515 HOMEWORK 3:
PREDICTIVE TEXT GENERATION

Your assignment is to use the Datamuse API to generate a predictive text-driven Haiku on a topic of the user's choice. Like ChatGPT, you will be generating your own short text response based on a user's input!

About the haiku

Your program should first request that the user provide a topic to begin the Haiku generation. Then, your program should connect to the Datamuse API to find related words that satisfy the requirements of a Haiku (described below). Print your completed Haiku for the user to enjoy. Finally, your program should allow the user to generate as many Haikus as desired.

For the purposes of this assignment, the requirements of Haikus are:

- A Haiku is a 3-line poem, where the first line contains 5 syllables, the second line contains 7 syllables, and the third line contains 5 syllables.
- The last words of each line in the Haiku must rhyme.
- A Haiku may not use the same word more than once.

Using the Datamuse API

The Datamuse API supports several forms of queries that will assist you in this assignment. For the first word of the Haiku, choose a word related to the topic selected by the user. Use "https://api.datamuse.com/words?md=s&rel_trg=**word**" to search for a word related to your user's topic (replace "**word**" in the URL above with whatever topic you would like to search). This mode of the API searches for "related triggers," which the API abbreviates as "rel_trg." These words are not necessarily synonyms, but rather words that follow the same topic. For example, a related trigger of "cow" might be a word like "milk." These two words are not synonyms but pertain to the same topic. For example, if your topic were "food," you might use "https://api.datamuse.com/words?md=s&rel_trg=food" to search for related words. The first few words returned by this search are "FDA", "beverage", and "nutrition".

For the next words of your Haiku, rather than using related triggers, instead use "https://api.datamuse.com/words?md=s&sp=*&lc=**word**" to search for words that commonly follow the previous word. The "lc" in the URL stands for "left context," meaning that the word you search commonly appears to the left of (just before) whatever words the API returns. For example, if the previous word in your Haiku were "jump," then you might use "https://api.datamuse.com/words?md=s&sp=*&lc=jump" to search for words that commonly appear after "jump." The first few words returned by this search are "to", "in", and "into".

At the end of each line of the Haiku, you must meet the rhyming requirement mentioned above. To do so, use "https://api.datamuse.com/words?md=s&sp=*&lc=**word**&rel_rhy=**rhyme**" to search for words that are both related to the previous word in the sentence and rhyme with the end of each line. This URL uses two parameters in its querystring: the "lc" or left context to search, and the "rel_rhy" or rhyming word to match. For example, suppose that you needed a word that commonly followed "by" in a sentence and also rhymed with "grape." In this case, you might use "https://api.datamuse.com/words?md=s&sp=*&lc=by&rel_rhy=grape" to search for words. The first few words returned by this search are "shape", "tape", and "escape".

The Datamuse API returns its responses in a JSON array format. The API attempts to determine the words that are the best fit for your query, and it lists what it believes to be the most likely fits first, ranked by a score. For example, you may expect responses to use the following format:

```
[{"word": "fda",          "score": 1448,       "numSyllables": 2},
{"word": "beverage",      "score": 1437,       "numSyllables": 3},
{"word": "nutrition",     "score": 1382,       "numSyllables": 3}]
```

The Datamuse API is not perfect. As you can see above, it reports that the word "FDA" is 2 syllables (probably "F-DA"), whereas it is usually pronounced as 3 syllables ("F-D-A"). For the purposes of this assignment, you may assume that the results of the API are valid, and you do not need to correct any mistakes that the API may make.

Making a haiku

Allocating words to use exactly 5, 7, and 5 syllables in each line can be a complicated problem. For the purposes of this assignment, you may have some flexibility for how you address that problem. One approach, which is strongly recommended, is to always use a fixed number of syllables for each word in the Haiku. For example, the implementation in the example below uses (3-2) to achieve 5 syllables in the first line, (3-2-2) to achieve 7 syllables in the second line, and (3-2) achieve to 5 syllables in the third line. You may experiment with different schemes, or you may use the same scheme mentioned here. Occasionally, using this approach, you may find that some topics cannot result in valid Haikus because of a lack of related and/or rhyming words. Should this occur, simply print a message stating that no valid Haiku could be generated.

Since the API returns a JSON array, one approach for each word in the poem is to create a list of the words that meet the syllables requirement. If the list contains some word(s) after processing the entire JSON array (that is, the length of the list is greater than zero), then you can choose the first word in the list (index zero) and continue to the next word. On the other hand, if the length of the list is zero, then you may determine that no valid Haiku can be generated.

Last considerations

Ensure that the entirety of your program is case *in*sensitive. Each time that the user provides a response to one of your prompts, ensure that the response would be treated the same way whether it is typed in lowercase, uppercase, or a mixture of the two.

Consider the possibility that, when attempting connection to the Datamuse API, some connection issue occurs (that is, a status code other than 200). Ensure that your code handles this case and provides the user with a helpful printout if it does occur.

The words returned by the Datamuse API are always in lowercase formatting. Capitalizing word(s) in your Haiku is not required. Additionally, adding other punctuation such as commas or periods to your Haiku is not required.

Adding comments in your code is encouraged. You may decide how best to comment your code. At minimum, please use a comment at the start of your code to describe its basic functionality.

Please remember that all homework assignments in this course is considered individual assignments. While it is acceptable to discuss high-level ideas with others, you should write and submit only your own code.

