

## Solutions to Section #6

---

### 1. 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;
}
```

### 2. How Prime

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

### 3. Image processing

```

private GImage flipHorizontal(GImage image) {
    int[][] array = image.getPixelArray();
    int width = array[0].length;
    int height = array.length;
    for (int row = 0; row < height; row++) {
        for (int p1 = 0; p1 < width / 2; p1++) {
            int p2 = width - p1 - 1;
            int temp = array[row][p1];
            array[row][p1] = array[row][p2];
            array[row][p2] = temp;
        }
    }
    return new GImage(array);
}

```

### 4. 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(Map<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.
            Integer count = map.get(name);
            if (count == null) {
                // auto boxing -- creates a new Integer with value 1
                count = 1;
            } else {
                // auto unboxing to get old value of count, and
                // then auto boxing to create a new Integer for count
                // with the new value that is 1 greater than old value.
                count++;
            }
            map.put(name, count);
        }
    }
}

```

```

/*
 * Prints out list of entries (and associated counts) from the map
 * that is passed in as a parameter.
 */
private void printMap(Map<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);
    }
}
}

```

## 5. 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;
}

```