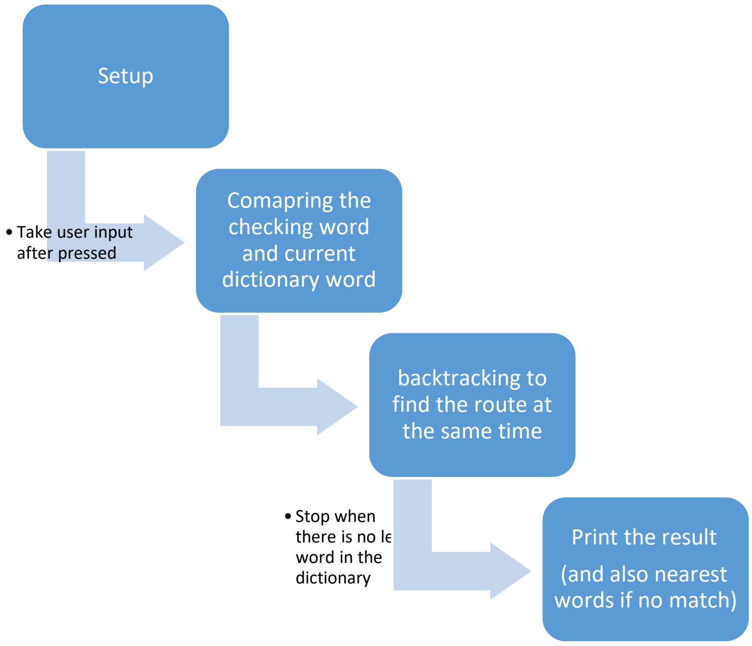
Assignment 2: Spelling Checking
Assignment Report
410421304 資工四 秘子尉

How do I do this assignment?

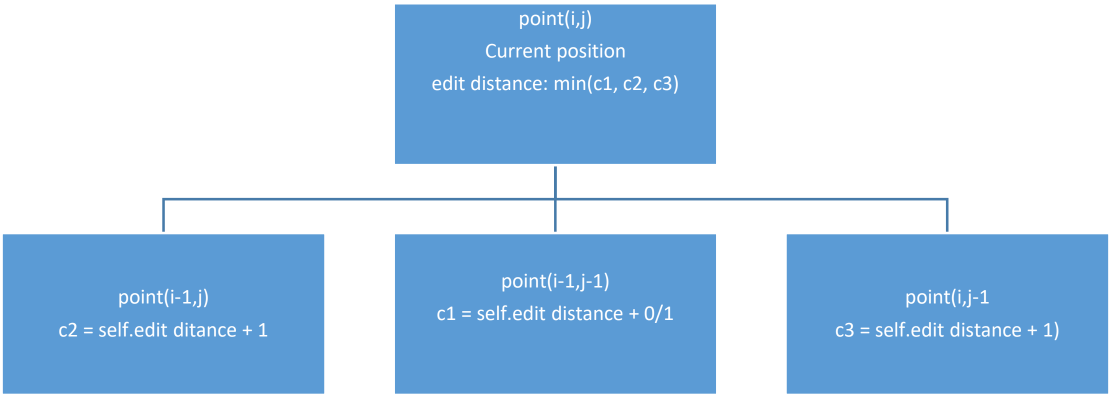
In this assignment, we need to follow the guide to finish the homework, which is to use edit distance, so call Levenshtein distance, to check whether the input word is spelling correct. Giving a dictionary text file, we are going to compare the input word with every words in the dictionary to find out the edit distance between them. If the minimum edit distance of 0 exist in the dictionary file, we can say that the word is correct spelling. Otherwise, the word is not spelling correct, and we will list the words in range of edit distance 1 to 3.

In the programing parts, first we get the words one by one from the dictionary, store them up in a big enough list. In this phase, we also store the words amount, in case we would use it later.

Then, we get the input from user, and compare it one by one to each dictionary words. By using the algorithm we had been given, we implement following code: For each word, first we need to drop the change line character(which we would get these from the text file, compare first character to set to initial distance. Then, we calculate the edit distance, using the algorithm, set up the distance of first row and first column, base on this we could calculate rest distance.


Setup
• Take user input after pressed
Comapring the checking word and current dictionary word
backtracking to find the route at the same time
• Stop when there is no le word in the dictionary
Print the result
(and also nearest words if no match)

For each position, comparing the edit distance of current position(assume I,j), by the one in the left(i-1,j) + 1, in the bottom(I, j-1) + 1, and the edit distance of left bottom corner(i-1, j-1) plus one or zero, by condition of whether the character of this position are the same for the input word and the dictionary word now are the same or not(same is 0, different is 1). These calculation is from the algorithm, when we comparing two words, each modification, include substitute, inserition and deletion, would plus the edit distance by 1. We the edit distance from the beginning(0,0) to the character of the current position is the minimum value of 3 of above. After all calculation, we would get a 2D graph which stores the edit distance from beginning to each position. Then when we take D[I][J], where I is the length of checking word, J is the length of the chosen dictionary word, would return a value shows the edit distance between these two words.


point(i,j)
Current position
edit distance: min(c1, c2, c3)
point(i-1,j)
c2 = self.edit ditance + 1
point(i-1,j-1)
c1 = self.edit distance + 0/1
point(i,j-1
c3 = self.edit distance + 1)

We would store the result of the words which had edit distance 1, 2 or 3 in 3 different list, also check whether we have same character in the dictionary. The rest part are output results. If we had the same word from the dictionary, we will telling the user the spelling is correct; otherwise we would list the word from the list we had store in last process(when calculating edit distance).

