SLP Assignment D September 29, 2020

**Regular Expressions**

**D1. Tokenization**

Tokenizing is the task of cutting a string into identifiable linguistic units. Many toolkits include tokenizers, but here we will consider for ourselves what this involves (following Steven Bird et al., 2009).

`When *I'm* a Duchess,' she said to herself, (not in a very hopeful tone though), `I won't have any pepper in my kitchen *at* *all*. Soup does very well without--Maybe it's always pepper that makes people hot-tempered,' she went on …

Here the tokens would be “when”, “I’m” and so on. Write a regular expression describing all word separators --- involving whatever sequences of newlines, spaces and punctuation they may include --- and a FSA for your regular expression.

My regular expression:

/(;|:|,|.|”|’|?|!|(|)|-|\_|--|\n|\s|\t)\*/

My FSA  
A close up of text on a whiteboard

Description automatically generated

5 points. May be done in pairs. Due October 1.

**D2: Finding Syllables**

In this assignment, adapted from one by Jason Baldridge, you will write and test a simple regular expression and apply it. Specifically, you identify how many words in the War of the Worlds passage below are “one syllable words.” Use the (imperfect) heuristic that one syllable words consist of zero or more consonants, followed by one or more vowels, followed by zero or more consonants (i.e., there is only one group of vowels in the word). In answering this question, you should not distinguish between capitalized and non-capitalized words. The vowels to use are *a*, *e*, *i*, *o*, and *u*. All other letters are to be considered consonants. Hyphenated constructs such as *knee-deep* are to be counted as two words.

After the glimpse I had had of the Martians emerging from the cylinder in which they had come to the earth from their planet, a kind of fascination paralysed my actions. I remained standing knee-deep in the heather, staring at the mound that hid them. I was a battleground of fear and curiosity.

I did not dare to go back towards the pit, but I felt a passionate longing to peer into it. I began walking, therefore, in a big curve, seeking some point of vantage and continually looking at the sand heaps that hid these new-comers to our earth. Once a leash of thin black whips, like the arms of an octopus, flashed across the sunset and was immediately withdrawn, and afterwards a thin rod rose up, joint by joint, bearing at its apex a circular disk that spun with a wobbling motion. What could be going on there?

From Chapter 5 of *The War of the Worlds*, by H. G. Wells.

Turn in your code and the count of words. 5 points. May be done in pairs. Due October 1.

**D3. Date Conversions**

In this assignment, also adapted from one by Jason Baldridge, you will write a simple format conversion program for dates. There are a lot of different ways that you can express a particular date, e.g. “September 5, 2017”, “09-05-2018”, “9/5/2018”, “the 5th of September, 2011”, and “next Wednesday”.

a) Write a regular expression that covers date expressions of the form “MONTH DAY, YEAR”, where

* MONTH is one of January, February, . . . , December
* DAY is a number between 1 and 31 (You do not need to check that you don’t match February 31 and similar impossible dates.)
* YEAR is a number between 1900 and 2099, so you only need to check years in the 1900s or 2000s

b) Write a regular expression ShortDate that covers date expressions of the form M/D/Y and M-D-Y where

* M is a number 1 to 12, with single digit numbers optionally preceded by 0, e.g. 01, 02.
* D is a number 1 to 31, with single digit numbers optionally preceded by 0, e.g. 01, 02.
* Y is a number 1900 to 2099

c) Write a function normalizeDate that takes as input a date in any of the formats above and outputs it in normalized format YEAR-MONTH-DAY, e.g. /September 7, 2011/, /09-07-2011/ and /the 7th of September, 2011/ all are normalized to /2011-9-7/.

Hand in your code and evidence that it works. 5 points. Due date TBD.

**D4: A Simple Chatbot**

While the best chatbots today involve sophisticated response lookup based on semantic similarity models, often including flow models based on neural networks, in this assignment you will create a simple chatbot using pattern matching.

1. Chose a domain and user population. For example, you may chose freshmen complaining about parking, fans talking about UTEP football, or CS students talking about favorite professors.
2. Study the domain, ideally by obtaining a small collection of dialogs.
3. Write some patterns that capture some of the response behaviors you observe or that you would otherwise like your system to handle.
4. Write an engine, perhaps in python, that reads user input and responds by lookup based on pattern matching, probably using regular expressions.
5. Test it and note some weaknesses --- for example, a lack of robustness to capitalization or inflection or synonyms or various grammatical forms.
6. Improve your system.
7. Run a test with a user who is not one of the developers. Save a transcript of the interaction. Note at least 2 things that went well and 2 things that did not.

Write a report showing or describing what you did, with sections for items 1 through 7.

Come prepared to give a live demo to the class. Due date TBD. 15 points. To be done in pairs.