Russell Stirling
250 757 946 - rstirli2

Assignment 3

Please Note: Make sure build is set to x86 in VS. Also In order to minimize submission folder size to be under 10MB I had to remove all text files with sample data from directory. Please make sure files are in same directory as executing file. You may also have to update the project properties to make the projects point to and reference nGrams project correctly.

Question 1)

Part A:

Code can be found in file **P1.cpp**

Part B:

For the **DostoevskyKaramazov.txt** we were able to achieve a coverage of **49.7853%** with a **k = 59**.

For the **DrSeuss.txt** we were able to achieve a coverage of **49.9655%** with a **k = 35**.

While the size of the text may influence this slightly, this difference in k values is likely due to **DostoevskyKaramazov.txt** having a much more diverse vocabulary than a **DrSeuss.txt.**

Question 2)

Part A:

Code can be found in file **P2.cpp**

Part B:

“DostoevskyPart1.txt” (as first) and “DostoevskyPart2.txt” (as second).

n	1	2	3	4	5	6
Percentage	33.1106	68.6534	87.5173	96.7292	99.2807	99.837

A value of **n = 18** gives no common nGrams.

The largest common nGram was at n=17 and is:

repulsion that s what i m afraid of that s what may be too much for me

Part C:

“Dickens.txt” (as first) and “KafkaTrial.txt” (as second).

n	1	2	3	4	5	6
Percentage	32.8802	77.4582	94.5332	99.1954	99.8854	99.9768

A value of **n = 8** gives no common nGrams.

The largest common nGrams occur at n=7 and are:

in the middle of the table and
there is no such thing as a

Part D:

"MarxEngelsManifest.txt" (as first) and "SmithWealthNations.txt" (as second).

n	1	2	3	4	5	6
Percentage	84.1851	97.468	99.5337	99.9199	99.987	99.9984

A value of **n = 7** gives no common nGrams.

The largest common nGrams occur at n=6 and are:

of nature and of reason the
is the same as that of
to keep up the rate of
in order to keep up the
of a man s own labour
from them what they have not

Part E:

In part A the two texts are written by the same author. As a result, it is much more likely for the same phrase to be reused and create larger common nGrams. In the texts in part D the percentage difference starts out much higher than any of the other two, but by the n=4 value the percentages have become approximately equal. This shows that even when we start off with very different word vocabularies, the likely hood of that many words being strung together is very low. Part C and D also have a similar size for largest nGram. This is likely because this is about the size of the largest commonly strung together phrases.

Question 3)

Part A:

Code can be found in file **P3.cpp**

Part B:

The

n	Sentence generated
1	k this to this to <END>
2	he once more hearings so apologetic <END>
3	but i lose a lot of regret <END>
4	but why do you doubt it <END>
5	in that case you won t need me or any other kind of help <END>
6	she exclaimed from time to time <END>

In n=1 we have no context so we simply see the most common words used in the text with some randomness involved. In n=2 and upwards we see words that most commonly start a sentence as the

Russell Stirling
250 757 946 - rstirli2

first word because these are what generally come after a period. In n = 2 we start to have more context and the probabilities allow us to string together words most often used in succession. This context increases as n grows and the sentences start to make some sense. At n=6 the context is large enough that we are able to form a reasonable sentence.

Part C:

Sentence generated from **MarxEngelsManifest.txt**:

thus the ten hours bill in england and the old wants satisfied by the machine and it feels that strength more <END>

This sentence ended up much longer. The sentence generated also speaks in the 3rd person, as the text it learned from did, whereas the text in Part B speaks in first person like the text it learned from.

Part D:

The text was generated from **TomSawyer.txt**:

pap used to sleep there sometimes long with the hogs but laws bless you he just lifts things when he snores <END>

Question 4)

