Cattrina DeGruchy Mott
Department of Computer Science & Media Technology
Linnaeus University
cm222pr@student.lnu.se

Introduction

The first assignment for this course was to perform some simple text processing techniques on a given text. The solutions were to be written using Python 3.

Tasks

Task 1

The first task was to count the number of sentences in the given text and output this information on the screen. I decided to approach this using a simple split on full stop (".") to create an array of strings, each containing a sentence. I then used len to calculate the length of this array, giving me an integer. The .format method allowed me to use string interpolation to display this information.

```
splitBySentence = textA.split(".")
numberSentences = len(splitBySentence)
print ("There are {sentences} sentences in this piece of
text".format(sentences=numberSentences))
```

Task 2

The second task was to perform the same process as above but this time calculating the number of words.

This time I used a split on spaces to generate my array.

```
splitByWord = textA.split( )
numberSentences = len(splitByWord)
print ("There are {words} words in this piece of
text".format(words=numberWords))
```

Task 3

Task 3 was more complicated and forced me to rethink my approach to the first two tasks. It required me to order all the sentences in the text by length and output the results as:
'The longest sentence in this text contains "X" words"; The second longest sentence in this text contains "Y" words, ..., and the shortest sentence in this text contains "Z" words'

I knew I could use string interpolation to display the values and wanted to do the same for the adjectives (longest, shortest etc). My original attempt at this involved sorting the list and then iterating through the list by index. When i (index) was 0 or had reached the maximum length of the list it would display "longest" or "shortest" respectively. This was overly complicated and involved several errors where I conflated i with list[i] and ended up with:

```
The longest sentence has 0 words
The next longest sentence has 9 words
The next shortest sentence has 8 words
The next shortest sentence has 8 words
The next longest sentence has 6 words
The next longest sentence has 6 words
The shortest sentence has 10 words
```

Instead, I decided to use Python's max(list) and min(list) methods which was significantly easier and used less code.

```
 for i in range(len(arrayOfSentenceLengths)):
   x = arrayOfSentenceLengths[i]
   if x == max(arrayOfSentenceLengths):
      adjective = "longest"
   elif x == min(arrayOfSentenceLengths):
      adjective = "shortest"
   else:
      adjective = "next longest"
    print ("The {adjective} sentence has {xxx} words".format(adjective =
adjective, xxx=x))
```

However, now I noticed that my shortest sentence had 0 words. This meant that my approach to Task 1 was incorrect. I went back and removed empty arrays from my sentence list.

```
splitBySentence = textA.split(".")
sentenceList = filter(None,splitBySentence)
```

Now both Task 1 and Task 3 output correctly.

Task 4

The final task was to find the 5 longest words in the text and output them in a sentence:
‘Five longest words in this text ordered alphabetically are: “word1”, “word2”, “word3”, “word4”, “word5”’).

On my first test, I realised that some words included the punctuation that immediately followed. I went back to Task 1 and used translate() to remove these and then used strip() to remove the resulting whitespace from the sentences. The output was better but contained duplicates, I converted into a set and back to a list to only output unique values. However, I hadn’t accounted for capitalisation so I went back to Task 2 and converted all of the strings to lowercase.
Now I had a list of unique strings which I sorted alphabetically. The alphabetically sort() function in Python is stable so this allowed me to sort my words by length and output them in a way that is first ordered by length and within that, alphabetically.

I decided this would be a good opportunity to make use of a function, as it would allow me to change the input text and the number of words displayed.

```
def longest_words(listOfWords, integer):
  listOfWords.sort(key=len, reverse=True)
  listLongestWords =  ", ".join(listOfWords[:integer])
  print ("The {number} longest unique words in this list are
{list}.".format(number = integer, list = listLongestWords))
```

Task 5

I had already used a function in Task 4 but decided my code would be more readable if I put Task 3 into its own function.

Conclusion

The tasks in this assignment became easier when I began thinking of Python in a more mathematical way (rather than Javascript which I think of in terms of objects). Python includes a lot of built-in methods for list comprehension and I feel I could have made better use of those to keep my code concise. There are also some areas where my code is not DRY (such as removing punctuation) and I hope to go back and clean this up when I have a better grasp of the language.