Unit 1 Assignment

**ANLY:520-51 (Fall 2016)**

Dean D’souza

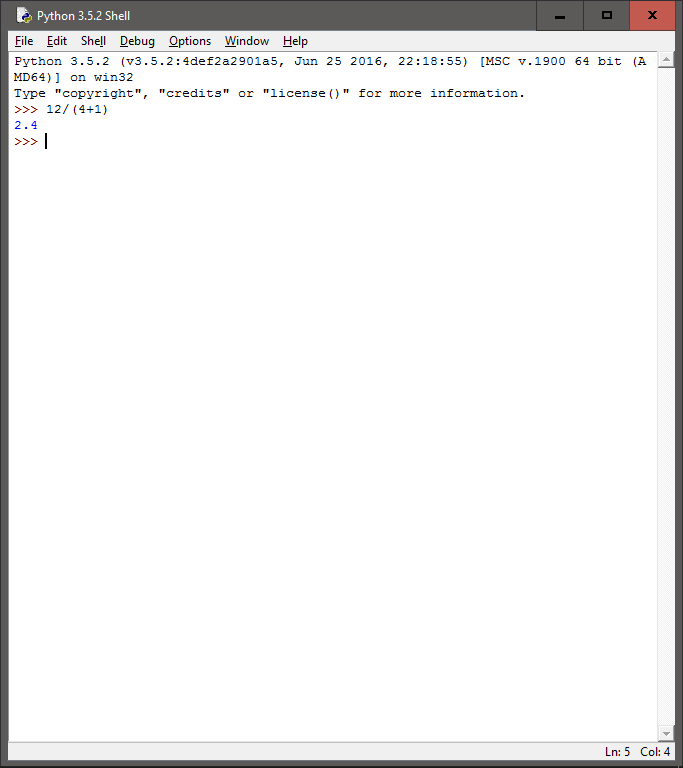
H.U. ID: 168424

# **Solutions:**

## Required 5:

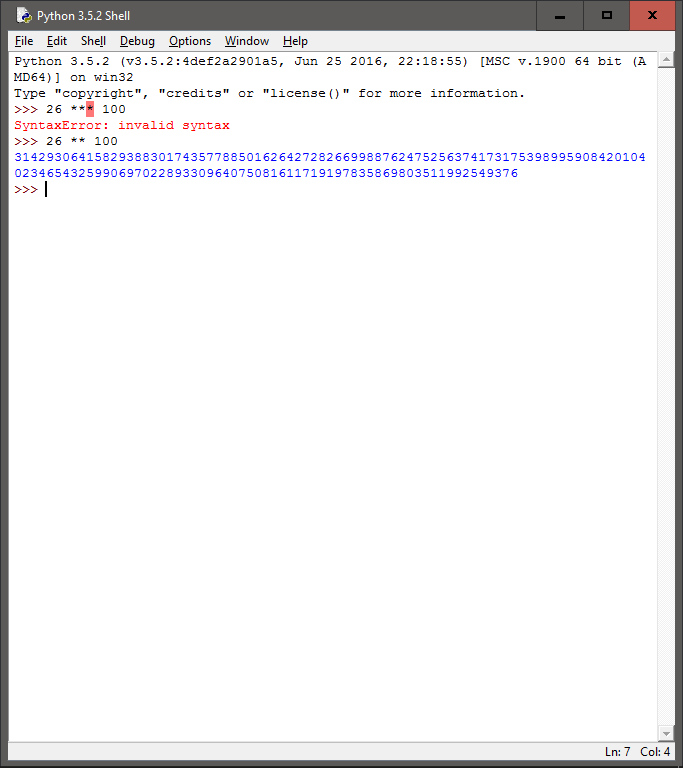
1. **Try using the Python interpreter as a calculator, and typing expressions like 12 / (4 + 1).**

The following screenshot gives the output for the above calculation:



