CS1671 Homework #1

Due 11:59pm 1/25/2016.

Overview

The basic goal of this assignment is to write a program that takes a text file as input and convert all expressions involving numbers to their appropriate word forms (that is, something that someone would say -- for example: "January 15, 2016" would become "January fifteenth, two-thousand sixteen.") In other words, your program is something like a speech generation preprocessor: turning numbers and symbols into words.

Basic Requirement

Your program should be written in one of these three languages: Python, Java, or C++. Please see me if you cannot work with any of those languages. The input to your program will be a text file that contains one sentence per line. Your program should use regular expressions to identify the numerical expressions in each sentence and rewrite them with words, and then print the new sentence to standard out.

Your program should handle *at least* the obvious cases for these five types of numbers:

- **Basic numbers**: for example, "1,234.567" becomes "one thousand two hundred thirty four point five six seven."
- **Dates**: for example, "January 15 , 2015" becomes "January fifteenth , two thousand fifteen."
- **Dollar amounts**: for example, "$ 3.25" becomes "three dollars and twenty five cents."
- **Percentages**: for example, "0.35 %" becomes "zero point three five percent" [or "point three five percent" is OK too].
- **Fractions**: for example, "7\/8" becomes "seven eighth"; "3 1\/2" becomes "three and a half." (Note: in this corpus, forward slash has been escaped by a backslash. So instead of writing 7/8, it shows up as 7\/8).

By obvious cases, I mean that there are no ambiguities or other complications – that the numerical expression will clearly belong to one of those five categories.

Augmentation

Add to your basic program so that it handles:

- Numerical expressions that don't fall into one of the above five categories.
- Numerical expressions that are a combination of the above five categories
- Numerical expressions that may be ambiguous as to which categories they belong.

Data file

Here is a sample input text file, and here is the corresponding expected output for those sentences. Additionally, here are some more (unannotated) text to help you develop your regular expression patterns. The data files have already undergone one pass of preprocessing: the punctuations have been separated into their own tokens, and left and right parentheses have been replaced by the special tokens "-LRB-" and "-RRB-" respectively; the forward slash has been escaped (\/).

What to Turn In

- All source codes, clearly documented so the TA understands what you did.
- A README.txt file to help the grader understand your program
    - It should describe how to run your program.
    - It should list known bugs (if any)
    - It should state whether you've attempted just the basic requirement or if you've attempted the augmented portion -- if so, what cases does your program handle?
- A short report (in pdf, docx, or txt). It should discusses the following issues:
    - For each of the specified types, what are some variations that your program can cover?
    - What was your strategy for identifying variations, given that you probably don't want to read every sentence?
    - Were there difficult cases that your program cannot correctly handle using regular expression matching? If so, what additional tools do you think might be needed?

Grading Scheme

The assignment will be graded on a non-linear 5-point scale.

- 5 (equivalent to 100): Excellent work. The program has satisfied all the basic requirements; substantial work has gone into the augmentation phase; the report is clear and insightful.
- 4 (equivalent to 93): Very good work. The program has satisfied all the basic requirements, and it has made a good attempt at augmenting over the basics; the report addresses all the issues we asked about.
- 3 (equivalent to 80): OK work. The program has satisfied all the basic requirements and made a good attempt at augmenting over the basics, but there were some minor bugs; the report cursorily addresses the issues we've asked about.
- 2 (equivalent to 60): Basic work. The program has satisfied all the basic requirements, but the augmentation portion is missing or just too simplistic; OR, the report is skimpy (even if the program were substantial and correct).
- 1 (equivalent to 40): Incomplete work. Only a portion of the assignment is completed.
- 0 (equivalent to 0): no submission.

Additional Grading Comments from the TA/Grader

It is recommended (though not required) that the you try to run your programs on the Windows/Unix lab machines in Sennott Square before submission. Write a script called run.sh or run.py or anything that can run the program and print the output. It is not the grader's job to track down unusual packages or tools that your program needs. If your program does not produce any output or if it does not compile, then the TA will have to grade you solely on the second part of the report ("What was your strategy for identifying …") with a score of at most 1.

If partial results are printed out on screen or written to a file before the program crashes, the TA will grade based on how much the output covers the basic part and the augmented part. Here are two example scenarios:

Example Scenario 1: Your program crashed after it had printed the output of three cases in the basic part. This means you have not met the basic requirement. Even if your program can produce the output for the augmented part when the code for the basic part is skipped, the augmented part will be ignored.

Example Scenario 2: If your program crashed after it had printed the output for all basic cases and two cases in the augmented part, then we will consider you to have made a good attempt at augmenting over the basics.

Submission

Please package everything up into a zipped file. Then you should upload it at this submission URL:
https://intranet.cs.pitt.edu/~hwa/cs1671/subs/

- Make sure that you have vpn running if you are not directly on Pitt's network.
- Make sure that your url says "intranet" rather than "www"
- If you cannot reach the submission page, please try the following:
    - Copy the following link onto a clipboard: https://sremote.pitt.edu/~hwa/cs1671/subs/,DanaInfo=intranet.cs.pitt.edu,SSL+index.php
    - Log in at sremote website using your Pitt network ID and password – https://sremote.pitt.edu/cs
    - Paste the link you previously copied into the address bar
- If you can reach the site OK, but you are having trouble submitting, make sure that you are using your pitt email login (without the pitt.edu part) as your username and your Peoplesoft number as the password (no leading or trailing blanks).