Generating Sentences

For this lab, you will build an English sentence generator. That is, you will write a program that will be able to produce random English sentences, and English sentences containing specific words.

Some Background

Natural Language Processing (NLP) deals with making computers understand what people mean when they speak or write. This is a much harder problem than it might seem. Some of the challenges include ambiguity, the use of idioms, and the use of sarcasm. Such sentences as "Will Will will the will to Will?" are difficult for people to understand, much less a computer. People are much better at figuring out meaning based on context clues. For example, take the word "bank". You could say "I need to deposit money in the bank." You could also say "The men are fishing from the bank of the river." Needless to say, NLP has had to overcome numerous challenges, and they still aren't all solved!

For your lab, you will create a program to generate English sentences. We aren't concerned with understanding the sentence. Generating a sentence (in any language) that makes sense in context is also a hard problem. Ever talked to a chat-bot? It's very hard for a computer to grasp the *meaning* of a sentence. For example, say you have the sentence "That movie was great. I was able to catch up on the sleep I was missing." A computer is likely to think that this sentence means the person actually liked the movie since it contains the word "great". However, we can see that the person didn't like the movie at all since he/she fell asleep during the movie.

For this lab, you'll just generate random sentences for fun.

Program Requirements

A grammar is given below that describes several syntactically correct forms of English sentences. You will use this grammar to generate sentences. Your program should be able to generate random sentences that are grammatically correct. They most likely will not make sense, however.

The Grammar

S: Sentence
NP: Noun Phrase
VP: Verb Phrase
PP: Prepositional Phrase (on the couch, in the class, over the hill, ...)
ART: Article (a, the, an, ...)
NOUN: Noun (boy, dog, rug ..) (person, place, thing, or idea)
VERB: Verb (run, walk, swim, is, are, were ...)(action or state of being words)
ADJ: Adjective (purple, fuzzy, sharp, fast ...) (words that describe nouns)
ADV: Adverb (lazily, quickly, slowly ...) (words that describe verbs, often end in "-ly")
PREP: Preposition (on, over, through, in, around ...) (words that fit in the sentence "The bunny went *PREP* the log")

$$
\begin{aligned}
<S> &::= <NP><VP> \\
<NP> &::= <NOUN> \mid <ART><NOUN> \mid <NP><PP> \\
<VP> &::= <VERB> \mid <VP><NP> \mid <VP><ADJ> \mid <VP><ADV> \\
<PP> &::= <PREP><NP>
\end{aligned}
$$

Generating a Random Sentence

Generate a sentence that follows the grammar by picking random substitutions from the grammar and fill in with random words from the word lists. I have provided lists of Nouns, Verbs, Adjectives, Adverbs, Articles, and Prepositions. You may modify these lists if you want, but keep the words safe for work! For example, one sentence could be

S
NP VP
NP PP VP
NP PP VERB
ART NOUN PP VERB
ART NOUN PREP NP VERB
ART NOUN PREP ART NOUN VERB
The bunny in the cage slept.

Testing

Have your program generate 4 random sentences. The sentences should have variations in them (i.e. you can't pick all of the NOUN VERB sentences). For each sentence, show that the sentence is a valid sentence. **Turn in a pdf of screen shots of the sentences and the derivations.** You may use derivation trees.

Example

Sentence: a water prefer many research like place near history behind the night
Words given: None
Production:
(S)
(NP) (VP)
(NP) (VP) (NP)
(NP) (VP) (NP) (PP)
(NP) (VP) (NP) (PREP) (NP)
(NP) (VP) (NP) (PP) (PREP) (NP)
(NP) (VP) (NP) (PREP) (NP) (PREP) (NP)
(NP) (VP) (NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(NP) (VP) (ADV) (NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(NP) (VP) (ADV) (NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VP) (ADV) (NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN)(VP) (ADV) (NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV)(NP) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NOUN) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NOUN) (PREP) (NP) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NOUN) (PREP) (NOUN) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NOUN) (PREP) (NOUN) (PREP) (NP) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NOUN) (PREP) (NOUN) (PREP) (NOUN) (PREP) (NP)
(ART)(NOUN) (VERB) (ADV) (NOUN) (PREP) (NOUN) (PREP) (NOUN) (PREP) (NP)
(ART)(NOUN)(VERB)(ADV)(NOUN)(PREP)(NOUN)(PREP)(NOUN)(PREP)(ART)(NOUN)
a water prefer many research like place near history behind the night

You should use color or italics or bold to show which non-terminal you are replacing at each step

Hints and Tips

- I have given you a Dictionary class that reads in the lists of words and creates vectors containing each part of speech. You can then get random words. You don't have to use it, but it'll probably make your life easier. You may also modify this class.
- You may assume that the words the user gives you are actually the part of speech they claim. For instance, if you ask for a noun, you may assume that the user gives you a noun.
- I suggest having at least the following functions. The more you break up your code, the easier it will be to piece everything together. Recursion is also your friend!!!
  - string generateNounPhrase()
  - string generateVerbPhrase()
  - string generatePrepPhrase()

Check list

Your code must have a name header as described in the syllabus. Each of these items is worth two points of your grade.
Your output must

- Have a title in the output
- Include a short paragraph indicating the purpose of the application to the user
- Use complete and clear sentences – do not assume the user knows what you expect from him/her.
- Have NO grammatical errors and/or spelling errors.
- Have NO typos – this includes using capital letters at the beginning of a sentence and punctuation at the end of sentences.

Example Output

Here are some examples for what your program's output could look like. Yours doesn't have to look exactly like this. User input is in blue. The user's words are in all capital letters to make them stand out in the generated sentence.

```
Title Goes Here
----------------------------
Purpose Goes Here
---------------------------

Choose an option:
1.  Generate Random Sentence
2.  Quit
1

guy copy education along a side after a case

Choose an option:
1.  Generate Random Sentence
2.  Quit
1

the name tell financial east an system same

Choose an option:
1.  Generate Random Sentence
2.  Quit
1

the government concerning a office anti a eye down the idea cry unusual little

Choose an option:
1.  Generate Random Sentence
2.  Quit
3
```

What to Turn In

Upload your .cpp and .h (if you have any) files and the pdf of your output and derivations as described in the program requirements to Canvas.