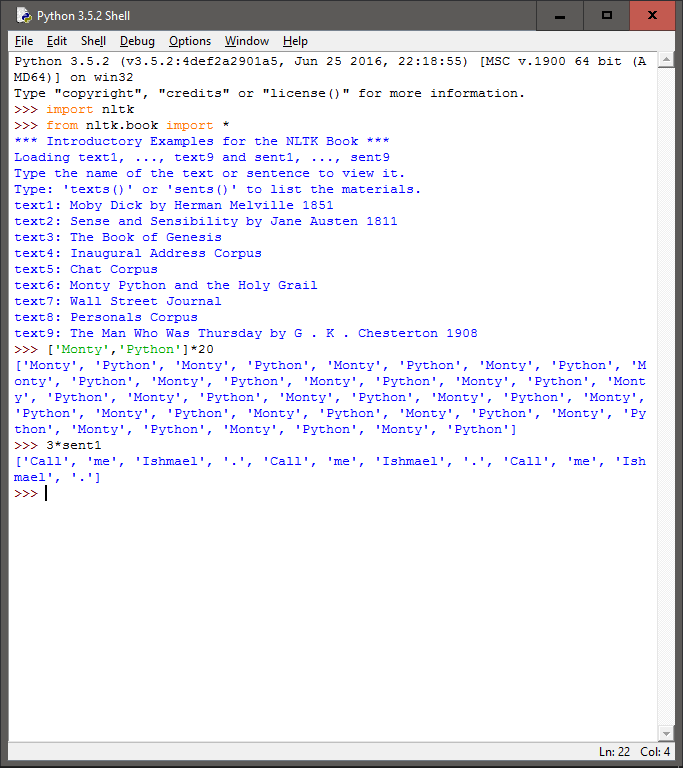
1. **Given an alphabet of 26 letters, there are 26 to the power 10, or 26 \*\* 10, ten-letter strings we can form. That works out to 141167095653376. How many hundred-letter strings are possible?**

We calculate the number of strings we can form as follows:



1. **The Python multiplication operation can be applied to lists. What happens when you type ['Monty', 'Python'] \* 20, or 3 \* sent1?**

The output for the above commands are as follows:



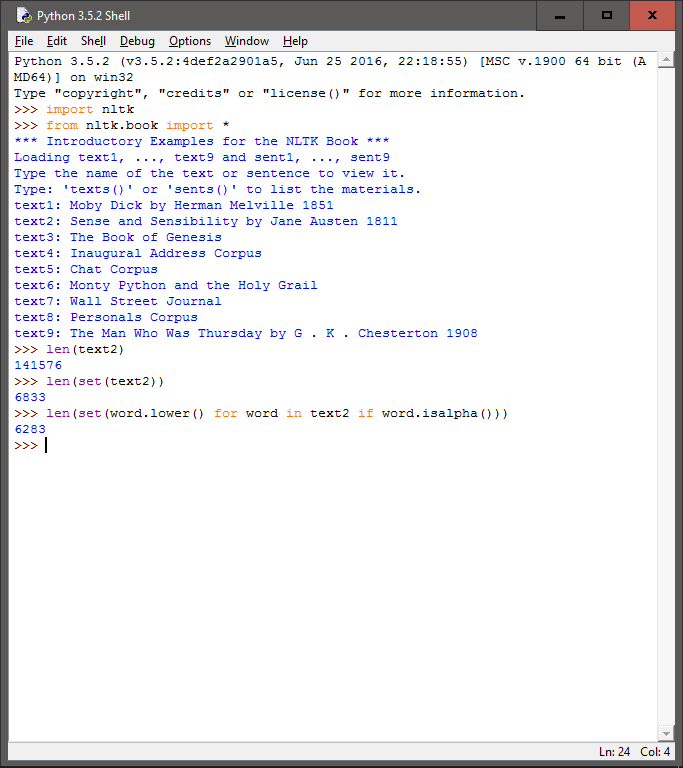
1. **Review 1 on computing with language. How many words are there in text2? How many distinct words are there?**

We calculate the required counts as follow:

