Program #2 - Wherefore art thou, Word Cloud

“Let’s raise the cultural bar with some Shakespeare visualization”

Logistics:

- Part 1 due: **Mon Apr 27, 2015**
- Part 2 due: **Mon May 4, 2015**
- Worth: **10 points** (10% of your class grade)

Part 1 - Counting Shakespeare’s words

What are the top 20 most frequently-used words in Shakespeare’s Hamlet?
Yeah, I don’t know either. So, let’s do Program #2 and count them.

There’s lots of stuff in my `program2` folder on the k: drive.

- Some data - I have downloaded the text files for 5 of Shakespeare’s best: A Midsummer Night’s Dream, Hamlet, King Lear, Macbeth, and Romeo and Juliet. It’s all from a great site, The Gutenberg Project: www.gutenberg.org/ebooks/search/?query=shakespeare
- `WordTracker.java` - a word counting interface for you to implement
- `WordCount.java` - a simple class (a `String` and its count) used in `WordTracker`.
- `Program2Helper.java` - some Prof Bill code for you to use.

Create a simple console `main()` to test your warden. Mine looks like this:

```
Enter file name:
hamlet.txt

Select your action, Jackson:
      1. Set Top N
      2. Set min word size
      3. Add excluded word
      4. Show Top N words
      X. Exit
```

This should be a rough testing ground… functional, but nothing fancy.

Part 2 - Shakespearean Word Clouds

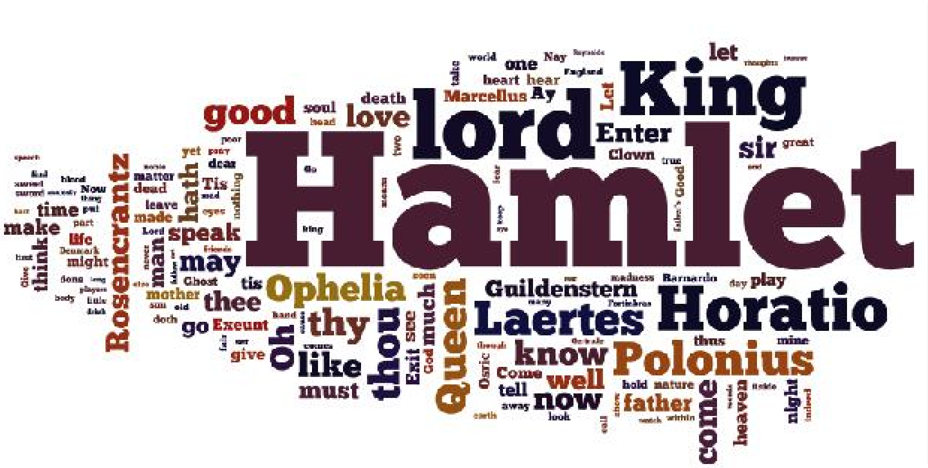
In part 2, we'll use our WordTracker to create a Word Cloud. Wikipedia defines it:

> *A tag cloud (word cloud, or weighted list in visual design) is a visual representation for text data, typically used to depict keyword metadata (tags) on websites, or to visualize free form text.*
> - en.wikipedia.org/wiki/Tag_cloud

There aren't any hard and fast rules to word clouds. So, I figure we'll just make our rules up in class. Here are some guidelines, I guess:

- Words used more often are BIGGER.
- A nice color palette is used.
- Words are clumped together, but they don't overlap.

For grins, here's an example for Hamlet… (No, it isn't mine. (ha!) I'm not that far yet… and, I don't even know if this was automagically generated or not.



Source: trippinglyonthetongue01.wordpress.com/hamlet-william-shakespeare/

So, let's talk Hamlet. After part 1, we'll have Tracker with all the word counts for Hamlet. So, we're good to go. In Part 2, we want to build a GUI that allows us to change parameters (Top N, min word size, colors, etc) to get the best-looking cloud results that we can. Then, we'd like to save that result to an image file. Done!

3. Design

Program #2 is more of an algorithmic challenge than Program #1. It's also open to a lot more of your creativity. It's more open-ended.

We'll talk in class and I'll summarize our talks here:

Program #2 Design Notes

Go!

4. Grading

Create a `program2` folder in your k: drive.
Place these files in that folder:

- A `README` file describing the state of your program
- All the Java files that comprise your Program #2 solution
- JPG or PNG image files of the favorite clouds you created and saved

All your code must follow our 161 Coding Guidelines. Ugly code will be penalized with a 0-100% reduction in points. A program that doesn't even compile is worth 0 points.

Good luck.
yow, bill

PS - My favorite is Shakespeare's "A Midsummer Night's Dream".
I played Oberon, King of the Fairies, in the 6th grade play… to rave reviews, of course!