CISC 181 Spring 2017
Practice Set 7
Assigned: April 17
Due: Apr 23 at 11:55PM on Sakai

Practice sets are to be completed individually. You are free to consult other students to help complete the practice sets (see syllabus for collaboration policy). However, keep in mind that each practice set is designed to cover basic material on which you will be quizzed and tested.

This practice set is intended to cover the following major topics:

- Using the List, Set, and Map interfaces for built-in data structures
- Using built-in methods in the java.util.Collections class
- Use Scanner with Files for simple input/output
- Use command line to compile, run, and provide arguments for a java program without using Eclipse

~~PART A~~ [15 POINTS] WORDCOUNT PARSE:

1. Create a PS7 Java Project in Eclipse. Download TestWordCount.java, hamlet.txt, romeoandjuliet.txt and copy them into your PS7/src folder.
2. In the default package, create a WordCount class with a Map<String, Integer> property for counts. Make a default constructor that creates a new HashMap for counts.
3. Add a parse method to your WordCount class that will take a Scanner (java.util) and a Pattern (java.util.regex) and keep track of the number of times a Pattern match appears in the stream.

   The general template for processing a Scanner stream this way is:

```
// while Scanner has more tokens
while (in.hasNext()) {
    // get the next token
    String token = in.next();
    // match the pattern within the token
    Matcher matcher = pattern.matcher(token);
    // process each match found in token (could be more than one)
    while (matcher.find()) {
        // get the String that matched the pattern
        String s = matcher.group().trim();
        // now do something with s
        System.out.println(s); // replace this line with your code
    }
}
```

4. Replace the System.out.println with code that will check your counts map for s as a key, if it doesn't exist put s in with a count of 1, otherwise put s in with current count + 1
5. Test your parse method with the given test in TestWordCount

~~PART B~~ [15 POINTS] WORDCOUNT REPORT:

1. Add a report method to your WordCount class that takes a PrintStream as a parameter and prints a sorted report of wordcounts to the stream.
    a. First, create a List<Map.Entry<String, Integer>> of results that contains all of the entries from counts (Hint: the entrySet() method of a Map returns a Collection of Map.Entry, and the ArrayList constructor can take a Collection as a parameter)
    b. Now sort the List using some helper methods from the Collections (java.util) class. These helpers create a Comparator that will sort our entries by value (which is the Integer count) in descending order:

```
Collections.sort(results, Collections.reverseOrder(Map.Entry.comparingByValue()));
```

    c. Print each of the results to a line in the stream
2. Test your report method with the given test in TestWordCount

PARTC [20 POINTS] WORDCOUNT COMPILE AND RUN FROM COMMAND LINE:

Eclipse is a nice IDE, but all Java programs are able to be compiled and run without Eclipse. In this part of the exercise we are going to compile and run and generate some output from your operating system "shell".

1. Add the following main method to your WordCount program:

```
public static void main(String[] args) throws IOException {
    // use the first argument as the file to read from
    Scanner in = new Scanner(new File(args[0]));
    // use the second argument as the pattern to match
    Pattern pattern = Pattern.compile(args[1]);
    // use the linefeed character to delimit each token in the stream
    in.useDelimiter("\n");
    // count and report the "words" that match the given pattern in the stream
    WordCount wc = new WordCount();
    wc.parse(in, pattern);
    wc.report(System.out);
}
```

2. The next step is slightly different for Windows/Mac:
    a. [Mac] Open Terminal (in Applications/Utilities). Change directory into your PS7/src folder. The easiest way to do this is to open Finder and browse to your PS7/src folder and check the full path location shown. Then in Terminal type cd /Full/Path/To/PS7/src, for example on my computer I typed:

```
cd /Users/jatlas/Dropbox/ud/teaching/181/jatlas/17S/workspace/PS7/src
```

    b. [Windows] Open Command Prompt (see https://www.lifewire.com/how-to-open-command-prompt-2618089 for how to do this on different Windows versions). Change directory into your PS7/src folder. The easiest way to do this is to open Windows Explorer and browse to your PS7/src folder and click inside the address bar to reveal the full path location. Then in Terminal type cd C:\Full\Path\To\PS7\src, for example on my computer I typed:

```
cd C:\Users\jamesatlas\Dropbox\ud\teaching\181\jatlas\17S\workspace\PS7\src
```

(also run the next two lines which tell the Windows prompt where to find the Java compiler on your system -- this only works if you installed the JDK correctly)

```
for /d %i in ("\Program Files\Java\jdk1.8*") do set JAVA_HOME=%i
set PATH=%PATH%;%JAVA_HOME%\bin
```

3. Compile your .java source code into a .class file:

   javac WordCount.java

   This shouldn't produce any errors or output, but you should now have WordCount.class in your src folder. WordCount.class contains machine code instructions for our program.
4. Run the machine code program with hamlet.txt and a pattern that matches all of the character prompts (so it gives us the counts of how many speaking chances each character has):

```
java WordCount hamlet.txt "[A-Z]{3,}[A-Z ]+"

HAMLET=409
HORATIO=128
KING CLAUDIUS=120
LORD POLONIUS=87
QUEEN GERTRUDE=83
LAERTES=73
ROSENCRANTZ=70
OPHELIA=68
GUILDENSTERN=55
MARCELLUS=41
OSRIC=29
POLONIUS=28
BERNARDO=26
REYNALDO=16
FRANCISCO=10
ACT IV=7
VOLTIMAND=7
PRINCE FORTINBRAS=7
CORNELIUS=6
ACT I=5
SCENE II=5
SCENE I=5
ACT III=4
FORTINBRAS=4
SCENE IV=3
SCENE III=3
LUCIANUS=3
CLAUDIUS=2
SCENE V=2
ACT II=2
ACT V=2
SCENE VI=1
GERTRUDE=1
DRAMATIS PERSONAE=1
```

```
SCENE=1
SCENE VII=1
GHOST=1
KING=1
```

5. Run the machine code program again with the same pattern and the romeoandjuliet.txt, but this time have your operating system direct the output to a file:

```
java WordCount romeoandjuliet.txt "[A-Z]{3,}[A-Z ]+" > rj_report.txt
```

6. In Eclipse, right-click your PS7 project and refresh. Verify that your rj_report.txt file contains the counts from Romeo and Juliet by opening it in Eclipse:

```
ROMEO=181
JULIET=135
BENVOLIO=75
MERCUTIO=70
FRIAR LAURENCE=65
CAPULET=63
LADY CAPULET=56
PARIS=33
ROMEO AND JULIET=27
TYBALT=23
SAMPSON=22
PRINCE=19
BALTHASAR=18
GREGORY=18
PETER=18
MONTAGUE=16
ACT II=7
ABRAHAM=7
LADY MONTAGUE=6
FRIAR JOHN=6
ACT III=5
SCENE III=5
ACT IV=5
ACT I=5
PAGE=5
SCENE II=5
SCENE I=5
SCENE V=4
SCENE IV=4
ACT V=3
LADY  CAPULET=2
PROLOGUE=2
SCENE VI=1
NURSE=1
DRAMATIS PERSONAE=1
SCENE=1
ESCALUS=1
LAURENCE=1
FRIAR=1
```

Submit your PS7 project to Sakai by exporting it from Eclipse as an archive.