Total number of words: 141576

Total number of distinct words: 6833

Total number of distinct words which are purely alphabetical: 6283

****

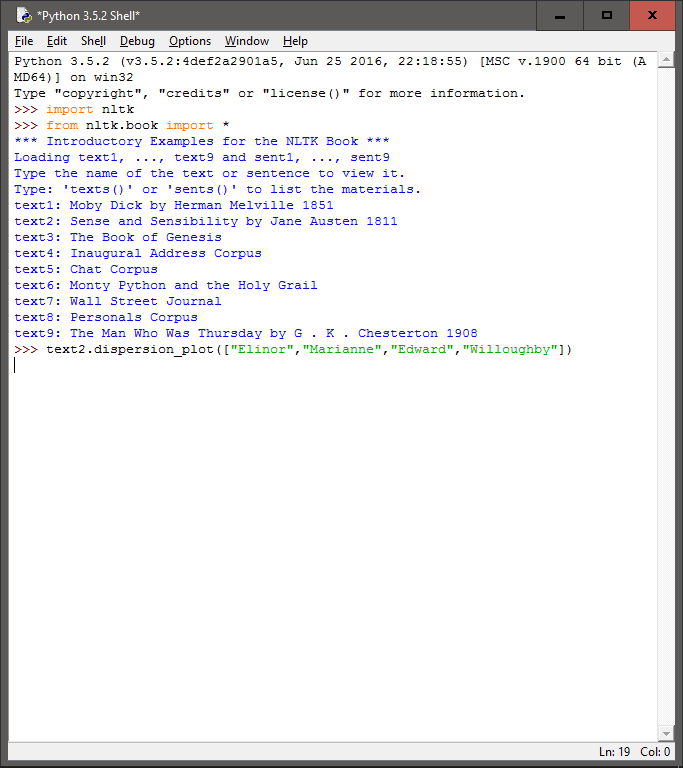
1. **Compare the lexical diversity scores for humor and romance fiction in 1.1. Which genre is more lexically diverse?**

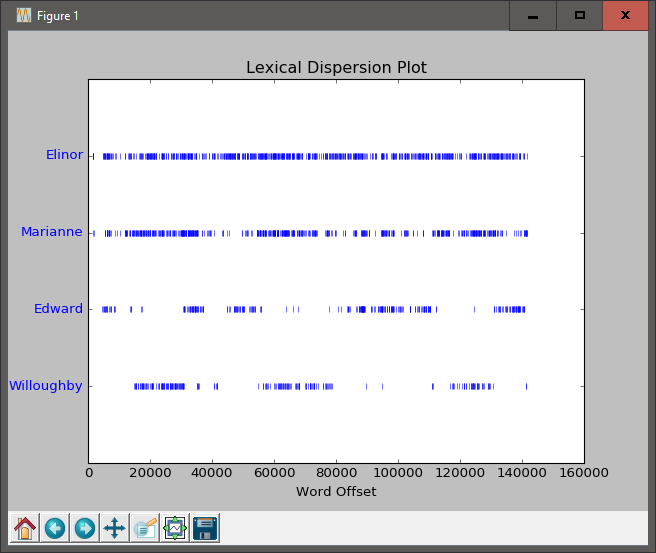
Based on the values for lexical diversity present in the table (table 1.1), we can conclude that the humor genre has more lexical diversity as compared to romance fiction. However, in terms of the number of types and the number of tokens, the romance fiction genre has significantly more than the humor genre.

## Extra Questions:

1. **Produce a dispersion plot of the four main protagonists in Sense and Sensibility: Elinor, Marianne, Edward, and Willoughby. What can you observe about the different roles played by the males and females in this novel? Can you identify the couples?**

The dispersion plot with the command is as follows:

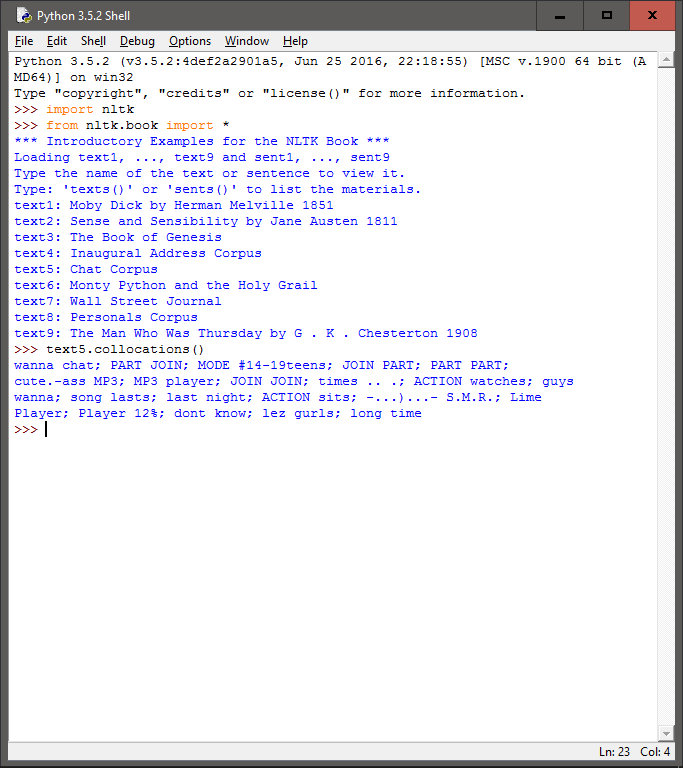




Based on the dispersion plot above, we can see that Elinor is the main character of this literary work and Marianne has more of a side/supporting role. If we were to compare the references to the characters throughout the text with the help of the dispersion plot, we can guess that the first couple is Eleanor and Edward and the second couple should be Marianne and Willoughby.