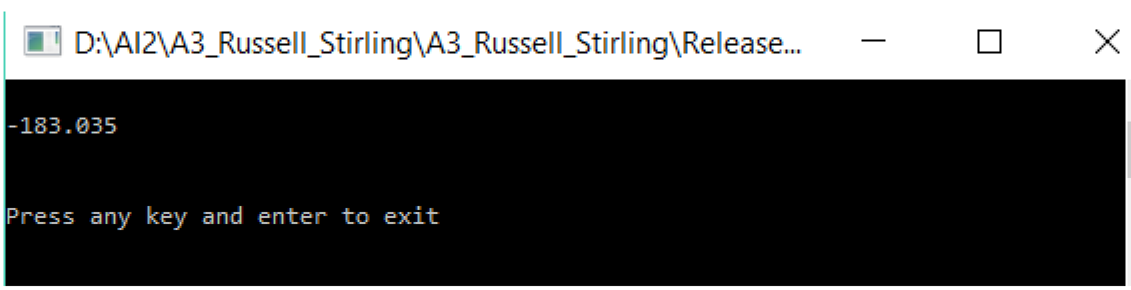
Part A:

Code can be found in file **P4.cpp**

Part B:

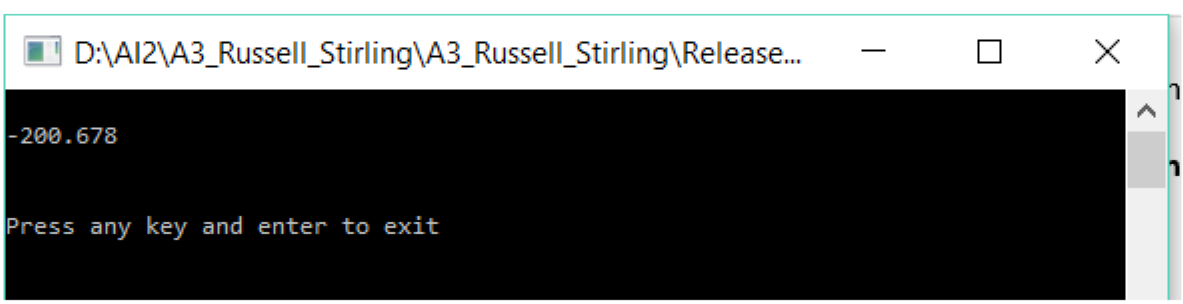
P4 KafkaTrial.txt testFile.txt 1 1

-183.035



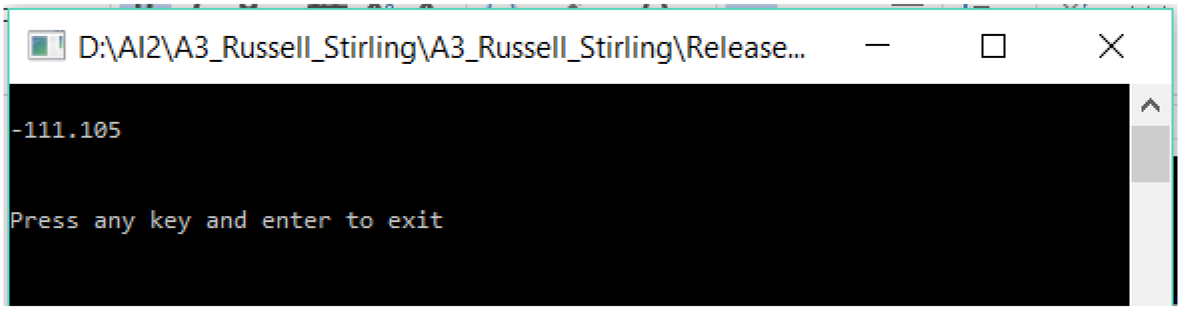
P4 KafkaTrial.txt testFile.txt 2 1

-200.678



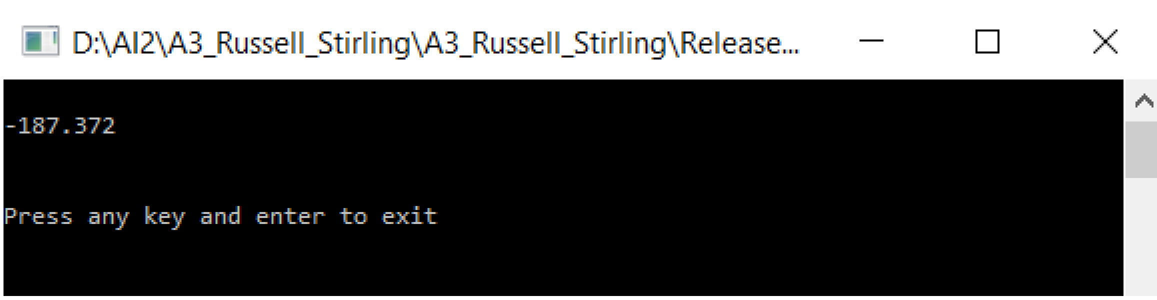
P4 KafkaTrial.txt testFile.txt 2 0.001

-111.105



P4 KafkaTrial.txt testFile.txt 3 0.001

-187.372



Question 5)

