CSE 142, Spring 2008
Programming Assignment #6: Baby Names (20 points)
Due: Tuesday, May 20, 2008, 4:00 PM

Special thanks to Stanford lecturer Nick Parlante for the concept of this assignment!
also see: http://www.peggyorenstein.com/articles/2003_baby_names.html

Problem Description and Program Behavior:

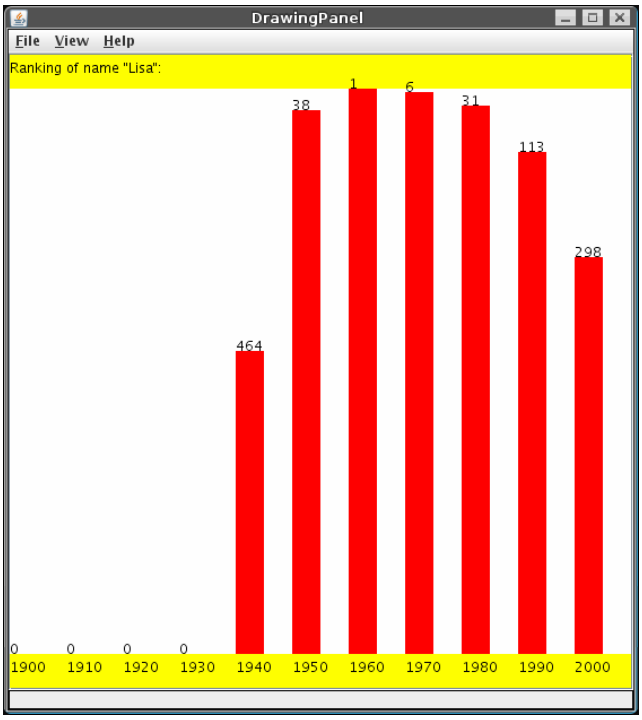
This assignment will give you practice with file processing. It generates graphical output, so you will need `DrawingPanel.java` in the same directory as your program. Turn in a file named `BabyNames.java`.

Every 10 years, the Social Security Administration gives data about the 1000 most popular boy and girl names for children born in the US. This data is provided on the web at http://www.ssa.gov/OACT/babynames/. Your task in this program is to prompt the user for a name, and then to display popularity statistics about that name for each decade since 1900. You will display both a text output of this data and a graphical bar chart of this data on a `DrawingPanel`.

```
This program graphs the popularity of a name
in Social Security baby name statistics
recorded since the year 1900.

Type a name: Lisa

Popularity ranking of name "Lisa"
1900: 0
1910: 0
1920: 0
1930: 0
1940: 464
1950: 38
1960: 1
1970: 6
1980: 31
1990: 113
2000: 298
```



Your program is to give an introduction and then prompt the user for a name to display. Then it will read through a data file searching for that name. If the name is found in the file, you should print the name and the statistics about that name's popularity in each decade.

Your program will read its data from a file named `names.txt`. Each line of this file has a name, followed by the popularity rank of that name in 1900, 1910, 1920, and so on. The default input file has 11 numbers per line, meaning that the last number represents the ranking in the year 2000. A rank of 1 was the most popular name that year, while a rank of 999 was not very popular. A rank of 0 means the name did not appear in the top 1000 that year at all. Here is a sample of the data:

```
Lionel 387 344 369 333 399 386 408 553 492 829 972
Lisa 0 0 0 0 464 38 1 6 31 113 298
Lise 0 0 0 0 0 997 0 0 0 0 0
Lisette 0 0 0 0 0 0 0 816 958 0 864
```

"Lionel" was #387 in 1900 and is slowly decreasing. "Lisa" made the list in 1940 and peaked in 1960 at #1.

If the name is found, you must also construct a `DrawingPanel` to graph the data. Your panel must exactly reproduce the window appearance of the examples for the same user input.