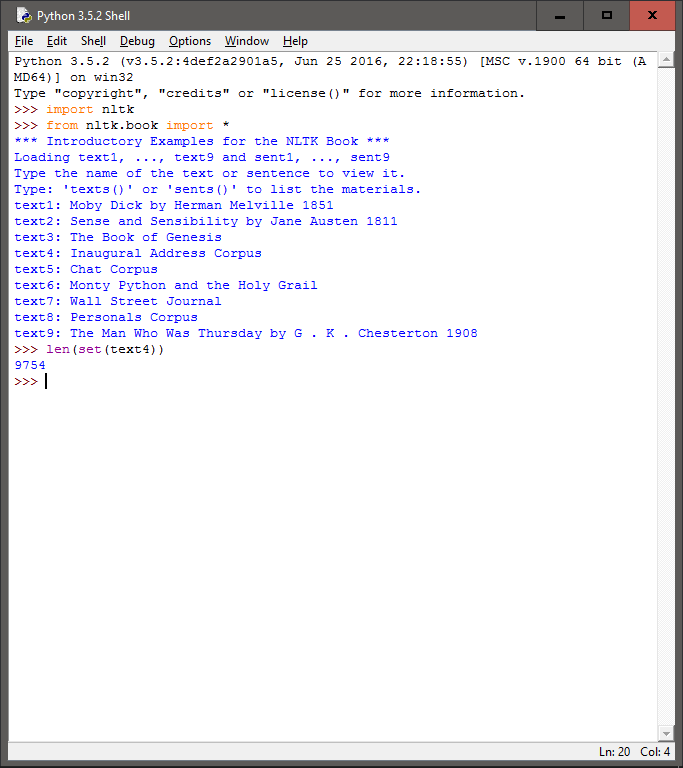
1. **Find the collocations in text5.**

We can find the collocations as follows:



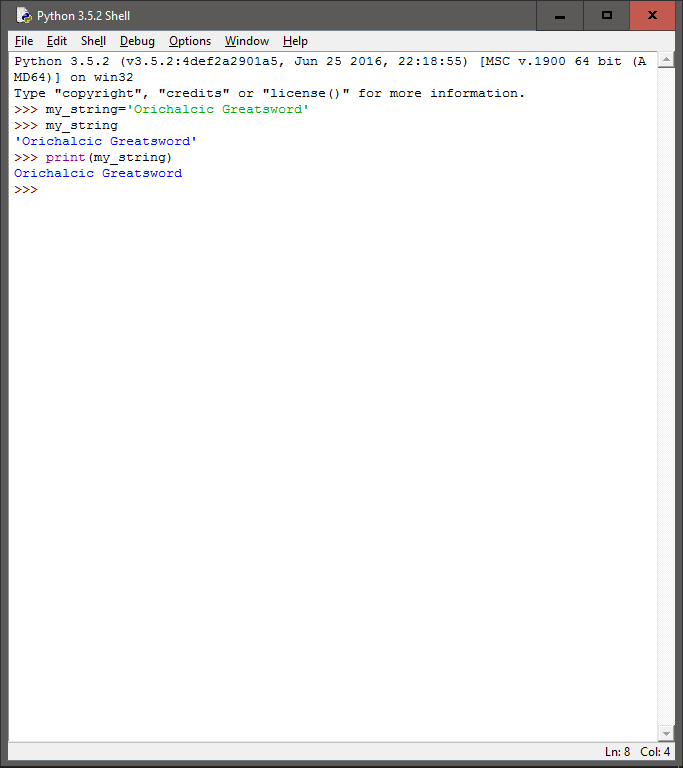
1. **Consider the following Python expression: len(set(text4)). State the purpose of this expression. Describe the two steps involved in performing this computation.**

The expression is used to calculate the total number of unique types in the text:text4. The two steps that takes place can be described through the functions, where set() function is first used to get the unique types in the text4 (based on the properties of sets), and then len() function is used to essentially count the number of such types. We can thus obtain the result as seen in the following screenshot:



