The code for this assignment is available through Snarf as assignment/digit as well as on the course webpage.

Ambient: https://www.cs.duke.edu/csed/ambient/

Java Help

You should expect to take some time getting used to the Java language. Refer to Resources on Sakai such as the basic Java to Python and MATLAB syntax cheatsheet. Additionally the Java API docs can be of use when looking up how specific method or classes work.

Finally your classmates and TAs are available on Piazza to answer any questions you have. **DO NOT post your own code on Piazza publically. This will be removed and is in violation of course policy (sharing code with classmates is not allowed).** You can post your code in a private Piazza question though, although it is often more helpful to bring specific code question to helper or office hours.

Background

This nifty assignment was developed by Steve Wolfman, Pratt 1997, ECE/CS Dual Major. We've made a few minor edits to give it a bit of a local flavor.

Please read this assignment all the way through before you start programming.

INTRODUCTION:

The natural world is full of hidden and beautiful mathematics. The whorls of a conch shell hide the Fibonacci sequence and its Golden Ratio, plants grow in fractal patterns, and comets trace hyperbolic patterns through the solar system. All those beautiful patterns hide in the grungy data of human observation.

So, what are the populations of every town in North Carolina and the number of posts by various authors to a Duke sports bulletin board hiding from you?

Related reading

- Wolfram MathWorld's page on Benford's Law.
- Possible fraud from a polling agency detected by looking at the digit distribution

ASSIGNMENT OVERVIEW:

In this assignment, you will write a program that determines the distribution of initial digits in a set of data (see the data directory for examples). In the end, we want a program that reads in a number n and a list of numbers from a file and outputs a list of 10 values: the frequency with which each digit 0–9 appears as the n\text{th} digit of one of the input numbers.

To be human-readable, the data files should also allow labels for the data. We'll accomplish this by allowing commenting in the input file. Your program should ignore anything between (* and *). You may assume that the (* and *) symbols will be surround by whitespace and that nested comments — comments inside other comments — are not allowed. Below are enrollments in Research Triangle Park colleges and universities in Fall 2000 (thanks to the "Research Triangle Regional Partnership" website: http://www.researchtriangle.org/data/enrollment.html).
If the user enters 0 as the value for \( n \), and thus chooses the first digit, then your program should print:

- 0s: 0 (0%)
- 1s: 3 (27%)
- 2s: 4 (36%)
- 3s: 1 (9%)
- 4s: 0 (0%)
- 5s: 2 (18%)
- 6s: 1 (9%)
- 7s: 0 (0%)
- 8s: 0 (0%)
- 9s: 0 (0%)

In your README, you should describe the distribution of digits for each of the data files and how they conform or differ from what you would expect from a random list of numbers.

Note: throughout this problem, you may assume that the numbers processed are non-negative or you can use the absolute value function `Math.abs` to help you handle negative numbers in a reasonable way.

**Assignment Plan:**

We provide you with a small amount of starter code for this assignment. You should snarf the assignment digit for the data directory and `Prestidigitation.java`. Below a suggested decomposition of the problem into functions that you can implement. However, you can ignore the framework below and create whatever methods you like as long as you produce the same output in the end.

1. **Write a method `nthDigit`.**

   \( \text{nthDigit}(n, \text{num}) \) finds the \( n \)th highest order digit of \( \text{num} \), i.e., the \( n \)th digit from the left. We take the leftmost digit to be the 0th. `nthDigit` should evaluate to 0 for digits beyond the "end" of the number. For example:

   - \( \text{nthDigit}(0, 678) \Rightarrow 6 \)
   - \( \text{nthDigit}(1, 678) \Rightarrow 7 \)
   - \( \text{nthDigit}(2, 678) \Rightarrow 8 \)
   - \( \text{nthDigit}(3, 678) \Rightarrow 0 \)
   - \( \text{nthDigit}(0, 0) \Rightarrow 0 \)
   - \( \text{nthDigit}(3, 18023) \Rightarrow 2 \)
2. Write a method `nthDigitTally`, using `nthDigit`.

`nthDigitTally(n, nums)` returns a tally of frequencies of 0–9 as the `n`th digits of all the numbers in `nums`. Below is a sample test case using the enrollments above and repeated in the table below.

<table>
<thead>
<tr>
<th>Institution</th>
<th>Enrollment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duke University</td>
<td>12176</td>
</tr>
<tr>
<td>North Carolina Central University</td>
<td>5476</td>
</tr>
<tr>
<td>Louisburg College (Junior College)</td>
<td>543</td>
</tr>
<tr>
<td>Campbell University</td>
<td>3490</td>
</tr>
<tr>
<td>University of North Carolina at Chapel Hill</td>
<td>24892</td>
</tr>
<tr>
<td>North Carolina State University</td>
<td>28619</td>
</tr>
<tr>
<td>Meredith College</td>
<td>2595</td>
</tr>
<tr>
<td>Peace College</td>
<td>603</td>
</tr>
<tr>
<td>Shaw University</td>
<td>2527</td>
</tr>
<tr>
<td>St. Augustine's College</td>
<td>1465</td>
</tr>
<tr>
<td>Southeastern Baptist Theological Seminary</td>
<td>1858</td>
</tr>
</tbody>
</table>

Assume the variable `enrollments` is an `ArrayList` of integers containing the enrollment numbers from that table. Then:

`nthDigitTally(0, enrollments) ⇒ [0, 3, 4, 1, 0, 2, 1, 0, 0, 0]`

3. Write a method `readNumbers` that reads whitespace-separated integers from a `Scanner` and returns a list of the numbers suitable as input to `nthDigitTally`. Here's the university enrollment data from above:

```
12176 5476 543 3490 24892 28619 2595 603 2527 1465 1858
```

From this, `readNumbers` should produce the list `[12176, 5476, 543, 3490, 24892, 28619, 2595, 603, 2527, 1465, 1858].`

4. Finally, compose your `main` method to prompt the user for the number `n` and to choose a file for the data set. The program should tally the `n`th digits of the numbers in the data set and print out a table of the results. For example, given that `n=0` and the following file:

```
12176 5476 543 3490 24892 28619 2595 603 2527 1465 1858
```

Your program should print:

0s: 0 (0%)
1s: 3 (27%)
2s: 4 (36%)
3s: 1 (9%)
EXTRA CREDIT:

If you want to find the patterns hidden in the numbers around you, try the following three-part bonus problem:

i. Find a data source on the web *that no one else has used* (see next part) and transform it into a format suitable for this assignment. The data must all be separate measurements of a single type of phenomenon. For example: measurements of university/college enrollments across different institutions (like above) or at the same institution across different years; measurements of the flow rates of all the major rivers in North Carolina; measurements of the height of 10000 randomly chosen Durham residents; measurements of the number of hits per day on the Duke computer science website over three years; measurements of the length in characters of each article in the Wikipedia; measurements of the population of the 1000 largest cities and townships in the US; etc. Furthermore, **there must be at least 250 measurements in the list** (but more would be better!).

ii. Post all of the following items to the digit-data folder on Piazza with a title describing your data: the URL for your data source, a description of the data source, one attachment with bare data (i.e., no comments), and one attachment with labelled data (using the (* *) style above).

iii. Submit with your assignment the URL of your data, a description of the data source, and digit tallies for digit 1 and digit 2 of your data. Are there any oddities in the tallies? What about in other students' data?

SUBMITTING:

Submit Prestidigitation.java, README.txt, and any extra credit new data files under assignment name *digit*. Provide comments for the methods you write and include any tests that you wrote for the individual methods.

Last modified: Thu Sep 4 22:42:32 EDT 2014