Chris Piech
CS 106A

Section #6
February 23, 2018

Solutions to Section #6

1. How Prime!

```
/* File: SieveOfEratosthenes.java
 * ------------------------
 * This program prints out prime numbers in the range
 * up to and including UPPER_LIMIT.
 */
import acm.program.*;

public class SieveOfEratosthenes extends ConsoleProgram {
    private static final int UPPER_LIMIT = 1000;

    public void run() {
        // resolved[i] represents the number i + 2;
        boolean[] resolved = new boolean[UPPER_LIMIT - 1];
        for (int i = 0; i < resolved.length; i++) {
            resolved[i] = false;
        }
        for (int n = 0; n < resolved.length; n++) {
            if (!resolved[n]) {
                int number = n + 2;
                println(number);
                // Cross off all the multiples of n
                for (int k = n; k < resolved.length; k += number) {
                    resolved[k] = true;
                }
            }
        }
    }
}
```

2. Array Trace

```
Array 1: [10, 9, 9, 6, 6]
Array 2: [12, 12, 11, 11, 9, 8]
```

3. Switch Pairs

```
private String[] switchPairs(String[] arr) {
    String[] newArr = new String[arr.length];
    for (int i = 0; i < newArr.length - 1; i += 2) {
        newArr[i+1] = arr[i];
        newArr[i] = arr[i+1];
    }

    // For an odd number of elements, the last one is unchanged
    if (newArr.length % 2 == 1) {
        newArr[newArr.length - 1] = arr[arr.length - 1];
    }

    return newArr;
}
```

4. Flip Vertical

```
private GImage 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;
        }
    }
    return new GImage(pixels);
}
```

5. Stretch

```
private GImage 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];
        }
    }
    return new GImage(result);
}
```

6. Trace

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

7. Name Counts

```
/* File: CountNames.java
 * --------------------
 * This program shows an example of using a HashMap.  It reads a
 * list of names from the user and list out how many times each name
 * appeared in the list.
 */
import acm.program.*;
import java.util.*;

public class CountNames extends ConsoleProgram {
    public void run() {
        HashMap<String,Integer> nameMap =
            new HashMap<String,Integer>();
        readNames(nameMap);
        printMap(nameMap);
    }

    /*
     * Reads a list of names from the user, storing names and how many
     * times each appeared in the map that is passed in as a parameter.
     */
    private void readNames(HashMap<String,Integer> map) {
```

```
        while (true) {
            String name = readLine("Enter name: ");
            if (name.equals("")) {
                break;
            }

            /* See if that name previously appeared in the map.  Update
             * count if it did, or create a new count if it didn't.
             */
            if (map.containsKey(name)) {
                // auto-unboxing: gets an int instead of Integer
                int oldCount = map.get(name);
                // auto-boxing: convert int to Integer automatically
                map.put(name, oldCount + 1);
            } else {
                // auto-boxing: convert int to Integer automatically
                map.put(name, 1);
            }
        }
    }

    /*
     * Prints out list of entries (and associated counts) from the map
     * that is passed in as a parameter.
     */
    private void printMap(HashMap<String,Integer> map) {
        Iterator<String> it = map.keySet().iterator();
        while (it.hasNext()) {
            String key = it.next();
            int count = map.get(key);  // auto-unboxing
            println("Entry [" + key + "] has count " + count);
        }
    }
}
```

8. Mutual Friends

```
private HashMap<String, Integer> mutualFriends(
        HashMap<String, Integer> phonebook1,
        HashMap<String, Integer> phonebook2) {

    HashMap<String, Integer> result =
            new HashMap<String, Integer>();

    for (String name : phonebook1.keySet()) {
        int phoneNum = phonebook1.get(name);
        if (phonebook2.containsKey(name) &&
                phoneNum == phonebook2.get(name)) {

            result.put(name, phoneNum);
        }
    }
    return result;
}
```