodology for developing a loop does just that.

Computer science has its field called computational complexity; mine is called computational simplicity. — Prof. Gries

Array: object that stores lists of things.
Holds a fixed number of values of a declared type. (So all will always hold 4 int values.)
The type of array a0 is int[]
Store its name in a variable (as always).
Elements of array are numbered: 0, 1, 2, ..., x.length – 1

Array initializers
Instead of
int[] c = new int[5];
Use an array initializer:
int[] c = new int[] {5, 4, 7, 6, 5};
array initializer: gives initial values for the array items. Values must have the same type, in this case, int. Length of the array is the number of values in the list; so...

Notes on array length
We write x.length, not x.length(), because length is a field, not a method.
Length field is final: an array’s length (field or actual number of items) cannot be changed once the array is created. We omit this field in the rest of the pictures.
The length is not part of the array type, which is int[].
This means that an array variable can be assigned arrays of different lengths; x could later hold the name of a seven-item int array. (But not the name of a seven-item double array).
Use of an array initializer

```java
public class D {

    /** = the month, given its number m.*/
    public static String theMonth(int m) {
        String month;
        if (m == 1) month = "January";
        else if (m == 2) month = "February";
        // ... more cases...
        else month = "Did not recognize month number.";
        return month;
    }
}
```

Differences between array and Vector ("classier", I'dow)

**Declaration:** int[] a; Vector v;

**Creation:**
- Elements of a: int values
- Elements of v: any Objects

**Array always:** n elements
**Number of elements can change**

**Reference:**
- a[e] vs. v.get(e)

**Change element:**
- a[e] = e1;
- v.set(e, e1);

**Array locations:** a[0], a[1], ..., in successive locations in memory.
Access takes same time no matter which one you reference.

**Elements all the same declared type** (a primitive type or class type)
**Initialization shorthand exists. Class has no methods, can’t be extended.**

Procedure for swapping variable values

```java
/* = Swap x and y */
public static void swap (int x, int y) {
    int temp;
    x = y;
    y = temp;
}
```

Linear search

```java
public class D {
    /** = index of first occurrence of c in b[h..]*/
    public static int findFirst (int c, int[] b, int h) {
        int i = h;
        // invariant: c is not in b[h..i-1]
        while (b[i] != c) {
            i = i + 1;
            // b[i] = c and c is not in b[h..i-1]
        }
        return i;
    }
}
```

Procedure swap for swapping array elements

```java
/** = Swap b[h] and b[k] */
public static void swap (int[] b, int h, int k) {
    int temp = b[h];
    b[h] = b[k];
    b[k] = temp;
}
```