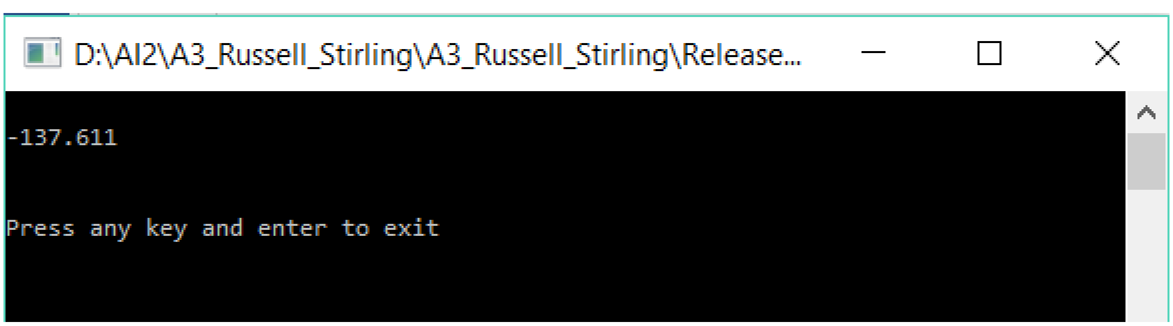
Part A:

Code in file: **P5.cpp**

Part B:

P5 KafkaTrial.txt testFile.txt 1 1

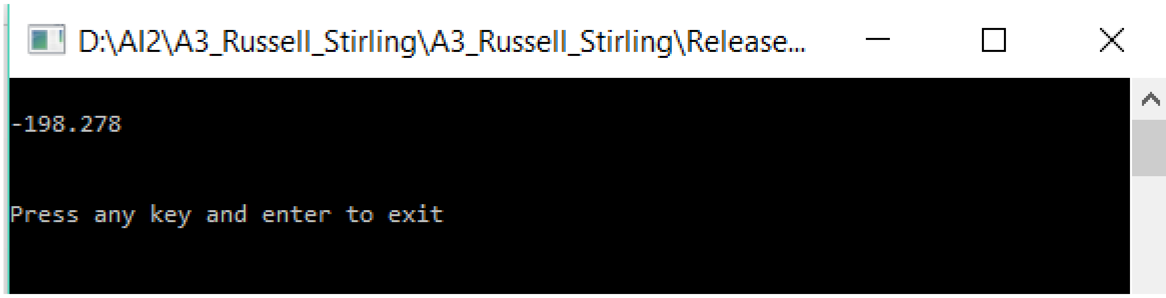
-137.611



Russell Stirling
250 757 946 - rstirli2

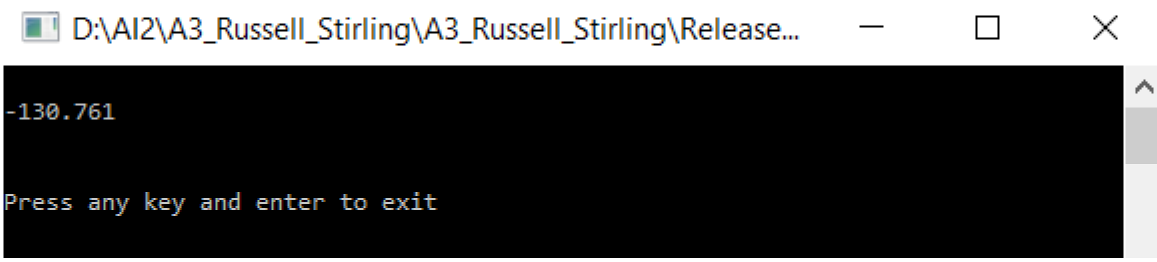
P5 KafkaTrial.txt testFile.txt 2 5

-198.278



P5 KafkaTrial.txt testFile.txt 3 5

-130.761



Russell Stirling
250 757 946 - rstirli2

Question 6)