The panel's overall size is 550x560 pixels. Its background is white. It has yellow filled rectangles along its top and bottom, each being 30 pixels tall and spanning across the entire panel, leaving an area of 550x500 pixels in the middle.

If the name is not found in the file, you should simply output that it was not found, and not print or draw any data. No `DrawingPanel` should appear if the name is not found. The following is an example:

```
This program graphs the popularity of a name
in Social Security baby name statistics
recorded since the year 1900.

Type a name: zoidberg
"zoidberg" not found.
```

Graphical Output:

Each decade is represented by a width of 50 pixels. The bottom yellow rectangle contains black text labels for each decade, left-aligned and with the text's bottom at y=546. For example, the text "1900" has coordinates (0, 546) and the text "1910" has the coordinates (50, 546).

Starting at the same x-coordinate, a red bar shows the name's ranking over each decade. The bar is 25 pixels thick (half as wide as each decade). The table at right shows the mapping between rankings and y-values. The y-values start at 30, and there is a vertical scaling factor of 2 between pixels and rankings, so you should divide a ranking by 2 when calculating its y-coordinate. For example, a ranking of 38 in 1950 results in a 25x482 bar occupying the pixels from (250, 49) through (274, 530).

Rank	Top y
1	30
2, 3	31
4, 5	32
...	...
998, 999	529
0	530

At the same coordinate as the top-left corner of each red bar, black text shows the name's rank for that decade. For example, "38" appears at (250, 49), because "Lisa" had a rank of 38 in 1950. A rank of 0 means the name didn't appear in the top 1000. No red bar should appear for such a case, and the number 0 should be drawn at the bottom of the plot range at y=530. Lastly, the heading "Ranking of name …" is located at (0, 16).

Implementation Guidelines:

Your program should work correctly regardless of the capitalization the user uses to type the name. For example, if the user asks you to search for `"LISA"` or `"lisa"`, you should find it even though the input file has it as `"Lisa"`. The name that is displayed on the console and `DrawingPanel` should have capitalization matching the way it appears in the file.

To draw the text labels on the `DrawingPanel`, you will need to use the `drawString` method of the `Graphics` object. Some of the text labels you'll want to write will be `int`s, but you can convert them into `String`s using the `+` operator with an empty string. For example, if you have an `int` named `x` with value `1900`, the expression `("" + x)` yields the string `"1900"`. To draw this at (0, 546), you'd write:

```
g.drawString("" + x, 0, 546);
```

Stylistic Guidelines:

For this program you should have **four class constants** to represent the following values:

- the name of the input file (default of `"names.txt"`)
- the starting year of the input data (default of `1900`)
- the number of decades' worth of data in each line of the file (default of `11`)
- the width used for each decade on the drawing panel (default of `50`)

If the constants' values are changed, your output should adapt appropriately. For example, if you change the starting year to 1800, the program will now assume the file data comes from the years 1800, 1810, and so on. The panel's overall width should adjust if your width or decades constants are changed; for example, if you change the width to 70 and the decades to 5, the panel's size should become 350x560. On the course website is a second input file named `names2.txt` that has 8 decades worth of data you can use to test your file name constant.

Use methods for structure and to avoid redundancy. For full credit, your methods should obey the following constraints:

- The `main` method should not draw on a `DrawingPanel`, nor should it read lines of input from a file.
- The code that asks the user for a name must not be in the same method as any code to read lines of input from a file.
- Split the task of displaying the data into at least two methods. For example, you could have one method to do the text output and one to do the graphical output, or you could have one method to draw the "fixed" graphical content (yellow bars, decade labels) and another for the content that comes from the file (red bars, ranks).

For this assignment you are limited to the language features in Chapters 1 through 6 of the textbook. In particular, **you are not allowed to use arrays to solve this assignment.** For reference, our solution is 105 lines long and has four methods other than `main`. Follow past stylistic guidelines about indentation, whitespace, identifier names, and commenting.