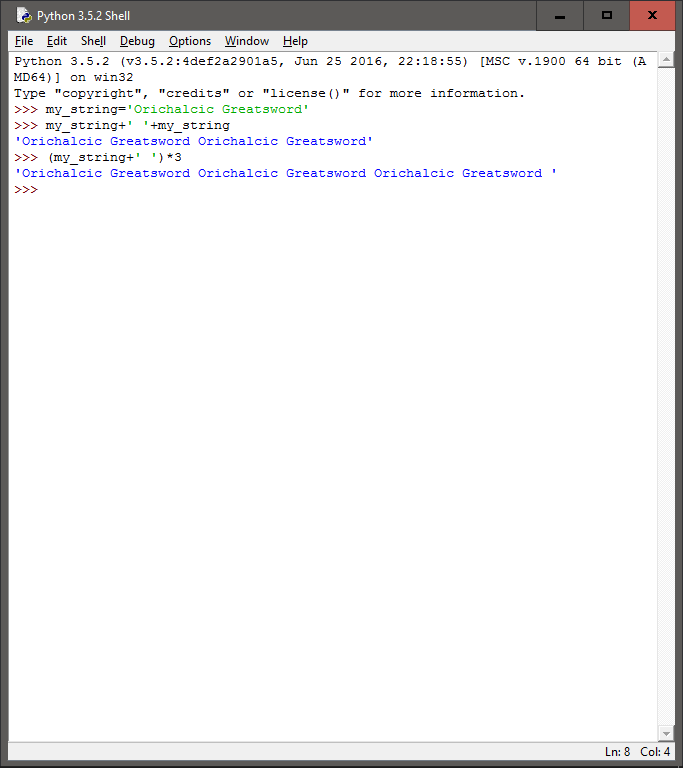
1. **Review 2 on lists and strings.**
2. **Define a string and assign it to a variable, e.g., my\_string = 'My String' (but put something more interesting in the string). Print the contents of this variable in two ways, first by simply typing the variable name and pressing enter, then by using the print statement.**

We perform the required commands as follows:



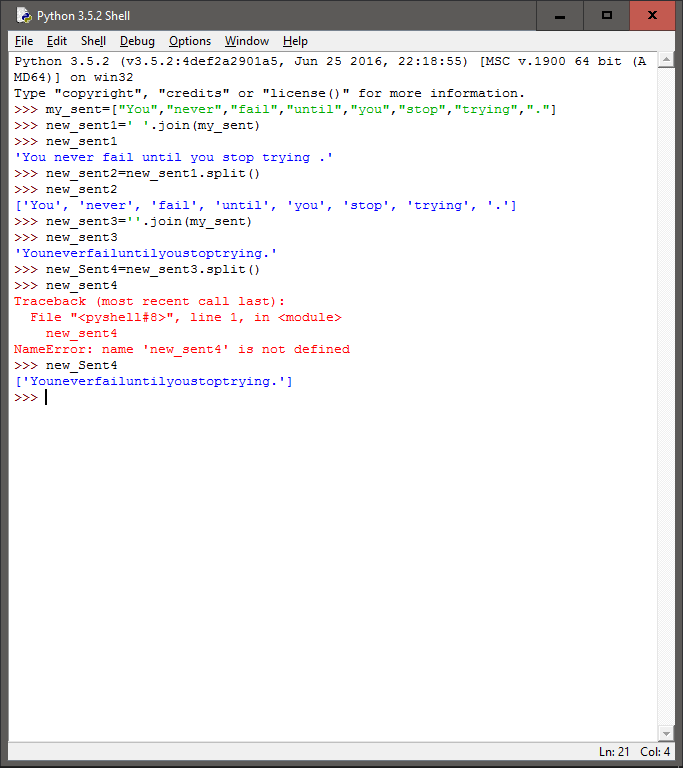
1. **Try adding the string to itself using my\_string + my\_string, or multiplying it by a number, e.g., my\_string \* 3. Notice that the strings are joined together without any spaces. How could you fix this?**

We can add spaces as follows:



1. **Define a variable my\_sent to be a list of words, using the syntax my\_sent = ["My", "sent"] (but with your own words, or a favorite saying).**
2. **Use ' '.join(my\_sent) to convert this into a string**
3. **Use split() to split the string back into the list form you had to start with.**