Part A:

Code in file: **P6.cpp**

Note: Our languages are indexes in the confusion matrix are as follows
Danish = 0
English = 1
French = 2
Italian = 3
Latin = 4
Sweedish = 5

Part B:

P6 1 0 50
Error = 7.48415%

```
D:\AI2\A3_Russell_Stirling\A3_Russell_Stirling\Release...

7.48415

6540 179 1 35 5 106
231 17434 158 251 139 82
1 55 5497 213 470 2
20 143 353 10618 690 0
0 4 271 567 12119 0
576 81 0 2 50 5706


Press any key and enter to exit
```

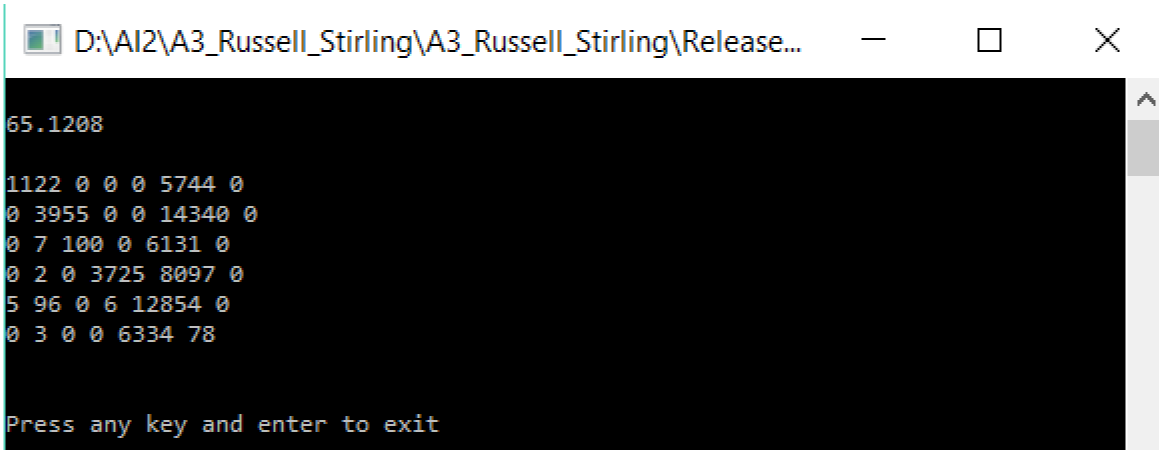
P6 2 0 50
Error = 31.2641%

```
D:\AI2\A3_Russell_Stirling\A3_Russell_Stirling\Release...

31.2641

4100 148 0 0 2589 29
101 15538 58 0 2552 46
5 110 3081 11 3010 21
82 483 76 8337 2814 32
87 2196 753 84 9804 37
127 36 0 0 4084 2168


Press any key and enter to exit
```

Russell Stirling
250 757 946 - rstirli2

P6 3 0 50

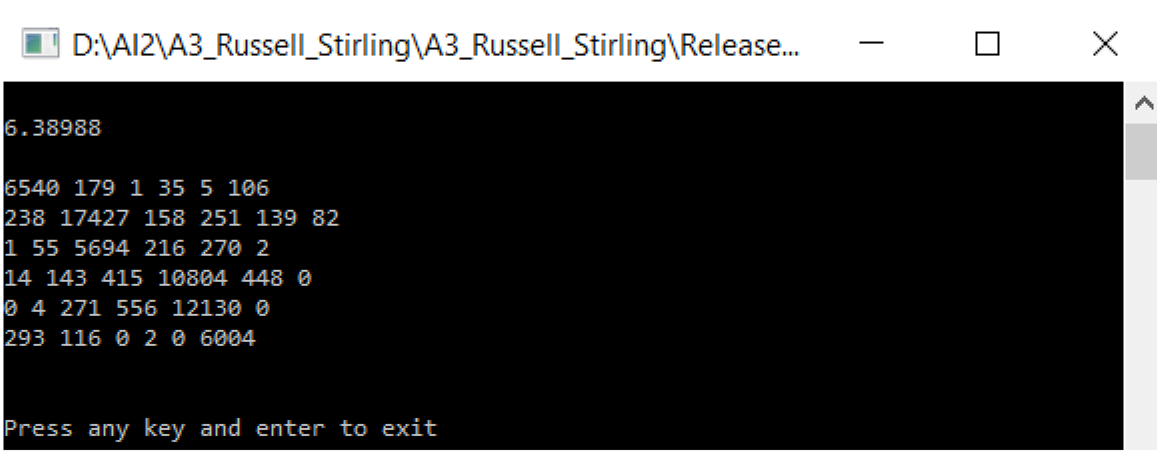
Error = 65.1208%



Part C:

