Linguistics Big Assignment

CSE 6339
Introduction to Computational Linguistics

Fatema Alabdulkareem
York University

Faa@yorku.ca
Fatima@cse.yorku.ca

Contents

Introduction

In this assignment I develop different programs to cover the requirements. Each question was created in different file named after the problem name so it will be easy to distinguish them.

All the problems need to use the correlation matrix, so I develop a separate file to add books then generate the orders (first, second, third and fourth order). The file name is "GenerateOrder.aspx.vb"

In the first three questions we need to use a dictionary to compare the words generated from the monkey problem with our dictionary and find meaningful words. For this I used a word list dictionary from http://www-01.sil.org/linguistics/wordlists/english/

The program was built using .NET framework with Telerik tools. The program is published on http://fatima-001-site1.smarterasp.net/

Description of Generate Orders Program

In Generate Orders, I generate firsts, second, third and fourth orders by generating 1, 2, 3 and 4 dimensional arrays respectively.

For the third order, I generate it first using 2 dimensional array where I loop through the text and take every 2 characters and store them in my array and count the third character occurrence for each two characters, but it took long time to generate the correlation matrix. So, I decided to try it with three dimensional array and it was much faster.
The function that generates the third order in 2 dimensional array is `RadThird2D_Click()`

I completed the rest of problems using the three dimensional array method, but I displayed my two dimensional array for the third order matrix in Problem 1e.

When I use the third dimensional array for third order monkey problem I smooth the array by adding 1 to all the elements in the array, because without the smoothing generating the text will start by generating some words then it will end up with typing aaaaaaaaa because the summation of the third character occurrence will lead up to 0 and 0 is the index of "a" so it will print "a".

While in fourth order matrix I didn't use the smoothing because using it didn't give me as good word yields as in the third dimensional array and the generated text was containing all the symbols that shouldn't be occurred that frequently. So to avoid the problem of typing aaaaaaaa in the generated text I added extra condition that if the summation of the fourth character is 0 don't print anything so we will not end up with aaaaaaaaaa.

Description of Problems and Sample Output

Problem 1a

Generate straightforward monkey problem, then compare the result with the dictionary file, the program runs to type 100,000 characters.
To do this, I used a random generator function, and the output of this function was considered to be the index of the alphabet.

```
Dim validchars As String = "abcdefghijklmnopqrstuvwxyz,.;:?!()-'@""# "
Dim idx As Integer = rand.Next(0, validchars.Length)
Dim randomChar As Char = validchars(idx)
```

To compare the text with the dictionary I used `arrayFind()` function, where each word from the generated text is compared with the dictionary in a binary search for faster output. This function is used in the first three problems.

The answer for this problem is in file "Prob1a.aspx.vb"
The result is shown in Figure 1.
As it is clearly seen that the matching words are small word and most of them are one letter like "a"

Figure 1: Sample of straightforward monkey problem

Problem 1b

Generate first order monkey problem from the character distribution provided in the assignment for Act III of Hamlet, the program runs to type 100,000 characters.
To do this, I build an array that contains the character distribution for Act III of Hamlet, and then I generated a random number between 0 and the total number of occurrence for all the characters. I used the algorithm provided in Bennett Ch4 page 112 to know which key the monkey hits.

```
Dim idx As Integer = rand.Next(0, total - 1)
For j As Integer = 0 To 27
       S = S + Dist(j)
       If idx < S And flag = False Then
              randomChar = validchars(j)
              sb.Append(randomChar)
              flag = True
       End If
Next j
```

The answer for this problem is in file "Prob1b.aspx.vb"

A Sample of the output is shown in Figure 2.
As we can see the word count in the first order monkey is more than the straightforward monkey problem, also with more varieties in words, although most of the words are short words with two to three characters.
The meaningful word count was 2700 words and the percentage of the correct words to the total number of typed words is 13.68%