Additionally, we would also use backtrack method to track the best route we had chosen in the process. To track the route, we start from the very last point we had been to, saying point(I,J) in the graph with I as the length of the word we are going to check, J as the length of the dictionary word for now. Start from the point, we would examine whether point(I-1,J), point(I,J-1) or point(I-1,J-1) had the minimum word distance. We looping until reach point(0,0). During the process, if the point we are going to examine is not include in our path, we would give it a maximum distance(will not choose this route).

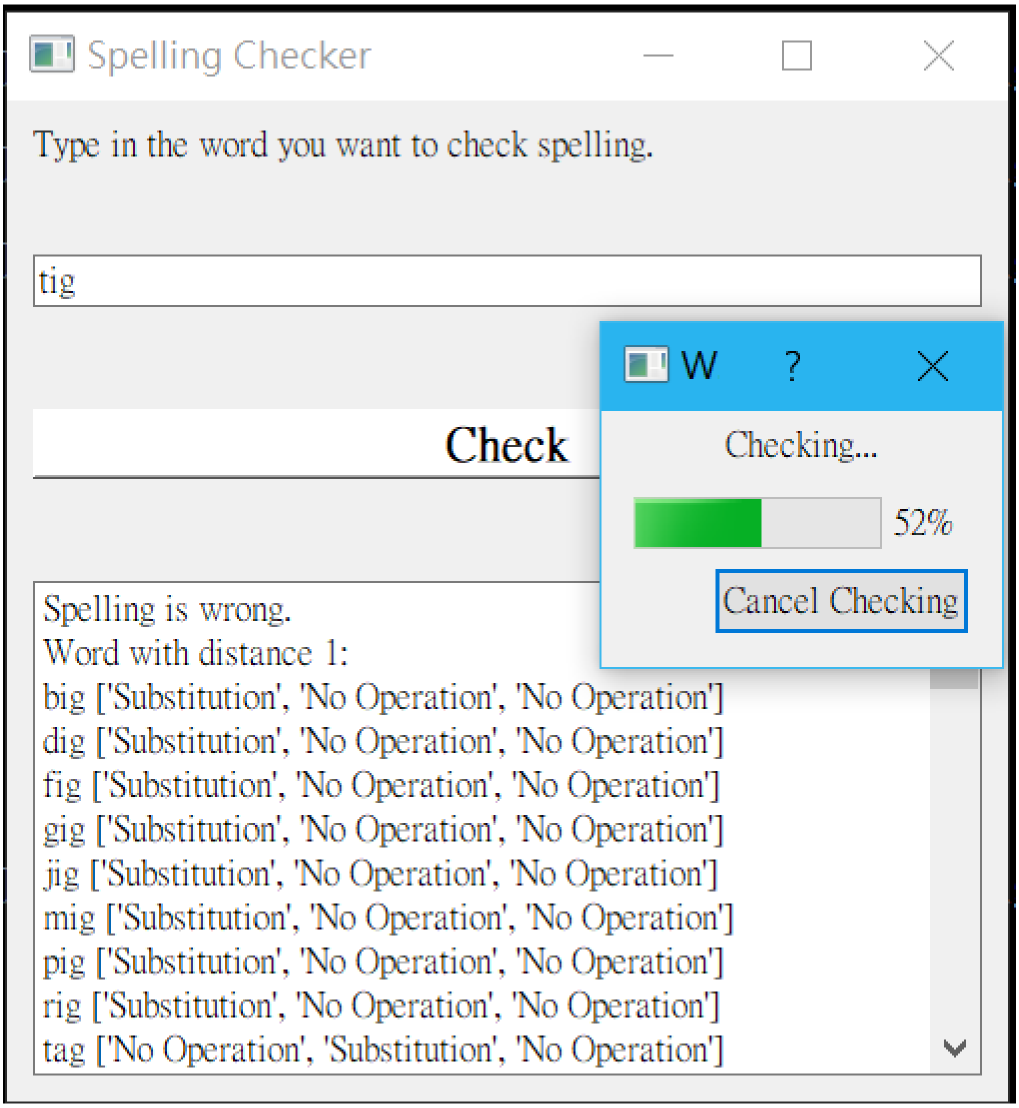
We will store the optimal edit route for every dictionary word which have edit distance below 3.