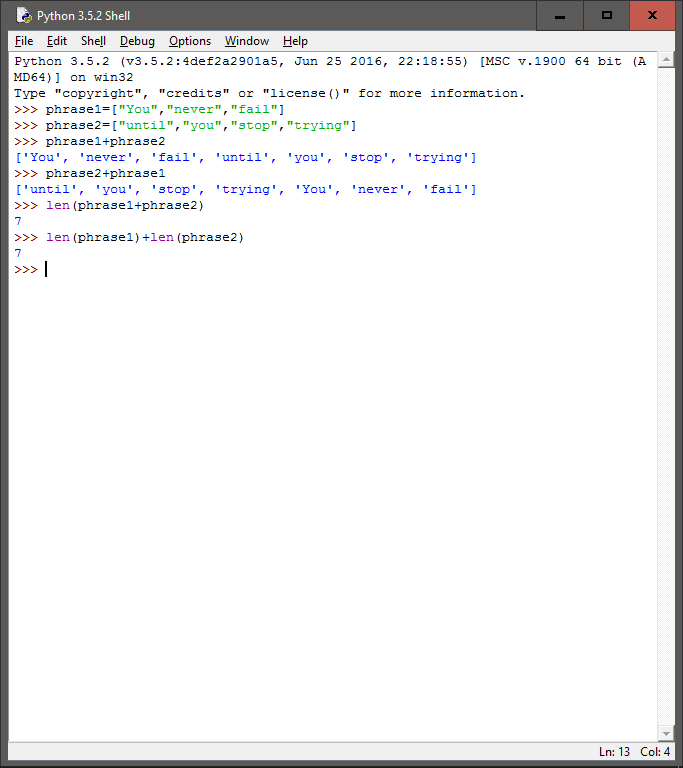
We perform the commands as follows:



We can notice that as long as the space is appropriately provided for the join() function, the split function will appropriately split the string formed.

1. **Define several variables containing lists of words, e.g., phrase1, phrase2, and so on. Join them together in various combinations (using the plus operator) to form whole sentences. What is the relationship between len(phrase1 + phrase2) and len(phrase1) + len(phrase2)?**

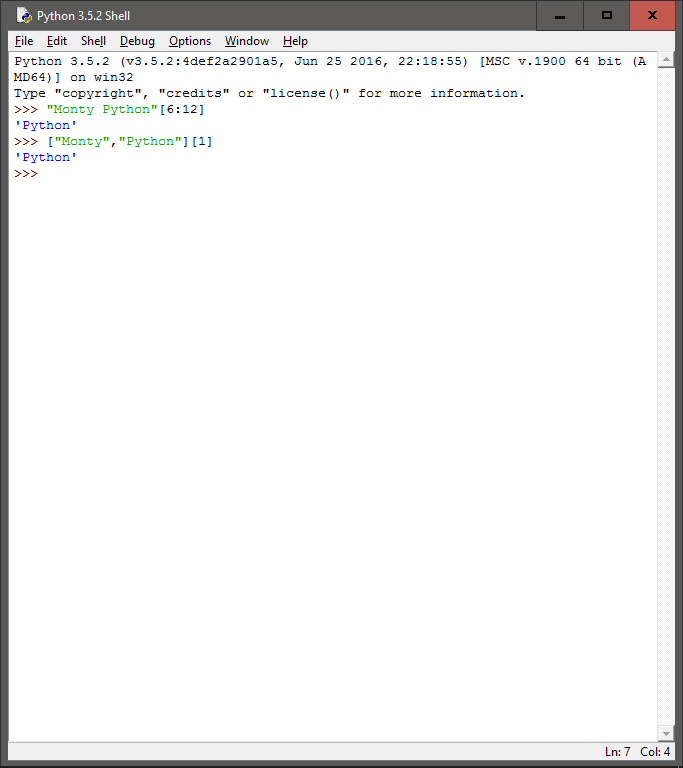
We perform the required commands as follows:



In both cases we are calculating the total number of types in the phrases. However, in the case of the first command, we first combine the two phrase and then count the new list formed while in the second command, we count the types in each phrase and the totals.

