Arrays

Some details ...

`int [ ] a` does not create the boxes;
`new array[6]` does

The variable actually holds a reference (pointer) to the array.
When we create the array with `new`, we have to know how many elements (boxes) to create.

Array variables are references

```java
int [ ] a = new int[4];
```

This is why we can change the items in a method. What we pass to the method is the reference.
Matrix
(rows and columns)

char [][] board = new char [3][3];

3 x 3 matrix:

Initialized arrays

int [] daysInMonth =
new int [] { 0, 31, 28, 31, 30, 31,
30, 31, 31, 30, 31, 30, 31 };

(Size comes from the initialization.)

Why did I make 13 months, and give the first zero days?

Methods can change arrays

static void swapItems( String [] stuff, int i, int k) {
    String tmp = stuff[i];
    stuff[i] = stuff[k];
    stuff[k] = tmp;
}

...  
String [] greeting = new String []
    { "Yabba", "Dabba", "Doo" };   
swapItems(greeting, 1, 2);

Example: Magic Square

2 7 6
9 5 1
4 3 8
Example: Magic Square

![Magic Square Diagram]

How would you check?

```java
static boolean isMagic(int[][] square) {
    First: Pseudocode
    Then: Java

    Get sum of first row ...
    int magicSum = 0;
    for (int col=0; col < square.length; ++col) {
        magicSum += square[0][col];
    }

    Other rows the same sum?
    for (int row=1; row < square.length; ++row) {
        int rowSum = 0;
        for (int col=0; col < square.length; ++col) {
            rowSum += square[row][col];
        }
        if (rowSum != magicSum) { return false; }
    }

    return true;
}
```
Columns the same sum?

```java
for (int col=0; col < square.length; ++col) {
    int colSum = 0;
    for (int row=0; row < square.length; ++row) {
        colSum += square[row][col];
    }
    if (colSum != magicSum) { return false; }
}
```

Diagonals the same sum?

```java
int diagSum = 0;
for (int row=0; row < square.length; ++row) {
    int col = row; // left to right, downward
    diagSum += square[row][col];
}
if (diagSum != magicSum) { return false; }
```

Diagonal right to left

```java
diagSum = 0;
for (int row=0; row < square.length; ++row) {
    int col = 2 - row; // diagonal from top right
    diagSum += square[row][col];
}
if (diagSum != magicSum) { return false; }
```

Some applications of matrices ...

Weather prediction, ecosystem simulation
- Each element represents an area
- At each time step, state of element changes depending on its neighbors
Medical and research imaging
- 3 dimensional grid of “voxels” representing a region in the head or body