COP 4521 Spring 2020
Extra Credit Homework 1

Total Points: 100
Due: Thursday 04/09/2020

1 Objective

The objective for this assignment is to make sure

- You have a working setup for Python outside linprog. Most of the homeworks and class examples for this class will involve libraries that are not installed ion linprog.
- You can write small to medium level programs in Python, using certain libraries.
- You can handle file and console input.

2 Specifications

For this project, you are required to build a solver and a checker for the game Doublets, also known as Word Ladders. This game, invented by Lewis Caroll, involves forming a chain of words between two words, where each word in the chain differs from the previous word in the chain by only one letter. You can find more information on the game here: `https://en.wikipedia.org/wiki/Word_ladder`

For example, the pair WARM-COLD would result in the following word chains :

W	A	R	M	W	A	R	M
W	O	R	M	W	O	R	M
W	O	R	D	W	O	R	K
C	O	R	D	C	O	R	K
C	O	L	D	C	O	R	D
				C	O	L	D

As shown above, there are several correct answers. You have to find the shortest chain. You will be given a CSV file containing word pairs. Each line of the file will be a word pair which can be used to play the game. A run of the program will involve reading this file and then choosing a word pair randomly. You will then solve the problem to obtain the length of the shortest word chain between these words. Only real English words are allowed. You can use the Enchant dictionary found here: `https://pypi.org/project/pyenchant/` to check for legitimate words.

Once you have solved the problem, you let the user play the game. Present the user with the two words and read in a series of words from the user. The user will enter "q" when they are done. You will now check if the user has a legitimate solution and if so, check if they have the best solution.

For this project, we will restrict the game to words with 4 or 5 letters. you are allowed to use the `random` and `enchant` libraries. Use of the `future` built-in is allowed for all projects. Using any other library will result in significant penalties. Turn in `tarball` of all your files to Canvas.

Please make sure your program conforms to the following requirements.

- Please start off all your source files with a docstring comment containing your name and FSU ID (5 points)
- You can use the CSV file on the course website for testing. Feel free to add more word pairs. The word pairs in the CSV file are all valid, that is, there is at least one solution for that pair.
- Read in the values in the CSV file. A list of lists is suggested, but you can use pretty much any data structure that allows for random selection. (10 points)
- Randomly select a word pair. (5 points)
- Using the enchant dictionary to make sure the intermediate words exist in the English language, form the shortest word chain between the words. We are not restricting you on the algorithm, though we suggest some form of dynamic programming or BFS. (50 points)
- Display the word pair to the user and get their input. You can assume that the user will only enter words of the right length. The user will enter "q" when they are done. (10 points)
- Check to make sure all the words entered by the user exist in the dictionary. Even if one word is made up, print the first made up word, tell the user it does not exist and exit the program. Next, check to make sure the words are following the constraint of only changing one letter. If this is violated even once, tell the user and exit the program. (10 points)
- If all the words are valid, check if the words form a legitimate word chain. If so, tell the user. Then check if the word chain is the shortest. If it is, just print a congratulatory message. If not, print the shortest word chain. (10 points)

2.1 The PyEnchant Module

To verify that a word is actually present in the English language, you are encouraged to use the PyEnchant module. The PyEnchant module may be installed on Ubuntu with the following command:

```
$ sudo apt-get install python-enchant
```

An example usage of the dictionary lookup functionality is shown below:

```
>>> import enchant
>>> d = enchant.Dict("en_US")
>>> d.check("Hello")
True
>>> d.check("sdjh")
False
```

2.2 Some Approaches to the Solution

- Since there are way too many 4 and 5 letter words in the English language, checking for all words would be an inefficient solution. Try and take advantage of the constraint. Only one letter changes at a time, so you could write an edit distance function and only examine words with an edit distance of 1 from the current word.
- You can store all the words you've already looked to avoid going around in circles.

- You could use Breadth First Search and generate all future possibilities from the current word and prune the less favorable branches.
- You can use recursive backtracking
- The majority of your implementation will likely rely on lists and strings. Make use of list comprehensions for brevity and use built-in string methods to make it easier on yourself.
- Come see me or Michael if you are struggling!

2.3 Sample Runs

Assume the following CSV File:

```
warm, cold
five, four
head, tail
grass, green
```

Here are some sample runs:

2.3.1 Sample Run 1

```
Welcome to the Doublet Game!.
Today's word pair is : head -> tail
Please enter your word chain (not including the start and end words.
Enter 'q' when you're done.
>heal
>teal
>tell
>tall
>q

All the entered words are in the dictionary.
All the words follow the one letter constraint.
You have successfully solved the problem.
Congratulations!. You have also solved the problem with the smallest number of words.
```

2.3.2 Sample Run 2

```
Welcome to the Doublet Game!.
Today's word pair is : grass -> green
Please enter your word chain (not including the start and end words.
Enter 'q' when you're done.
>crass
>cress
>tress
>trees
>frees
>freed
>greed
>q

All the entered words are in the dictionary.
All the words follow the one letter constraint.
```

```
You have successfully solved the problem.
There is a shorter solution:
grass -> trass -> tress -> trees -> treen -> green
```

2.3.3 Sample Run 3

```
Welcome to the Doublet Game!.
Today’s word pair is : five -> four
Please enter your word chain (not including the start and end words.
Enter ’q’ when you’re done.
>hive
>hire
>hour
>four
>q

All the entered words are in the dictionary.
You have violated the one letter change constraint. You lose.
```

2.3.4 Sample Run 4

```
Welcome to the Doublet Game!.
Today’s word pair is : warm -> cold
Please enter your word chain (not including the start and end words.
Enter ’q’ when you’re done.
>wirm
>cirm
>cird
>cild
>q

"wirm" is not a word in the dictionary. You lose.
```