Assignment 2.1: Assignment File I/O, Strings, and 2D Arrays

Due: 8am, Friday, November 6, 2008

Directions

The problems for this assignment are described below. You will submit this assignment by emailing it do Mr. Wulsin. The file will be named "Assignment 2.1-LastName.zip", where [LastName] is obviously your last name. All of your submitted assignments will be named in this fashion. The Zip file will contain a folder of the same name (so, "Assignment 2.1-LastName"), which will contain a folder for each of the subsequent problems. Each of these subfolders will have the name of its particular problem. Each of these problem subfolders will contain all of the necessary files for that particular problem. It's very important that you follow all of these naming conventions exactly as specified.

Problems

1) PalinTester (or PalTester if your politics prefer)

You will make a program that reads in a list of palindromes from a file and determines if each is a real palindrome. What is a palindrome, you say? It is a word or group of words that, minus the spaces, reads the same forward and reverse. For example,

racecar
tenet
Do geese see God
Was it Eliots toilet I saw
Never odd or even

are all palindromes. Your `PalinTester` class will only have one piece of private data, a String array that holds each of palindromes you read in from the text file (one per line). In your constructor your will pass in the path of the filename (I've provided you with a simple palindromes.txt file, but of course you can modify it to test other palindromes). The constructor will open and read that file by creating a `File` object and a `Scanner` object (see Chapter 14 in your book for more info). Remember that you need to create the `Scanner` object in a `try...catch` statement that tries to catch a `FileNotFoundException` and print out something like "Cannot file the file FileName," or something like that if the user gives the wrong filename. You should place your palindromes.txt file in the same folder as your bin and src folders if you want to pass in just the filename, like

```
sc = new Scanner(‘‘palindromes.txt’’);
```

The first thing in the palindromes file is a number, which indicates how many palindromes (or lines) are in the rest of the file. You need this number so you can create your String array of a proper size. You will then loop through the rest of the file, reading the next line (use `Scanner`'s `nextLine()` method), adding each line to the next slot in your String array.

Once you have successfully loaded your palindromes into your String array in your constructor, you will use two other methods. The public method `printResults()` simply loops throught each of your possible palindrome strings, prints it out, and then prints whether or not it's a true palindrome. The other, private method `isPalindrome(String palStr)` (what should its type be?) does the actual testing for whether the given String is a palindrome or not. You will call your `isPalindrome` method from your `printResults` method.

You will need to also make an external Tester class where your `public static void main` resides to actually create your `PalinTester` object and call its `printResults` method.

2) GraphingCalc
You will make a program that acts as a rudimentary graphing calculator, capable of graphing one or more linear equations (see GraphingCalcOutput.txt for an example). You will use a 2-dimensional `char` array (often called a matrix for obvious reasons) to store your characters at particular places in the graph. You will need to create axis lines in your graph using the `'|'` and `'-'` characters. Because of factors beyond our control, the graph won't be entirely square, but we'll get the idea. I have provided you a skeleton **GraphingCalc.java** file that you will need to fill in. One hint, when selecting the size of your graph, make sure you have the same number of rows and cols to the left/right and above/below your x- and y-axes lines (which themselves take up a row and column), respectively.

If you feel like you're stuck, go through your code line by line. Print out the values for each variable if you need to so you can see exactly what it happening.