After all of this, we would append the result to the textedit textbox to show the result. When the process of showing the result is done, the process bar would be terminate and popup a message box showing “Checking Done”, and the spelling checking process is done.

The result would be show in the textbox below. If your world exist in the dictionary, it would show” The spelling is correct!”; otherwise, it would show “The spelling is wrong!”, and print out all the dictionary words near the word we are checking, which are the ones with edit distance lesser than 3, and also the ways to edit the dictionary word to the checking word in the textbox. User can scroll down to see the whole result.

The Result:

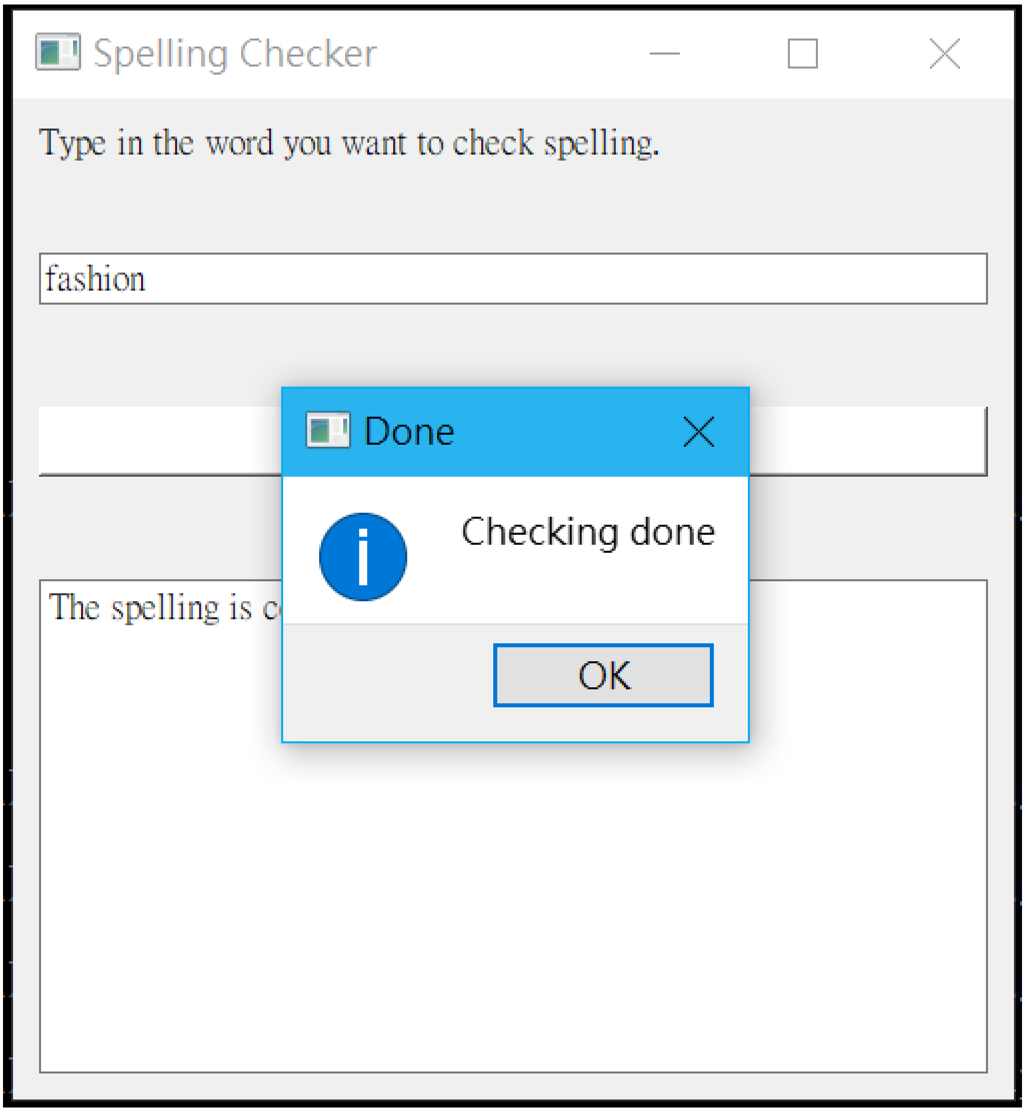
As the picture, we can see the result is well.