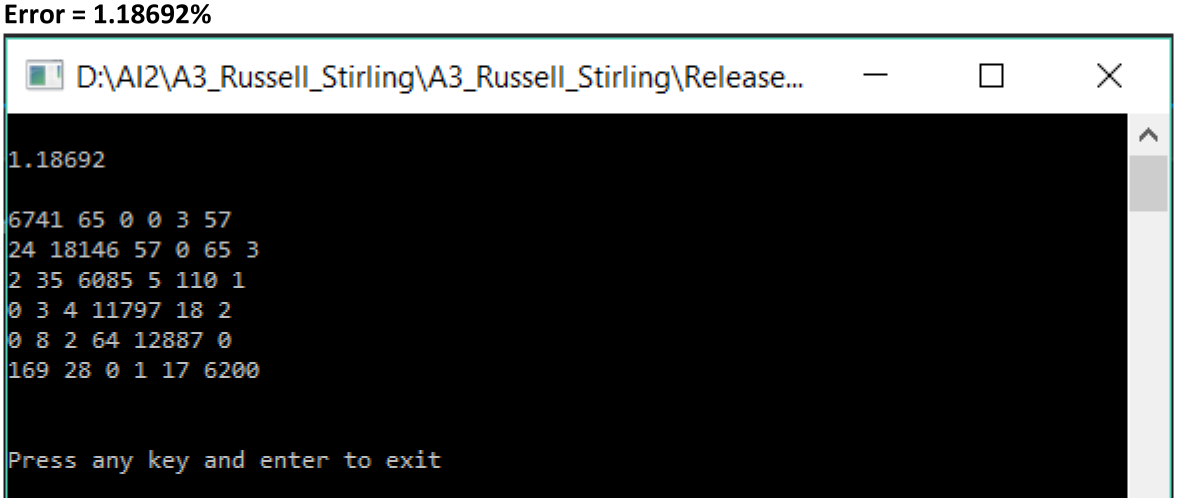
P6 1 0.05 50

Error = 6.38988%



P6 2 0.05 50

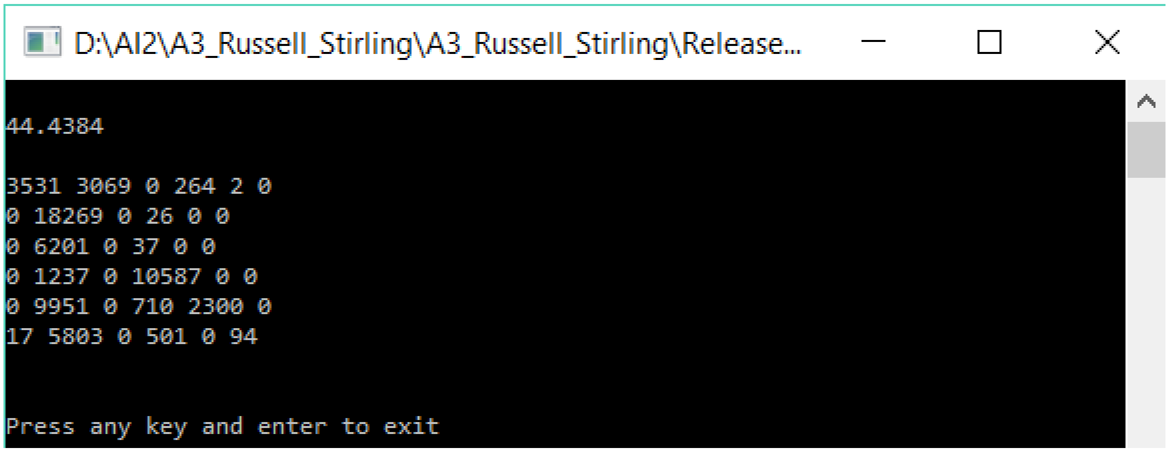
Error = 1.18692%



Russell Stirling
250 757 946 - rstirli2

P6 3 0.05 50

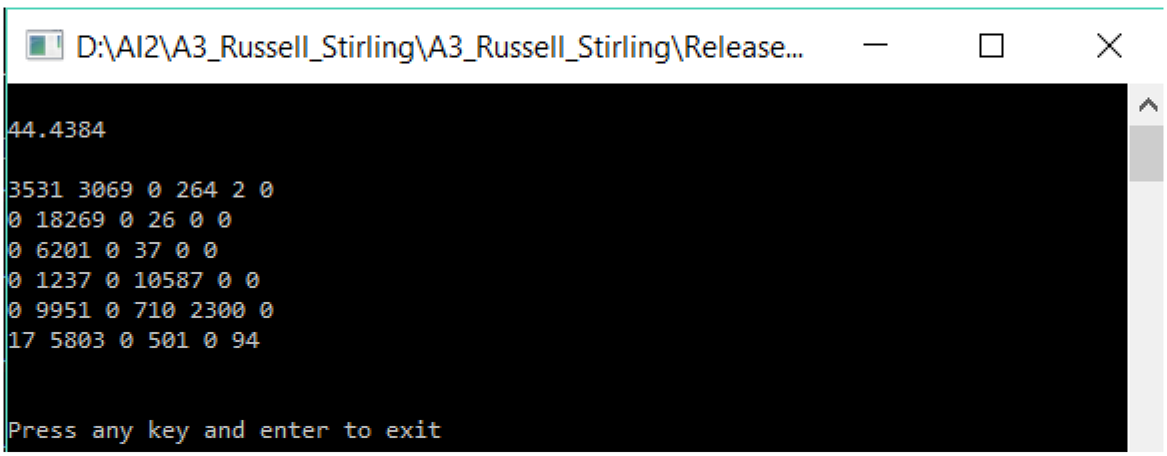
Error = 44.4384%



Part D:

