~~Detecting Fraud~~
Looking Out For Number One

Andrew Rosen
~~stolen~~ ~~adapted~~ shamelessly coopted from Steve Wolfman's assignment.

Abstract

This is a group assignment. The first part serves as review, the second as list practice, the third as file-reading and application, and the fourth as science.

Hidden patterns

Our world is controlled by mathematics – just ask any physicist. Sometimes math can be beautiful, such as when the golden ratio crops up in nature.

This assignment asks what does the number of social media posts each day, the population of every town in the US, the address of every faculty member of Temple, and the size of tallest 100 buildings have in common?

Please utilize the discussion board on Canvas for this assignment. Some parts are tricky, and I encourage you to ask your classmates for help. The final part of this project requires you to post your dataset on the message board.

0.1 Assignment

The end goal of this program is to write a program that determines the distribution of initial digits in a set of data. In the end, we want a program that reads in a number n and a list of numbers `nums` and outputs an n^{th} of 10 values: the frequency with which each digit 0-9 appears as the n^{th} digit of one of the input numbers. However, we'll break that problem down into easier steps.

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

1 Review

1.1 Count Digits

We'll start out with something we've done in class before. Write a function `countDigits` which takes in an `int` *num*, which will return the number of digits in *num*. For example `countDigits` should evaluate to:

- 1 for numbers 0–9,
- 2 for numbers 10–99,
- 3 for numbers 100–999, etc

Remember, use repeated division by 10 to calculate the number of digits in `num`. (There's also a tricky solution using logarithms that avoids the repeated division!)

1.2 n^{th} Digit Back

Write a method `nthDigitBack` which takes in two `int`s, n and *num*. This method finds the n^{th} lowest order digit in `num`. In other words, `nthDigitBack` returns the n^{th} digit from the right. The rightmost digit is considered the 0^{th} digit back. `nthDigitBack` should evaluate to 0 for digits out of range (so if you ask for the 1000th digit back of 7546, `nthDigitBack` will return 0). Here are some example method calls of `nthDigitBack`:

- `nthDigitBack(0,123)` $\Rightarrow$ `3`
- `nthDigitBack(1,123)` $\Rightarrow$ `2`
- `nthDigitBack(2,123)` $\Rightarrow$ `1`
- `nthDigitBack(3,123)` $\Rightarrow$ `0`
- `nthDigitBack(0,0)` $\Rightarrow$ `0`
- `nthDigitBack(3,18023)` $\Rightarrow$ `8`

1.3 nthDigit

Write a function `nthDigit`, which takes in two `int`s, n and num, and returns the n^{th} *highest* order digit of `num`. In other words, this method returns the n^{th} digit from the *left*, as opposed to `nthDigitBack`, which returns the n^{th} digit from the right. The leftmost digit is considered the 0^{th} digit here. Just like `nethDigitBack`, `nthDigit` should evaluate to `0` for digits beyond the "end" of the number.

If `nthDigit` calls `nthDigitBack` and `countDigits`, you can do the entire method in one line. Think about how to convert the problem of the n^{th} digit from the left into finding the same digit, but going from the right.

Example method calls:

- `nthDigit(0,123)` $\Rightarrow$ `1`
- `nthDigit(1,123)` $\Rightarrow$ `2`
- `nthDigit(2,123)` $\Rightarrow$ `3`
- `nthDigit(3,123)` $\Rightarrow$ `0`
- `nthDigit(0,0)` $\Rightarrow$ `0`
- `nthDigit(3,18023)` $\Rightarrow$ `2`

2 list Practice

2.1 Update Tally

Write a method called `updateTally`, which calls `nthDigit`. `updateTally` takes in three arguments, an `int` n, an `int` num, and an `int[]` $tally$. We assume that $tally$ is a `int[]` of 10 integers. The $tally$ contains the tally of the number of n^{th} digits seen so far. It updates $tally$ to reflect the n^{th} digit of num.

In other words, if 5 is the n^{th} digit of num, we increment index 5 of $tally$. If 2 is the n^{th} digit of num, we increment index 2 of $tally$. If d is the n^{th} digit of num, we increment index d of $tally$.

Examples showing how $tally$ changes, where $tally$ has an initial value of `[0,0,1,2,0,0,3,0,9,0]` in each example:

- `updateTally(2, 1072, tally)`

  $\Rightarrow$ `tally` is now `[0,0,1,2,0,0,3,1,9,0]`

- `updateTally(0, 2541, tally)`

  $\Rightarrow$ `tally` is now `[0,0,2,2,0,0,3,0,9,0]`

Remember, since we are `modifying` tally, `updateTally` will be a `void` method which returns nothing.

2.2 n^{th} Digit Tally

Write a method called `nthDigitTally`, which calls `updateTally`. The method takes in two arguments: `int` n, which is the digit we are interested in, and an `int[]` $nums$, which is an list of numbers. `nthDigitTally` returns a tally of frequencies of 0–9 as the n^{th} digit of all the numbers in `nums`.

Here's a sample test case. These are enrollments in Research Triangle Park colleges and universities in Fall 2000. [1]

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

Assume the variable `enrollments` contains the enrollment numbers from that table. Then: `nthDigitTally(0, enrollments)` ⇒ `[0,3,4,1,0,2,1,0,0,0]`

This is because none of the enrollment numbers begin with 0, three begin with 1, four begin with 2, one begins with 3, and so on.

[1] source http://www.researchtriangle.org/data/enrollment.html)

3 Reading Data

You will need to utilize `Scanner` to

3.1 Read Data

Write a method `readMysteriousNumbers` that reads whitespace-separated integers from **a file** and returns an list of 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
```

If the above is entered stored in a file and given to `readMysteriousNumbers`, the method should return the list `[12176, 5476, 543, 3490, 24892, 28619, 2595, 603, 2527, 1465, 1858]`.

3.2 Main

Finally, write your main method to read a number `n` from input. You can let a user enter a file name of a dataset or you can automatically use one. 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:

```
0
enrollment.txt
```

where `enrollment.txt` is our example enrollment data, your program should print:

```
0s: 0
1s: 3
2s: 4
3s: 1
4s: 0
5s: 2
6s: 1
7s: 0
8s: 0
9s: 0
```

4 The Reveal and Submitting

So it works. Now what?

4.1 Make Observations

We will now use our program to explore a hidden pattern that occurs for large sets of numbers.

Take a look at what happens to the first digit of items in a large dataset.

1. Test you program on one of my posted datasets. Alternatively, for the full 100 points, find a data source on the web that no one else has used (see next part) and transform it into a format suitable for input to `readMysteriousNumbers`. 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 the United States; measurements of the height of 10000 randomly chosen Philadelphians; the karma value of the top posts on Reddit; measurements of the length in characters of each article in the Wikipedia; measurements of the population of the 1000 largest cities and townships in Japan; etc. Furthermore, **there must be at least 250 measurements in the list** (but more would be better!).

2. Go to `canvas.temple.edu` and find the "discussions" tab in the course sidebar. Find the discussion I created with the title "Mysterious Numbers Discussion." Post all of the following items to the discussion with: the URL for your data source, a description of the data source, and one attachment with data suitable for `readMysteriousNumbers`.

3. On the assignment dropbox, 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 (using `nthDigitTally`). Are there any oddities in the tallies? What about in other students' data?