

## Section Handout #5: Files, ArrayLists, and Classes

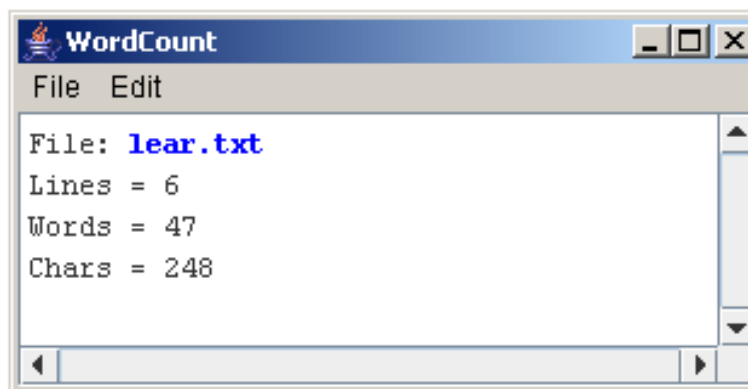
Portions of this handout by Eric Roberts

### 1. Word Count

Write a program `WordCount` that reads a file and reports how many lines, words, and characters appear in it. Suppose, for example, that the file `lear.txt` contains the following passage from Shakespeare's *King Lear*:

```
Poor naked wretches, wheresoe'er you are,  
That bide the pelting of this pitiless storm,  
How shall your houseless heads and unfed sides,  
Your loop'd and window'd raggedness, defend you  
From seasons such as these? O, I have ta'en  
Too little care of this!
```

Given this file, your program should be able to generate the following sample run:



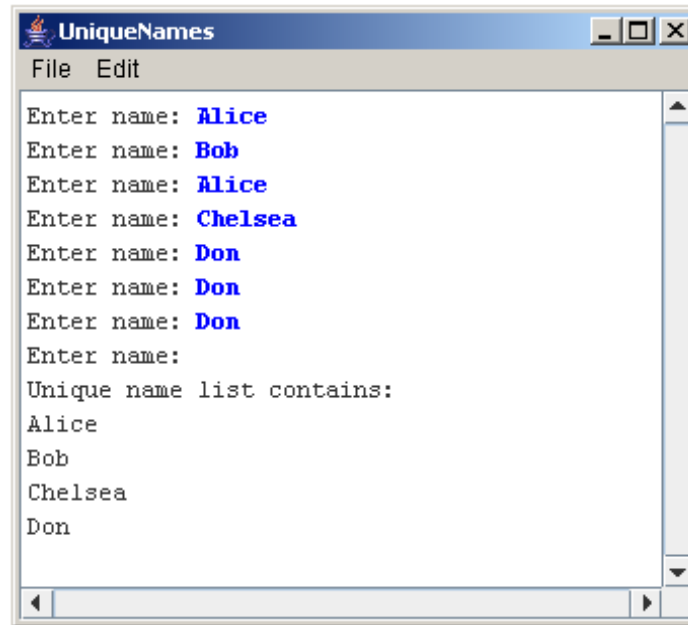
```
WordCount
File Edit
File: lear.txt
Lines = 6
Words = 47
Chars = 248
```

For the purposes of this program, a word consists of a consecutive sequence of letters and/or digits, which you can test using the static method `Character.isLetterOrDigit`. Also, you should not count the characters that mark the end of a line, which will have different values depending on the type of computer.

### 2. How Unique!

Write a program that asks the user for a list of names (one per line) until the user enters a blank line (i.e., just hits return when asked for a name). At that point the program should print out the list of names entered, where each name is listed only once (i.e., uniquely) no matter how many times the user entered the name in the program. You may find that using an `ArrayList` to keep track of the names entered by user may greatly simplify this problem.

A sample run of this program is shown below.