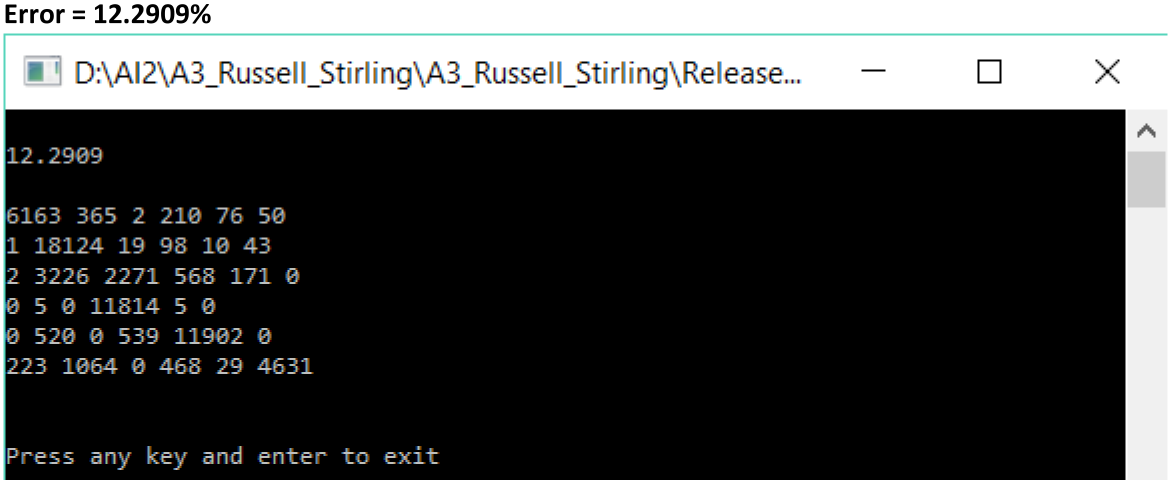
P6 3 0.05 50

Error = 44.4384%



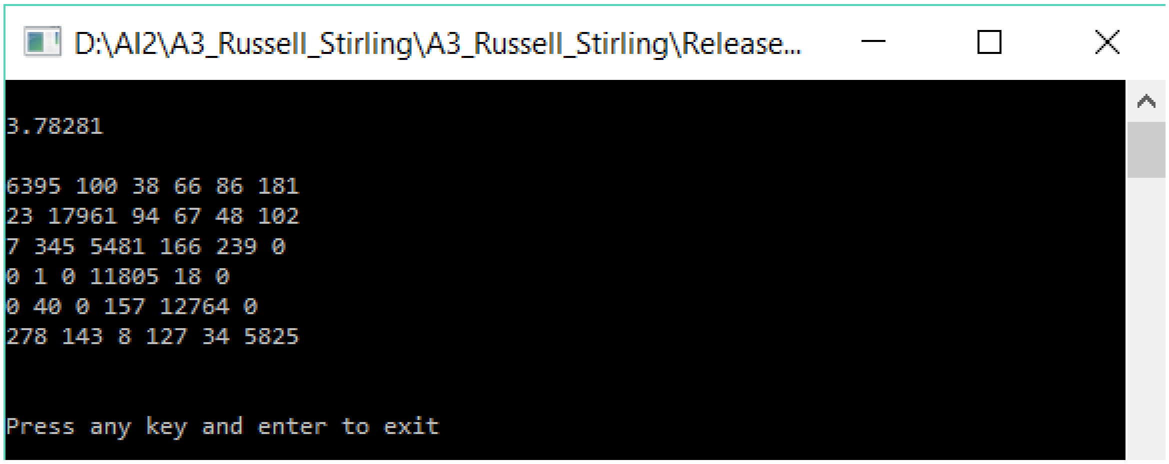
P6 3 0.005 50

Error = 12.2909%



P6 3 0.0005 50

Error = 3.78281%



Part E:

Part B clearly shows that with a 0 delta value the unigram model is most effective. This is likely because, as we get higher n values, the amount of nGrams we have not seen in the training text gets much higher and we are forced to set probability to maximum negative for these values. This makes the model fairly inaccurate.

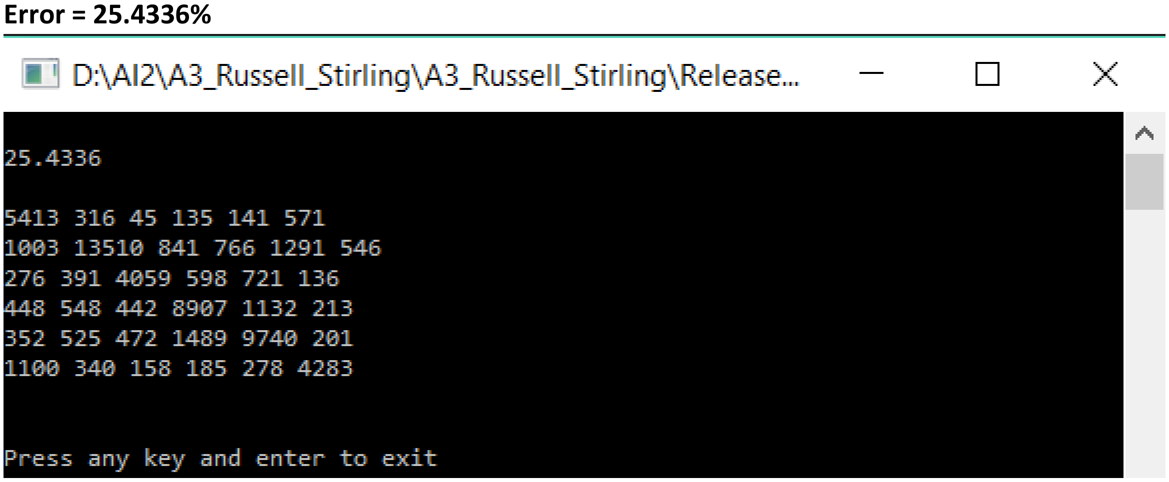
Pat C shows us that when we have a delta value of 0.05 the bigram model becomes the most accurate. This is likely because 0.05 models well with n=2 to represent the unknown values. However, once we get into the n=3 or higher, the 0.05 delta is too high and causes the weighting of unknown to be far too high.

Part D shows us that by lowering the delta to 0.005 and 0.0005 we can get a much better error result. This is because these delta values better represent the weight of our unknown nGrams and so allow us to more accurately predict results on new data.