Type in the word you want to check spelling.

tig

```
Spelling is wrong.
Word with distance 1:
big ['Substitution', 'No Operation', 'No Operation']
dig ['Substitution', 'No Operation', 'No Operation']
fig ['Substitution', 'No Operation', 'No Operation']
gig ['Substitution', 'No Operation', 'No Operation']
jig ['Substitution', 'No Operation', 'No Operation']
mig ['Substitution', 'No Operation', 'No Operation']
pig ['Substitution', 'No Operation', 'No Operation']
rig ['Substitution', 'No Operation', 'No Operation']
tag ['No Operation', 'Substitution', 'No Operation']
```

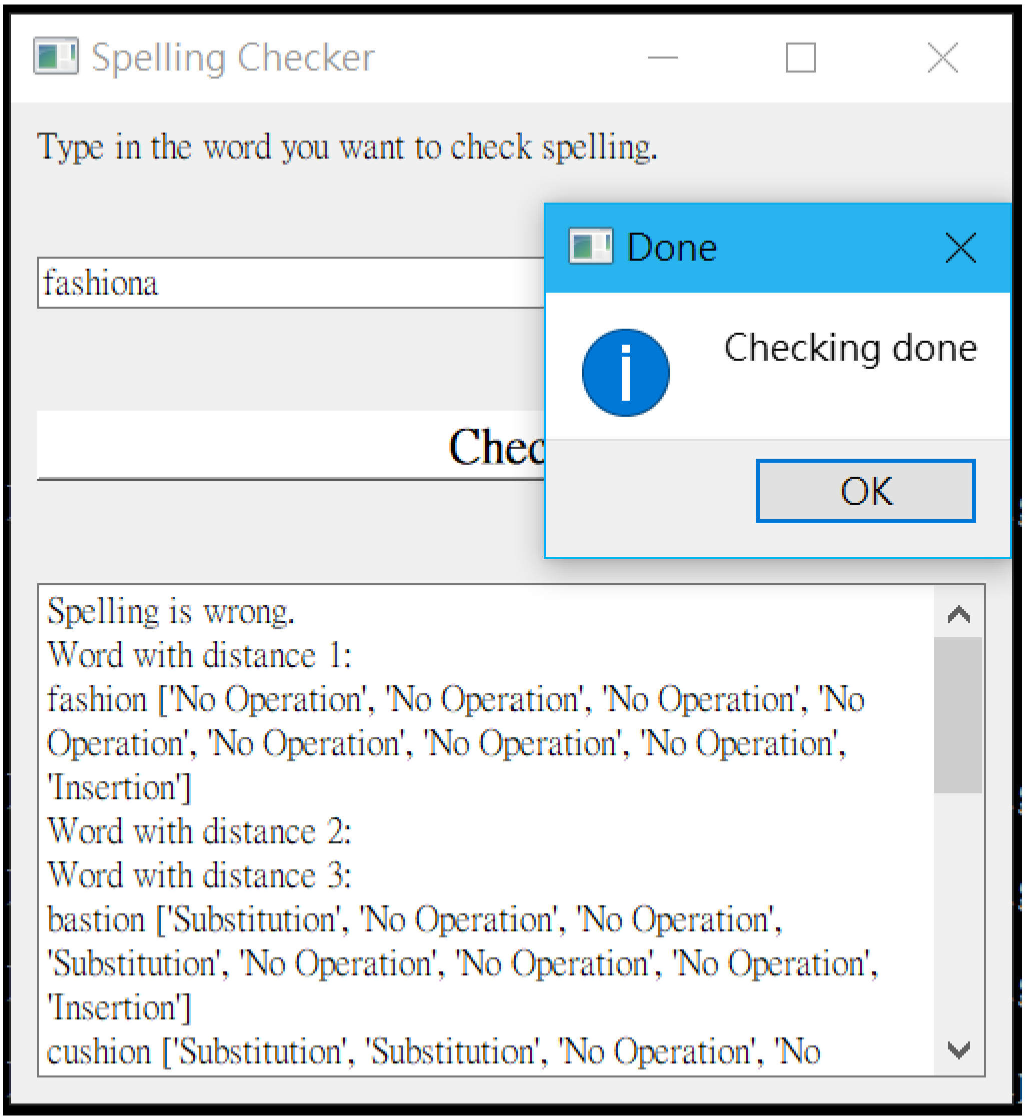


Type in the word you want to check spelling.