### 3. A Christmas Carol

Using the `student` class from Chapter 6 as an example, write a class definition for a class called `Employee`, which keeps track of the following information:

- The name of the employee
- The employee's nine-digit tax id number
- The employee's job title
- A flag indicating whether the employee is still active
- The employee's annual salary

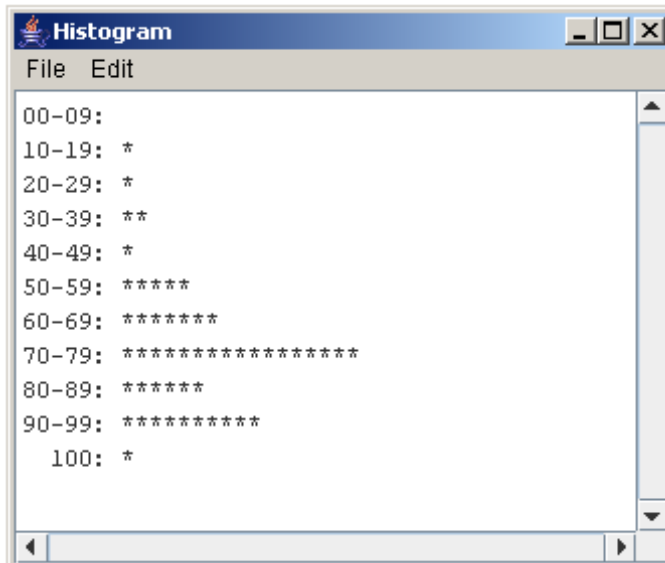
The first two fields should be set as part of the constructor, and it should not be possible for the client to change these values after that. For the other fields, your class definition should provide getters and setters that manipulate those fields. Once you have made this definition, write the code necessary to initialize the following `Employee` objects:

<code>ceo</code>	<code>partner</code>	<code>clerk</code>
<b>Ebenezer Scrooge</b>	<b>Jacob Marley</b>	<b>Bob Cratchit</b>
161803399	271828182	314159265
<b>CEO</b>	<b>Former Partner</b>	<b>Clerk</b>
<i>active</i>	<i>inactive</i>	<i>active</i>
<b>£1000</b>	<b>£0</b>	<b>£25</b>

What would you need to do to double Bob Cratchit's salary?

## 4. Histograms

Write a program that reads a list of exam scores from the file `MidtermScores.txt` (which contains one score per line) and then displays a histogram of those numbers, divided into the ranges 0–9, 10–19, 20–29, and so forth, up to the range containing only the value 100. If, for example, `MidtermScores.txt` contains the data shown in the right margin, your program should then be able to generate a histogram that looks as much as possible like the following sample run:



`MidtermScores.txt`

```
73
58
73
93
82
62
80
53
93
52
92
75
65
95
23
100
75
38
80
77
92
60
98
95
62
87
97
73
78
72
55
58
42
31
78
70
78
74
70
60
72
75
84
87
62
17
92
78
74
65
90
```

## 5. Happy Halloween

For the program below, trace through its execution by hand to show what output is produced when it runs.

```
/*
 * File: Halloween.java
 * -----
 * This program is just testing your understanding of parameter
 * passing.
 */
import acm.program.*;

public class Halloween extends ConsoleProgram {

    public void run() {
        int halloweenTown = 10;
        Skeleton bones = new Skeleton("bones");
        Pumpkin king = new Pumpkin(halloweenTown, bones);
        Skeleton skellington = bones;
        skellington.setName("skellington");
        halloweenTown = 5;
        println(king.toString());
    }
}

public class Pumpkin {

    private int x;
    private Skeleton y;

    public Pumpkin(int z, Skeleton w) {
        x = z;
        y = w;
    }

    public String toString() {
        return (y.getName() + " " + x);
    }
}

public class Skeleton {

    private String name;

    public Skeleton(String n) {
        name = n;
    }

    public String getName() {
        return name;
    }

    public void setName(String newName) {
        name = newName;
    }
}
```