Part F:

P6 2 0.05 10

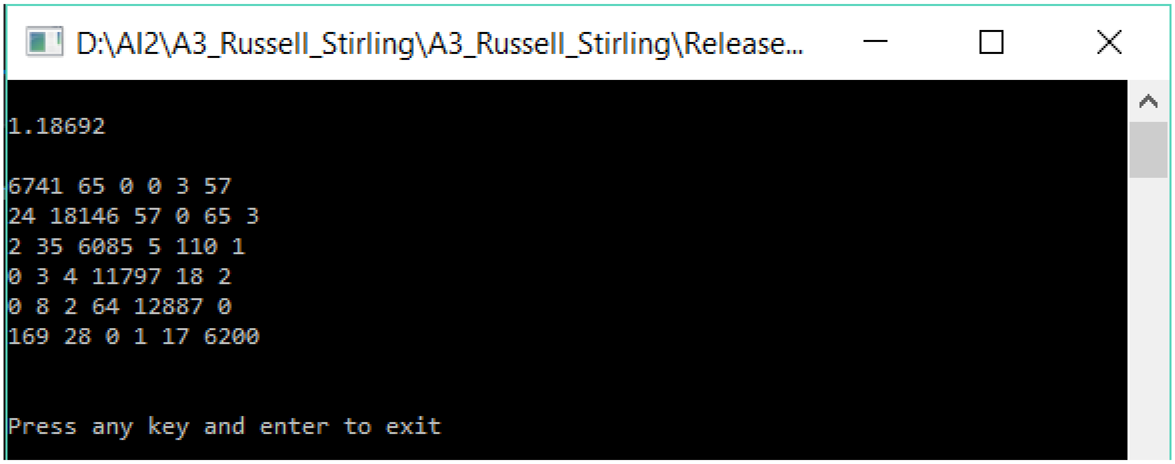
Error = 25.4336%



Russell Stirling
250 757 946 - rstirli2

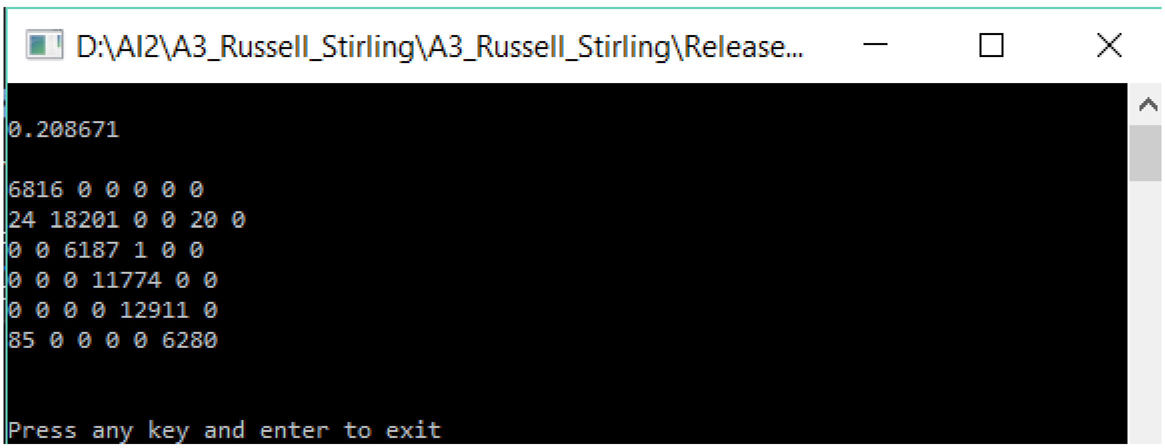
P6 2 0.05 50

Error = 1.18692%



P6 2 0.05 100

Error = 0.208671%



As can be seen, the longer the sentence length the more accurate our predictions tend to become. This is because we get a longer sample size for each classification and so we are less likely to run into duplicates from one language to another. That said this increase in sentence length increases the computation time required.

Question 7)

Part A:

Code attempt in file: **P7.cpp**

Part B:

Did not get code fully working.

Part C:

Did not get code fully working.

Part D:

Did not get code fully working.

Russell Stirling
250 757 946 - rstirli2

Question 8 EXTRA CREDIT)

Did not attempt.