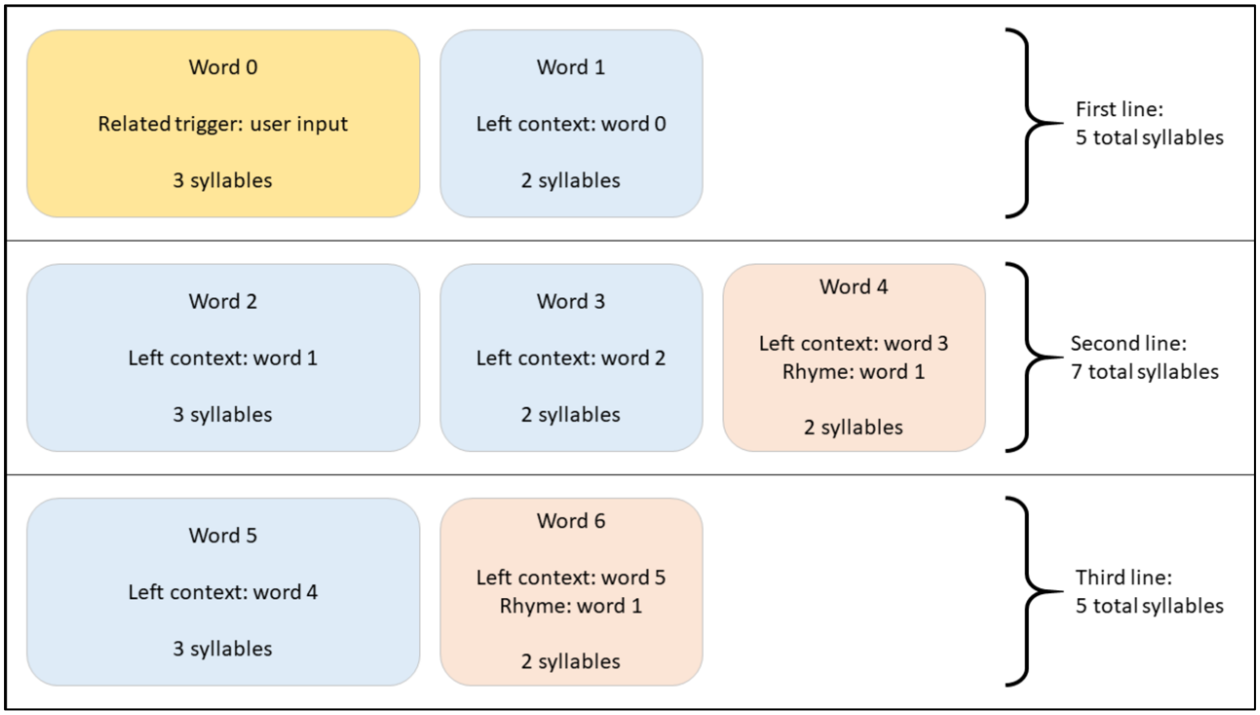
When submitting your assignment, please upload a Python file (.py) to Canvas. If working in Google Colab, you can export your work as a Python file under File > Download .py.

Please design your program based on the following examples. Datamuse API URLs are printed out here for demonstration purposes, but you do not need to include them in your final code.

```
Hello, welcome to the predictive text Haiku generator!
What would you like to see a Haiku about? teeth

URLs:
https://api.datamuse.com/words?md=s&sp=*&rel_trg=teeth
https://api.datamuse.com/words?md=s&sp=*&lc=incisors
https://api.datamuse.com/words?md=s&sp=*&lc=erupt
https://api.datamuse.com/words?md=s&sp=*&lc=suddenly
https://api.datamuse.com/words?md=s&sp=*&lc=became&rel_rhy=erupt
https://api.datamuse.com/words?md=s&sp=*&lc=corrupt
https://api.datamuse.com/words?md=s&sp=*&lc=practices&rel_rhy=er
upt

Haiku:
incisors erupt
suddenly became corrupt
practices disrupt

Would you like to see another Haiku (yes/no)? yes
What would you like to see a Haiku about? smell

URLs:
https://api.datamuse.com/words?md=s&sp=*&rel_trg=smell
https://api.datamuse.com/words?md=s&sp=*&lc=olfaction
https://api.datamuse.com/words?md=s&sp=*&lc=
https://api.datamuse.com/words?md=s&sp=*&lc=understand
https://api.datamuse.com/words?md=s&sp=*&lc=our&rel_rhy=
https://api.datamuse.com/words?md=s&sp=*&lc=
https://api.datamuse.com/words?md=s&sp=*&lc=undertake&rel_rhy=

Haiku:
Sorry, a valid Haiku could not be generated.

Would you like to see another Haiku (yes/no)? YES
What would you like to see a Haiku about? kindness

URLs:
https://api.datamuse.com/words?md=s&sp=*&rel_trg=kindness
https://api.datamuse.com/words?md=s&sp=*&lc=gentleness
https://api.datamuse.com/words?md=s&sp=*&lc=towards
https://api.datamuse.com/words?md=s&sp=*&lc=another
https://api.datamuse.com/words?md=s&sp=*&lc=person&rel_rhy=towar
ds
https://api.datamuse.com/words?md=s&sp=*&lc=records
```

```
https://api.datamuse.com/words?md=s&sp=*&lc=indicate&rel_rhy=tow
ards

Haiku:
gentleness towards
another person records
indicate rewards

Would you like to see another Haiku (yes/no)? no
```

Below is a diagram of the scheme used in this implementation:



Word 0
Related trigger: user input
3 syllables
Word 1
Left context: word 0
2 syllables
First line:
5 total syllables
Word 2
Left context: word 1
3 syllables
Word 3
Left context: word 2
2 syllables
Word 4
Left context: word 3
Rhyme: word 1
2 syllables
Second line:
7 total syllables
Word 5
Left context: word 4
3 syllables
Word 6
Left context: word 5
Rhyme: word 1
2 syllables
Third line:
5 total syllables