1. **Consider the following two expressions, which have the same value. Which one will typically be more relevant in NLP? Why?**
2. **"Monty Python"[6:12]**
3. **["Monty", "Python"][1]**

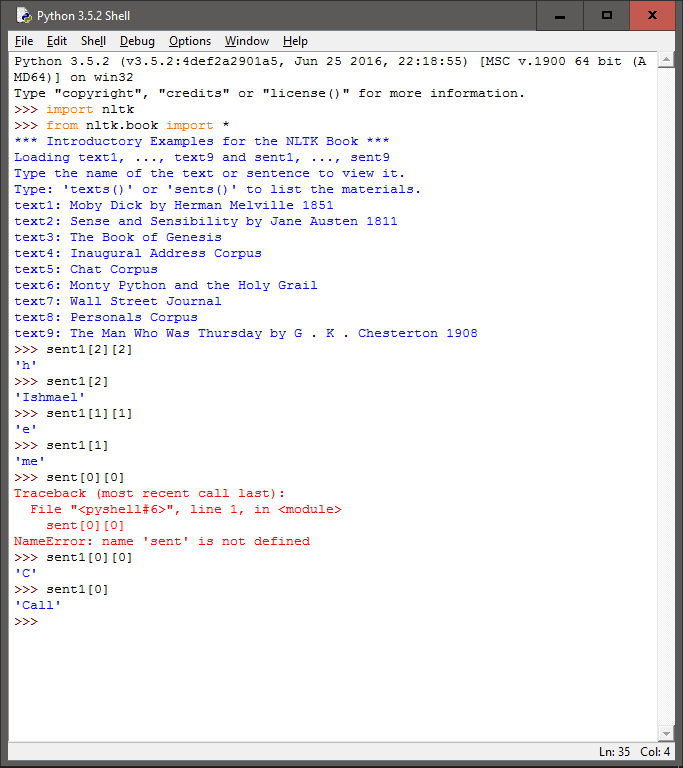
We perform the commands as follows:



According to me, the second command would be more relevant as even if we get a single string for the entire sentence or paragraph, we can split it into a list of types using the split() function.

1. **We have seen how to represent a sentence as a list of words, where each word is a sequence of characters. What does sent1[2][2] do? Why? Experiment with other index values.**

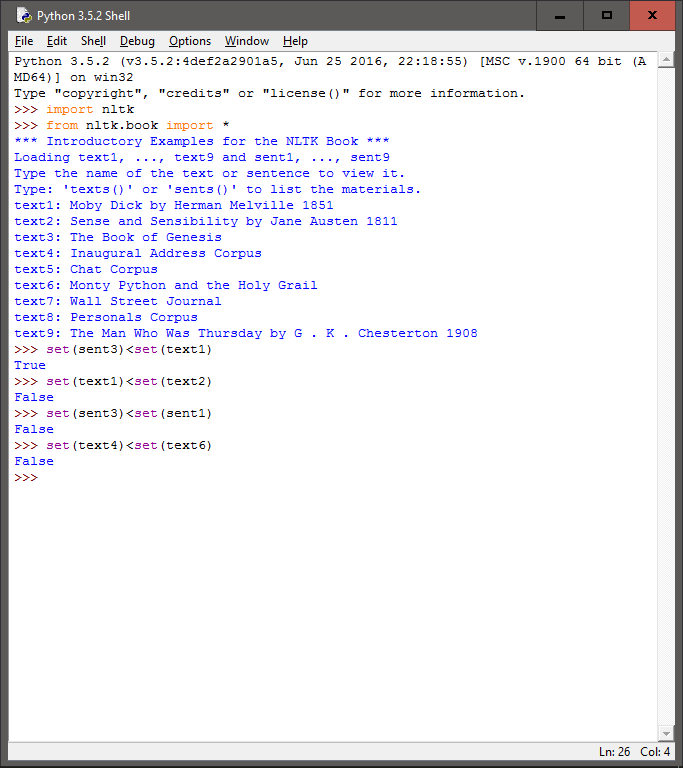
We perform the commands as follows:



As we can see from the above screenshot, the first index value refers to the index of the word in the list of words. The second index on the other hand refers to the index of the individual letters in the selected word.

1. **(Q 29 in nltk book) We have been using sets to store vocabularies. Try the following Python expression: set(sent3) < set(text1). Experiment with this using different arguments to set(). What does it do? Can you think of a practical application for this?**

We perform the command as follows:



The above form of commands essentially compares the number of types in each text or sentence and returns true or false based on if the condition is satisfied. The main purpose of such commands would be to compare different sets of text in order to compare how many types these texts consist of. We could also use such commands to verify if we have properly created subsets of larger texts.