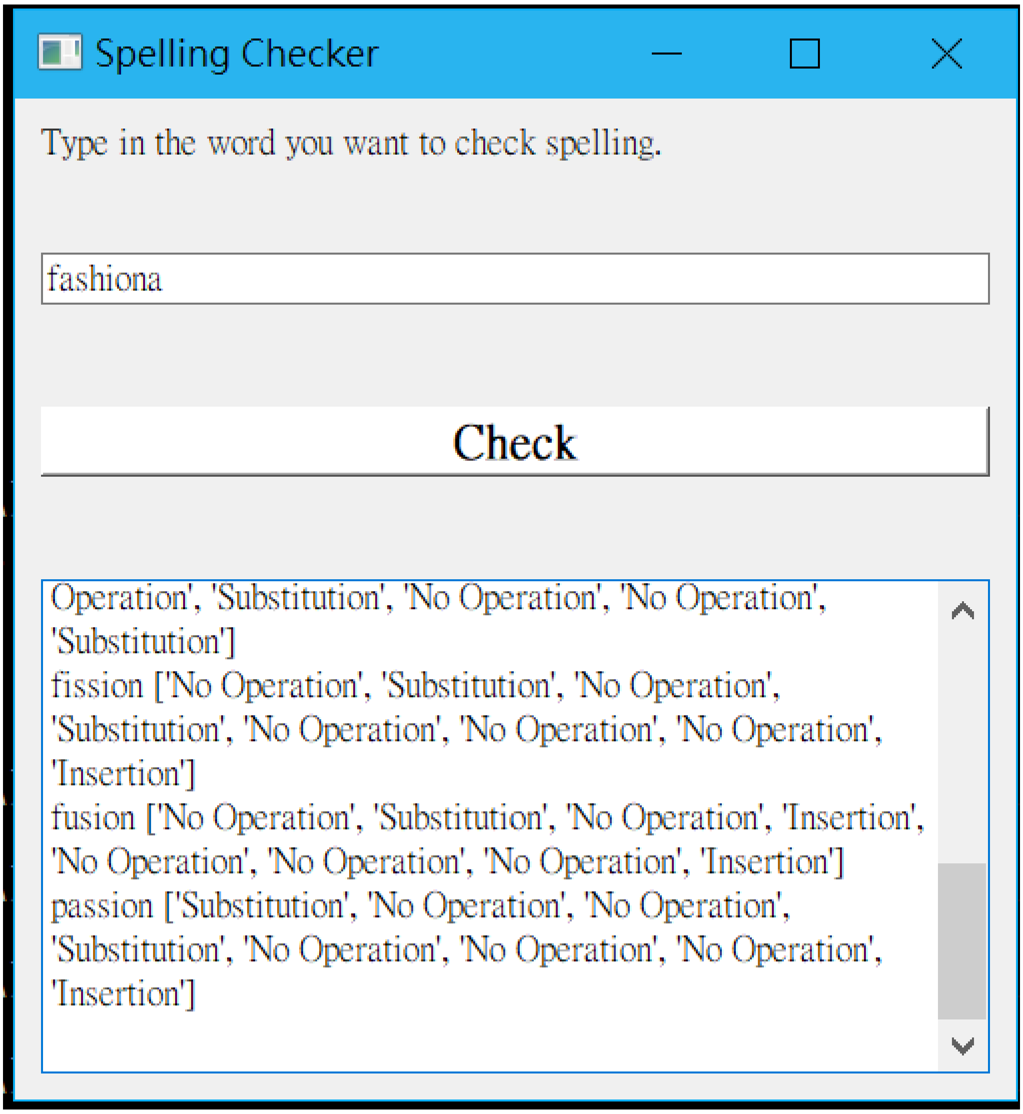
fashion

Checking done

The spelling is c



```
Spelling is wrong.
Word with distance 1:
fashion ['No Operation', 'No Operation', 'No Operation', 'No
Operation', 'No Operation', 'No Operation', 'No Operation',
'Insertion']
Word with distance 2:
Word with distance 3:
bastion ['Substitution', 'No Operation', 'No Operation',
'Substitution', 'No Operation', 'No Operation', 'No Operation',
'Insertion']
cushion ['Substitution', 'Substitution', 'No Operation', 'No
```



Type in the word you want to check spelling.

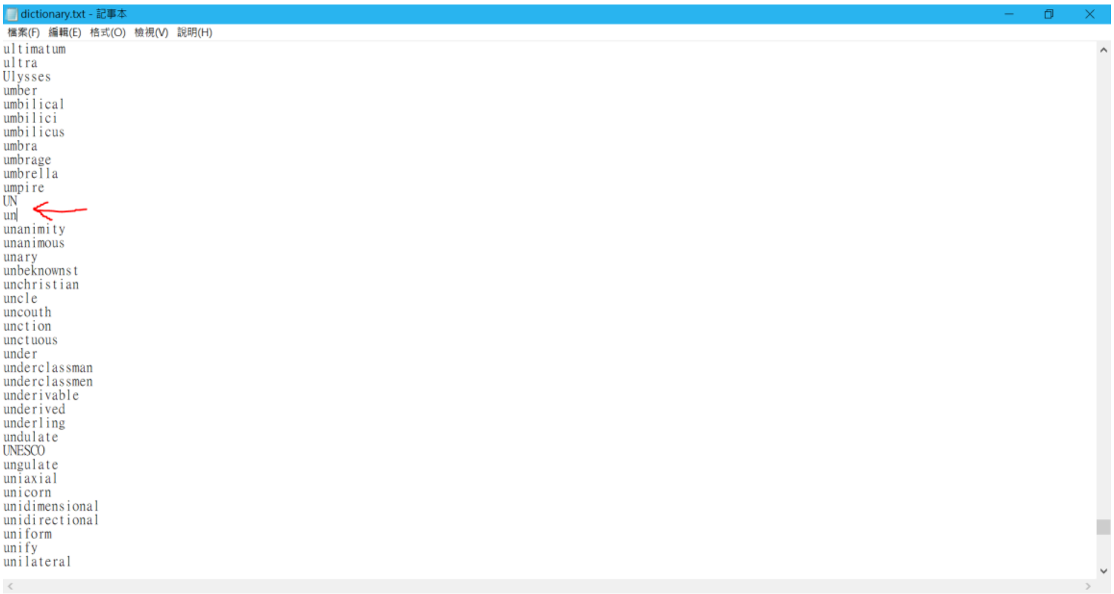
fashiona

```
Operation', 'Substitution', 'No Operation', 'No Operation',
'Substitution']
fission ['No Operation', 'Substitution', 'No Operation',
'Substitution', 'No Operation', 'No Operation', 'No Operation',
'Insertion']
fusion ['No Operation', 'Substitution', 'No Operation', 'Insertion',
'No Operation', 'No Operation', 'No Operation', 'Insertion']
passion ['Substitution', 'No Operation', 'No Operation',
'Substitution', 'No Operation', 'No Operation', 'No Operation',
'Insertion']
```

