Chris Piech Section #5
CS 106A February 14, 2018

Solution to Section #5

Portions of this handout by Eric Roberts and Marty Stepp

1. Word Count

```
/*
 * File: WordCount.java
 * --------------------
 * Counts the characters, words, and lines in a file.
 */

import acm.program.*;
import java.io.*;
import java.util.*;

public class WordCount extends ConsoleProgram {

   public void run() {
      int lines = 0;
      int words = 0;
      int chars = 0;
      Scanner fileScanner = openScanner("File: ");
      while (fileScanner.hasNextLine()) {
         String line = fileScanner.nextLine();
         lines++;
         words += countWords(line);
         chars += line.length();
      }
      fileScanner.close();

      println("Lines = " + lines);
      println("Words = " + words);
      println("Chars = " + chars);
   }

/**
 * Asks the user for the name of an input file and returns a
 * Scanner attached to its contents.  If the file does
 * not exist, the user is reprompted until they enter a valid filename.
 */
   private Scanner openScanner(String prompt) {
      Scanner fileScanner = null;
      while (fileScanner == null) {
         String name = readLine(prompt);
         try {
            fileScanner = new Scanner(new File(name));
         } catch (IOException ex) {
            println("Can't open that file.");
         }
      }
      return fileScanner;
   }
/**
 * Counts the words (consecutive strings of letters and/or digits)
 * in the input line.
 */
   private int countWords(String line) {
      boolean inWord = false;
      int words = 0;
      for (int i = 0; i < line.length(); i++) {
         char ch = line.charAt(i);
         if (Character.isLetterOrDigit(ch)) {
            inWord = true;
         } else {
            if (inWord) {
               words++;
            }
            inWord = false;
         }
      }
      if (inWord) {
         words++;
      }
      return words;
   }
}
```

2. How Unique!

```
/*
 * File: UniqueNames.java
 * ----------------------
 * This program continually asks the user for a name until the user
 * enters a blank line.  Then the program prints out the list of unique
 * names entered.
 */
import acm.program.*;
import java.util.*;

public class UniqueNames extends ConsoleProgram {

   public void run() {
      ArrayList<String> list = new ArrayList<String>();
      while (true) {
         String name = readLine("Enter name: ");
         if (name.equals("")) {
            break;
         }
         if (!list.contains(name)) {
            list.add(name);
         }
      }

      println("Unique name list contains:");
      printList(list);
   }

   /* Prints out contents of ArrayList, one element per line */
   private void printList(ArrayList<String> list) {
      for(int i = 0; i < list.size(); i++) {
         println(list.get(i));
      }
   }
}
```

3. Mirror

```
private void mirror(ArrayList<String> list) {
      for (int i = list.size() - 1; i >= 0; i--) {
            list.add(list.get(i));
      }
}
```

4. 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;
}
```

5. 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;
}
```

6. Collapse

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

7. Histograms

```
/*
 * File: Histogram.java
 * --------------------
 * This program reads a list of exam scores, with one score per line.
 * It then displays a histogram of those scores, divided into the
 * ranges 0-9, 10-19, 20-29, and so forth, up to the range containing
 * only the value 100.
 */

import acm.program.*;
import acm.util.*;
import java.io.*;
import java.util.*;

public class Histogram extends ConsoleProgram {

      public void run() {
            initHistogram();
            readScoresIntoHistogram();
            printHistogram();
      }

/* Initializes the histogram array */
      private void initHistogram() {
            histogramArray = new int[11];
            for (int i = 0; i < histogramArray.length; i++) {
                  histogramArray[i] = 0;
            }
      }

/* Reads the exam scores, updating the histogram */
      private void readScoresIntoHistogram() {
            try {
                  Scanner fileScanner =
                      new Scanner(new File(DATA_FILE));
                  while (fileScanner.hasNextLine()) {
                        String line = fileScanner.nextLine();
                        int score = Integer.parseInt(line);
                        if (score < 0 || score > 100) {
                              fileScanner.close();
                              throw new ErrorException(
                                  "That score is out of range");
                        } else {
                              int range = score / 10;
                              histogramArray[range]++;
                        }
                  }
                  fileScanner.close();
            } catch (IOException ex) {
                  throw new ErrorException(ex);
            }
      }

/* Displays the histogram */
      private void printHistogram() {
            for (int range = 0; range <= 10; range++) {
                  String label;
                  if (range == 0) {
                        label = "00-09";
                  } else if (range == 10) {
                        label = "  100";
                  } else {
                        label = (10 * range) + "-" + (10 * range + 9);
                  }

                  String stars = createStars(histogramArray[range]);
                  println(label + ": " + stars);
            }
      }

/* Creates a string consisting of n stars */
      private String createStars(int n) {
            String stars = "";
            for (int i = 0; i < n; i++) {
                  stars += "*";
            }
            return stars;
      }

/* Private instance variables */
      private int[] histogramArray;

/* Name of the data file */
      private static final String DATA_FILE = "MidtermScores.txt";
}
```