COP290 Major: Design document

Vishal Bindal
2018CS50425

I've used the following tokens:

WORD: `[a-zA-Z0-9]+`

WORDSP: `[vishalVISHAL]{1,3}` : This denotes the 'special word' for part C, where I check if the word has 1-3 letters, each being in my name. After matching function distinct() checks if they are all distinct (since all letters in my name are distinct)

[NOTE: For part C, I've assumed question's interpretation to be: Maximum 3 letters in the word, each letter part of my first name, no repetition (since 'permutation' was mentioned)]

PUNCT: `[,:;"']` : punctuation symbols

EOS: `[\.?!]` : End Of Sentence, can be '.', '?' or '!'

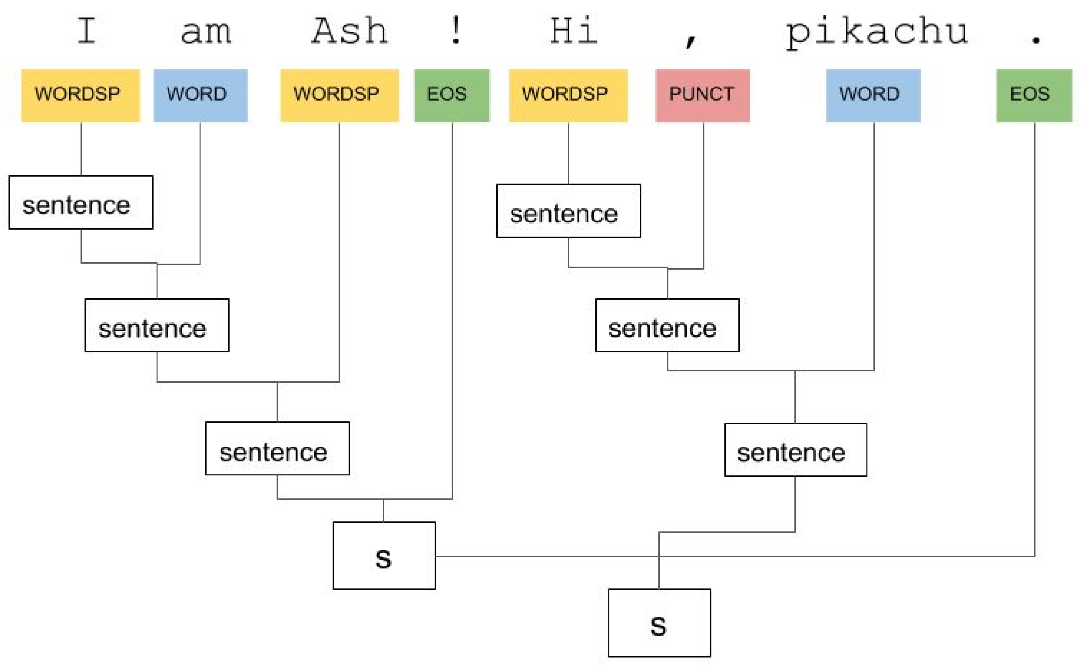
I've ignored other characters, including spaces, newlines, etc

The grammar is:

s (start) : sentence EOS, or s sentence EOS

sentence: WORD or WORDSP or PUNCT or sentence WORD or sentence WORDSP or sentence PUNCT

A demonstration of lexing and parsing logic is given below. Here, "I", "Ash" and "Hi" are special words as they contain 1-3 distinct letters of my first name.



Design for part D

I keep the following variables:
int cur_line_words25 = 0;
int cur_line_punct25 = 0;
int total_punct25[2] = {0,0};
Here cur_line_words25 denotes the no of words in the current sentence. cur_line_punct25 denotes no of punctuations in the current sentence. total_punct25 denotes required value of punctuations i.e. value of ‘d’ in the 2 documents.

While parsing,
whenever PUNC is encountered, cur_line_punct25 is incremented by 1.
whenever WORD or WORDSP is encountered, cur_line_words25 is incremented by 1.
when EOS is encountered, it is checked if cur_line_words25 is greater than or equal to 10. If yes, then it means that no of words in the sentence were >= 10, so the value of cur_line_punct25 is added to the corresponding position of total_punct25 array (depending on if it’s doc1 or doc2). First 2 variables are then again initialised to 0.

The final ‘d’ value for document is taken to be total_punct25[i] + 1, since End of sentence character (full stop, exclamation, question mark) is also taken to be punctuation.

Design for other parts

Following variables are kept:
int single_letter25[2] = {0,0};
three_letter25[2] = {0,0};
special_words25[2] = {0,0};
Whenever a WORD or WORDSP is encountered, the ‘len’ field of its struct ‘data’ is checked. If 1, then relevant position of single_letter25 is incremented, if 3 then relevant position of three_letter25 is incremented. When WORDSP is encountered, special_words25 is also changed accordingly.

Other design choices

I chose type of WORD and WORDSP to be a struct containing 2 fields, the text and its length. I did this because parts a-d need length value while part e needed text. However, I couldn’t complete part e on time

Assumptions

1) The sentence ends only at full stop, exclamation mark or question mark
2) Punctuation marks act as delimiters for words in a sentence, e.g. for a word “hasn’t” it will be taken as 2 words, “hasn” and “t” separated by a punctuation