Arrays in Java

data in bulk
Array

- Homogeneous collection of elements
  - all same data type
  - can be simple type or object type
- Each element is accessible via its index (random access)
- Arrays are, loosely speaking, objects
  - require initialization with the new operator
  - possess an instance variable (length)
Array declaration & initialization

• The syntax for array declaration is:
  dataType [] name;

• Examples:
  int [] numList;
  String [] names;
  Object stuff [];  // variant syntax – still allowed

• We initialize an array with new, specifying the length;
syntax:
  name = new dataType[size];
  e.g. names = new String[100];

• Declaration and initialization are often combined:
  int [] numList = new int[1000];
Quick check: use the space below to write code that declares & initializes an array of 100 ints to random values
Populating an array

• Simple type arrays are often populated using a simple count-controlled loop:

```java
Random rg = new Random();
for (int x = 0; x < numList.length; x++)
    numList[x] = rg.nextInt(5000);
```

• Relatively small arrays can also be initialized at declaration:

```java
String [] colors = {"red", "green", "blue"};
```
Iterating over an array

• Recent versions of Java have incorporated a new style of for loop, specifically for stepping through arrays

• Syntax:
  
  ```java
  for (data type item: arrayname)
  {
    // use item here instead of arrayname[x]
  }
  ```
Quick Check – re-write loop on the left using new style

double [] values = new double [100]
for (int x=0; x<values.length; x++)
values[x] = x * .5;
Arrays of objects

• The new operator that creates an array of objects creates an empty array
• The new operator and a constructor are required to populate the array with objects
• The next slide contains excerpts from a program containing several arrays of objects
• The program displays randomly-selected images in a slide show
private Icon [] imageArray;  // array of pictures for slide show
private int index = 0;  // index of next image – random #
private Container win;  // content pane of frame for image display
private JLabel pic;  // image gets embedded here for display
private Timer t;  // object that controls slide change
private Random rg;  // random # generator
private String prefix;  // holds name of image directory
private String [] fileNames;  // holds names of image files
private File picDir;  // used to obtain list of file names
private int numPix;  // used to size imageArray
prefix = new String
("C:\Documents and Settings\cshelle\My Documents\My Pictures");

picDir = new File(prefix);
// sets up directory object – refers to specified folder

fileNames = picDir.list();
// returns array of Strings – names of files in the My Pictures folder

numPix = fileNames.length;
// number of image files in the folder

imageArray = new ImageIcon[numPix];
// array of images to display – will be all images in folder
for(int x=0; x<imageArray.length; x++)
    imageArray[x] = new ImageIcon(prefix+"\"+fileNames[x]);
// populate imageArray with pictures from folder

rg = new Random();
// initialize new random # generator

pic = new JLabel(imageArray[rg.nextInt(numPix)]);
// grab first random image, embed in JLabel object

win.add(pic);
// put the picture in the window

t = new Timer(3000, this);
t.start();                // initialize & start Timer object
Arrays & methods

• An array can be either a parameter to or return value from a method
  – array parameter:
    void fillArray (int [] list)
  – array return value:
    int [] createList (int size)
  – examples of calls to these methods:
    int [] example = createList(100);
    fillArray(example);
Arrays & methods

• An important key point to remember when working with array arguments: you need to pass the array reference (the name of your array variable) when a parameter calls for an array argument

• No other notation is necessary, and would likely result in a syntax or logic error
Multi-dimensional arrays

• The arrays described thus far have been of the one-dimensional variety

• A multidimensional array is an array of arrays; we describe a two-dimensional array as having rows and columns
  – Each row is an array of columns
  – There are 2 indexes; the first indicates the row position, the second the column position
Declaring & using a 2D array

• Declaration:
  dataType [][] name;

• Initialization:
  name = new dataType[# rows][# columns];
  – note that if “dataType” is an object type, you still need to call the constructor for each object, just as you did for the 1-dimensional version

• 2D array is typically processed using nested loops
Example

```java
import java.util.*;

public class NumTableDemo {
    private int[][] table;
    private Random rg;

    public NumTableDemo (int size) {
        rg = new Random();
        table = new int[size][size];
        for (int x=0; x < table.length; x++)
            for (int y=0; y < table[x].length; y++)
                table[x][y] = rg.nextInt(size * 2) + 1;
    }
}
```
Quick check – write a method that prints out the table, nicely formatted
Finding sums of rows

```java
public int[] sumRows() {
    int[] rowSums = new int[table.length];
    for (int x=0; x < table.length; x++)
    {
        int sum = 0;
        for (int y=0; y < table[x].length; y++)
            sum = sum + table[x][y];
        rowSums[x] = sum;
    }
    return rowSums;
}
```
Finding sums of columns

public int[] sumColumns() {
    int[] colSums = new int[table.length];
    for (int x=0; x<table.length; x++)
    {
        int sum = 0;
        for (int y=0; y<table.length; y++)
            sum = sum + table[y][x];
        colSums[x] = sum;
    }
    return colSums;
}
public static void main (String [] args) {
    NumTableDemo demo = new NumTableDemo(5);
demo.showTable();
    int [] rowTotals = demo.sumRows();
    System.out.println("Sum of rows:");
    for (int x=0; x<rowTotals.length; x++)
        System.out.println(rowTotals[x]);
    System.out.println ("Sums of columns:");
    int [] colTotals = demo.sumColumns();
    for (int x=0; x<colTotals.length; x++)
        System.out.printf("%3d", colTotals[x]);
}
} // end of class
Quick check: write a method that returns the average of values in rows or columns
Class exercise: Sudoku

• I have provided the beginning of a class for playing Sudoku
• Several methods need to be added:
  – a method to process user input
  – a method that checks for rule violations
  – a method or set of methods that allows users to set up new games
• Work in groups of 2 or 3