Nick Troccoli
CS 106A

Section #5
July 24, 2017

Section Handout #5: Arrays

Portions of this handout by Marty Stepp and Chris Piech

1. Array Simulation

```
Array                       Final Array Contents
{10, 8, 9, 5, 5}            {10, 9, 9, 6, 6}
{12, 11, 10, 10, 8, 7}      {12, 12, 11, 11, 9, 8}
```

2. Index Of

```
private int indexOf(int[] list, int target) {
    for (int i = 0; i < list.length; i++) {
        if (list[i] == target){
            return i;
        }
    }
    return -1;
}
```

3. Unique Numbers

```
private int numUnique(int[] list) {
    if (list.length == 0) {
        return 0;
    }

    int count = 1;
    for (int i = 1; i < list.length; i++) {
        if (list[i] != list[i - 1]) {
            count++;
        }
    }
    return count;
}
```

4. Banish

```
public void banish(int[] a1, int[] a2) {
    for (int i = 0; i < a1.length; i++) {
        boolean found = false;   // see whether a1[i] is contained in a2
        for (int j = 0; j < a2.length && !found; j++) {
            if (a1[i] == a2[j]) {
                found = true;
            }
        }
        if (found) {   // shift all elements of a1 left by 1
            for (int j = i + 1; j < a1.length; j++) {
                a1[j - 1] = a1[j];
            }
            a1[a1.length - 1] = 0;
            i--;   // so that we won't skip an index
        }
    }
}
```

5. Collapse

```
public int[] collapse(int[] a) {
    int[] result = new int[a.length / 2 + a.length % 2];
    for (int i = 0; i < result.length - a.length % 2; i++) {
        result[i] = a[2 * i] + a[2 * i + 1];
    }
    if (a.length % 2 == 1) {
        result[result.length - 1] = a[a.length - 1];
    }
    return result;
}
```

6. Find Median (on next page)

```
private static final int MAX_TEMP = 100;   // max possible temperature

// Given a list of an odd number of temperatures, returns median temperature.
private int findMedian(int[] temps) {
   double halfTheEntries = temps.length / 2.0;
   int[] histogram = histogramFor(temps);
   int cumulativeTotal = 0;
   for (int i = 0; i <= MAX_TEMP; i++) {
      cumulativeTotal += histogram[i];
      if (cumulativeTotal >= halfTheEntries)
         return i;
   }
   return 0;  // can't get here, but Java requires us to return a value
}

// Given a list of temperatures, returns a histogram of those temperatures.
// Histogram is an array whose ith element is the number of temps of exactly i.
private int[] histogramFor(int[] temps) {
   int[] result = new int[MAX_TEMP + 1];
   for (int temp: temps) {
       result[temp]++;
   }
   return result;
}
```

7. The Sieve of Eratosthenes

```
import acm.program.*;

/* Computes prime numbers using the Sieve of Eratosthenes */

public class SieveOfEratosthenes extends ConsoleProgram {
   // The value up to which we should find prime numbers.
   private static final int UPPER_LIMIT = 1000;
   public void run() {
     // Create an array of booleans that track if we have crossed off
     // each number. Initially, each number has not been crossed off, so all
     // the booleans to all be false. Since this is what Java does anyway, we
     // don't need to explicitly set the boolean values to false.

     // crossedOff[i] represents the number i + 2
     boolean[] crossedOff = new boolean[UPPER_LIMIT  - 1];

     for (int n = 0; n < crossedOff.length; n++) {
        if (!crossedOff[n]) {
            println(n + 2);
            // Cross off all the multiples of n.
            for (int k = n; k < crossedOff.length; k += n + 2) {
                crossedOff[k] = true;
            }
        }
      }
   }
}
```

Images and Pixels (2D Arrays)

8. Flip Vertical

```
public void flipVertical(GImage image) {
    int[][] pixels = image.getPixelArray();
    int width = pixels[0].length;
    int height = pixels.length;
    for (int col = 0; col < width; col++) {
        for (int p1 = 0; p1 < height / 2; p1++) {
            int p2 = height - p1 - 1;
            int temp = pixels[p1][col];
            pixels[p1][col] = pixels[p2][col];
            pixels[p2][col] = temp;
        }
    }
    image.setPixelArray(pixels);
}
```

9. Stretch

```
public void stretch(GImage image, int factor) {
    int[][] pixels = image.getPixelArray();
    int[][] result = new int[pixels.length][pixels[0].length * factor];
    for (int row = 0; row < result.length; row++) {
        for (int col = 0; col < result[0].length; col++) {
            result[row][col] = pixels[row][col / factor];
        }
    }
    image.setPixelArray(result);
}
```

10. 2-D Array Simulation

```
4 5 6 6
5 6 7 7
6 7 8 8
```