Discussion on the result:

After try several test data, I found that the word with small word length is not really useful to find their similar word by edit distance, or it could better say that we probably could find the one we want, but we would find a bunch of result that we don't want. For example, we have word 'oig', which we were misspell from the word 'pig', but we would also use a lots of time to find those word within distance 3 and process them, such as 'abc', 'ACM',… these words were really far from the one we want to find. So maybe for shorter words, we should better adjust the maximum distance we want to find.

Uppercase and lowercase is the mistake we often made. Sometimes we would say that we didn't care about the case, but here we would do that. Since those were 'mistakes', here we also count 'Bob' and 'bob' as two different string with distance of 1. And also in this dictionary we could also find UN and un both exist, but they surely have different meaning, so here we are going to count case different.

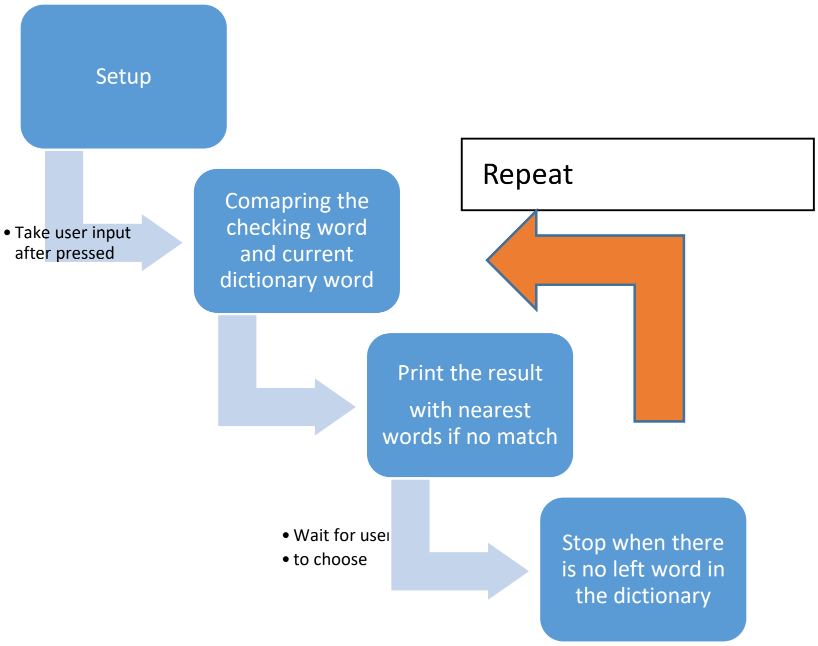


```
ultimatum
ultra
Ulysses
umber
umbilical
umbilici
umbilicus
umbra
umbrage
umbrella
umpire
UN
un
unanimity
unanimous
unary
unbeknownst
unchristian
uncle
uncouth
unction
unctuous
under
underclassman
underclassmen
underivable
underived
underling
undulate
UNESCO
ungulate
uniaxial
unicorn
unidimensional
unidirectional
uniform
unify
unilateral
```

In the dictionary we didn't have much words about "grammar", such as we have 'swim', but not 'swimming.' Thus if we have to deal with the grammar, our dictionary would not work well. These grammar things maybe are better to deal with the toolkit such as NLTK.

Additional Challenge: Paragraph Spelling Checking

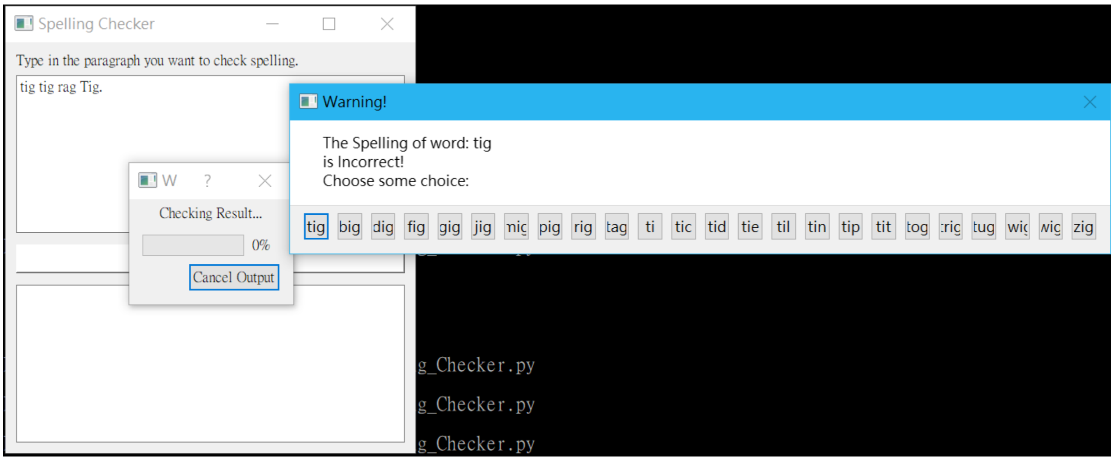
The spelling checking is the easy one, why not we go on a challenge? So I challenge myself to make an at least 'usable' paragraph spelling checker.


Setup
Take user input after pressed
Comapring the checking word and current dictionary word
Repeat
Print the result with nearest words if no match
Wait for user
to choose
Stop when there is no left word in the dictionary

It is kind of a transform of the spelling checker, but need to add few more works. Though some main parts remain unchanged, the structure were changed a lots.

Same as spelling checker, first we initialize our GUI surface. This time we would have two text edit textbox, which one of them replace the previously line edit in order to take more inputs from user. When user put a sentence in the textbox, press check, it will popup a process bar to show the process as well. But this time, we add some additional popup messagebox when the program detect s misspelling word.

The popup window will show the options of the word with edit distance 1 to let the user to choose which one they are going to choose. The word after correction or they were correct originally will be showing in the textbox as soon as they are available.


Spelling Checker
W ?
Checking Result...
0%
Cancel Output
Warning!
tig big dig fig gig jig nig pig rig tag ti tic tid tie til tin tip tit tog rig tug wig wig zig

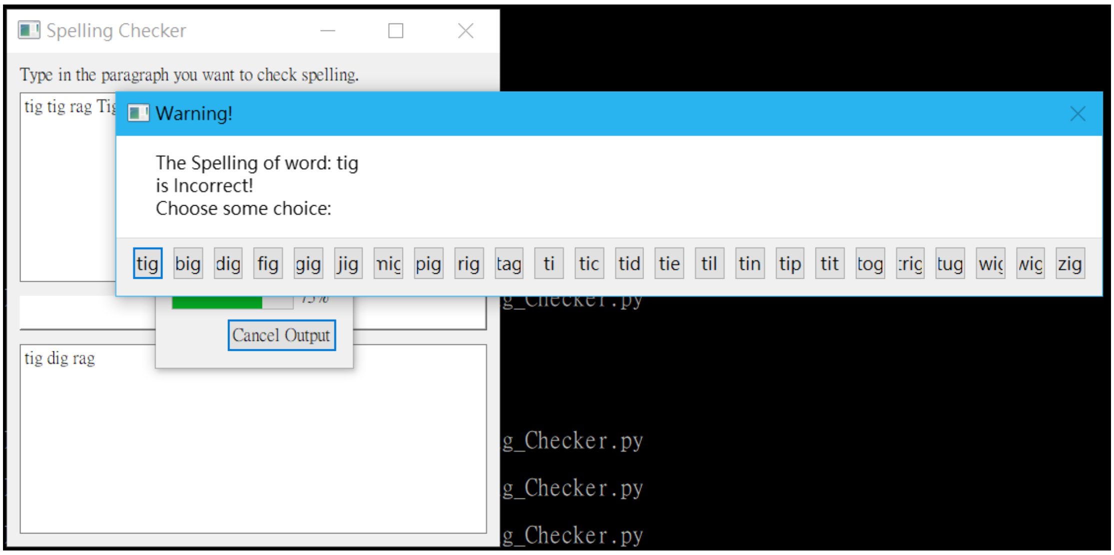
Type in the paragraph you want to check spelling.

tig tig rag Tig.

The Spelling of word: tig
is Incorrect!
Choose some choice:

```
g_Checker.py
g_Checker.py
g_Checker.py
```

In the programing part, first we initialize main window of GUI, include two textbox(for I/O), and check button. Ones the check button were press, we will start working with the sentence. The program read in the sentence, then split the words by whitespace. In the same time, for each word we would record whether it contains Uppercase(here we only consider the situation of the uppercase by title), and also record if we contain other characters, here we try . , ? ! " these five character, and only consider the case where they were in the end of the word, and would appear only one at a time.


Spelling Checker
Warning!
tig big dig fig gig jig nig pig rig tag ti tic tid tie til tin tip tit tog rig tug wig wig zig
75%
Cancel Output

Type in the paragraph you want to check spelling.

tig tig rag Tig

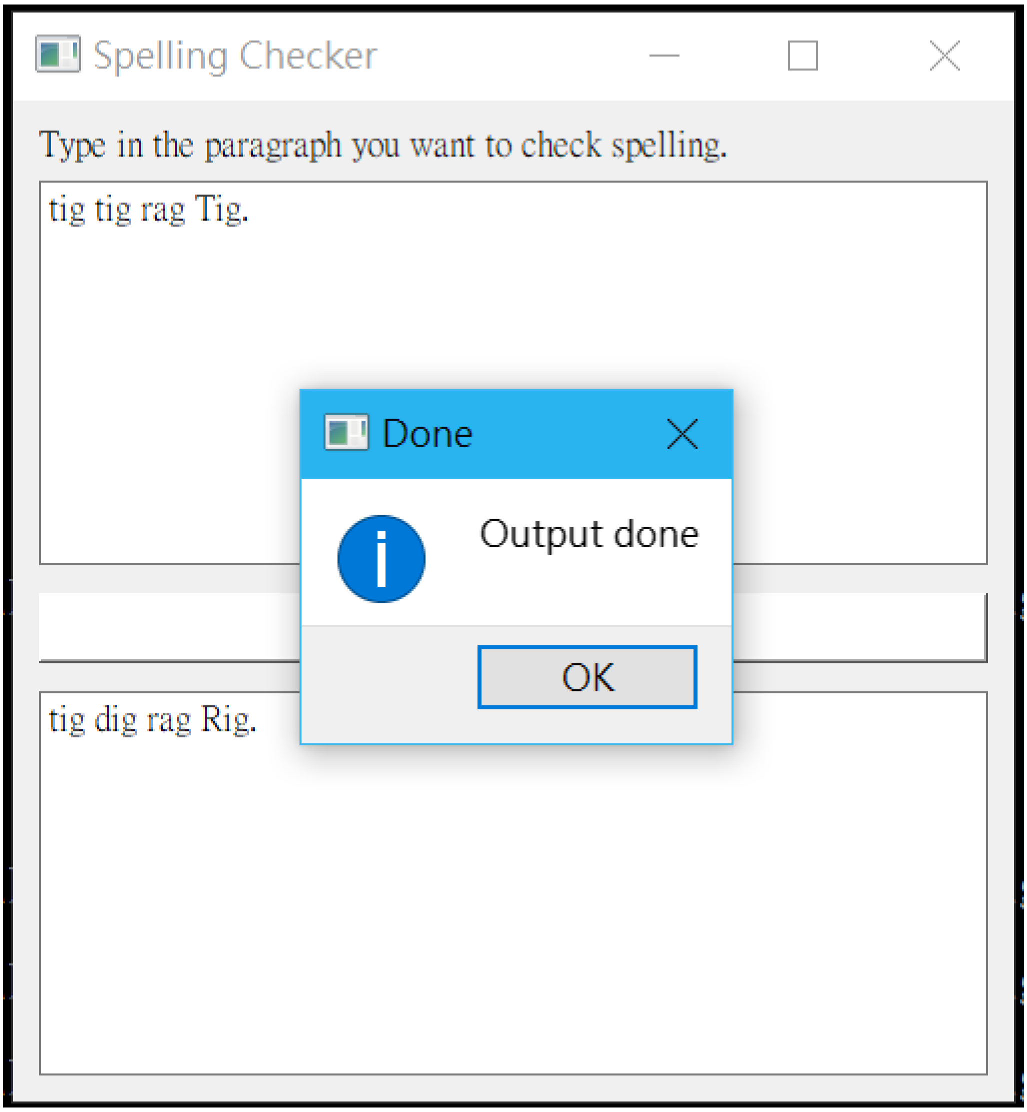
The Spelling of word: tig
is Incorrect!
Choose some choice:

tig dig rag

```
g_Checker.py
g_Checker.py
g_Checker.py
g_Checker.py
```

Then for each word we would strip the element we had mention above, check whether this word contain in the dictionary by the same way we have done in spelling checker, but popup a message box when we want user to choose the word they want to use.

After that, we restore the chosen character with original contain element, such as upper case, or the other character, and then output as plaintext to the output text box. During these process, process bar running as well, to show you the process of the percentage of the word we had finish checking. After all the words were checked, the program will pop up a message box says “Output done.”



Type in the paragraph you want to check spelling.

tig tig rag Tig.

Output done

tig dig rag Rig.

For this program, this is just some simple practice of paragraph spelling check, it may not consider much of the situation, but I surely try hard to polish this one.

Summary:

In this assignment I have learn something new about python and pyqt, include application of some skill and function I had learned before, how to better deal with popup screen, some GUI techniques; learn better about the Levenshtein distance and implement backtrack method.

In this practice, I had mix much of the method to build a real thing. Also, I had try to finish some goal that I had made for myself. I feel really excited when I finish the workable prototype, which I had barely thought I could finish at the beginning.

Although this assignment is much more easy than last one, I also had learned a lot during the process of debugging and finding useable functions which I had use or not use before. If I could keep this passion in the future, I think there would be a promising future for my programming skill.