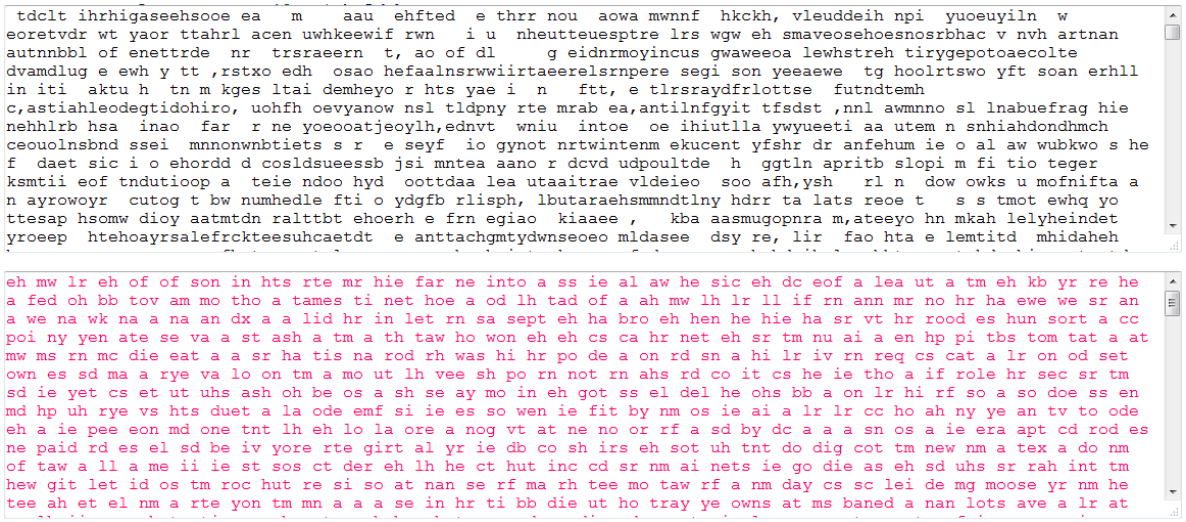

Figure 2: Sample of first order monkey from Act III of Hamlet

Problem 1c

Generate first, second and third order monkey problems, the program runs to type 100,000 characters then compares the typed characters with the dictionary.
The user can choose a book that he/she wants to generate the first, second and third order for from the drop down list that contains all the books that have a correlation matrix.
Then the user can generate the text and then compare it with the dictionary.

For first order, the same algorithm for the problem 1b was used; the function for this part is `RadFirst_G_Click()`

For second order, the algorithm from Bennett Ch4 pages 117,118 was used to generate the text from the second order correlation matrix; the function for this part is `RadSecond_G_Click()`

```
Dim idx As Integer = rand.Next(0, FO_intArray(firstCh))
For j As Integer = 0 To 39
       S = S + ResultsArray(firstCh, j)
       If idx < S And flag = False Then
              randomChar = validchars(j)
             sb.Append(randomChar)
             temp = j
             flag = True
      End If
Next j
firstCh = temp
```

At the beginning, I generated a random number between 0 and FO_intArray at the first character index. The "FO_intArray" Array is the sum of all the occurrence of the second characters given the first character. This information was stored in the system while generating the second order matrix.
Then, I loop to find the second order according to the books algorithm.

For Third order, the algorithm from Bennett Ch4 page 121 was used to generate the text from the third order correlation matrix, the function for this part is `RadThird_G_Click()`

```
For x As Integer = 0 To ResultArray.GetUpperBound(2)
       sum += ResultArray(firstCh, SecondCh, x)
Next
Dim idx As Integer = 0
idx = rand.Next(0, sum)
For j As Integer = 0 To 39
       S = S + ResultArray(firstCh, SecondCh, j)
       If idx <= S Then
              randomChar = validchars(j)
              sb.Append(randomChar)
              temp = j
              Exit For
       End If
sNext j
firstCh = SecondCh
SecondCh = temp
```

At the first loop, I sum all the occurrence of the third characters given the first and second characters, then I generated the random number between 0 and the sum I just calculate.

The last loop is where I found the third character that the monkey will type according to the books algorithm.

For fun and curiosity, I generated fourth order monkey. The program runs to type 1000,000 characters but I added a condition that if the sum is 0 which happens a lot, don't print anything. With this condition, the output text contains a few words comparing with the other orders but most of these words are meaningful words. The algorithm used for this part is the same as the one for the third order monkey but with four dimensional array and with the condition that if the sum is zero don't print anything. The function for this part is `RadFourth_G_Click()`

The answer for this problem with all the functions described earlier for first, second, third and fourth order text generator are in file "Prob1c.aspx.vb"

Results between first, second and third order are shown in Table 1.
Correct words according to the dictionary are counted and mentioned as "Word Count" also the percentage of the how many word count (meaningful words) compared to the original generated words are computed as "Pct".
Note: Any book can be added and first, second, third and fourth order monkey can be generated for it.

It's clearly seen that the number of correct words and percentage increase significantly with the number of order.
In order 1 the correct words are between 13-15% the highest percentage was for Dickens - A Tale of Two Cities, the percentage was 15.92% while the lowest percentage was 13.85% for Kafka - The Trial.
In order 2 the correct words are between 24-28% the highest percentage was for Haggard – Child of Storm, the percentage was 28.26% while the lowest percentage was 24.75% for Carroll - Through the looking glass. It's also worth mentioning that Twain - Adventures of Huckleberry Finn has the second highest percentage which is 28.00% and is close to the 28.26% of the Child of Storm.

In order 3 correct words are between 48-56% the highest percentage was for Twain - Adventures of Huckleberry Finn which was the second highest percentage in the second order, the percentage was 56.87% while the lowest percentage was 48.55% for Irving - Legend of sleepy hollow.

Also, we can notice that books from the same author have similar percentage among all three orders, as well as books written by Bronte sisters which have almost the same percentage in order one and three but differ slightly in order two.

A sample of Dickens - A Tale of Two Cities was generated for all orders. Figure 3 shows first order sample where most of the words are one to two letter words with few three letter words. Figure 4 shows second order sample, where we can see that the length of words increases to reach 6 letters per word. Figure 5 shows third order sample, where more long and meaningful words are generated, in this sample we can see the word "daughter" which contains of 8 letters. Figure 6, shown fourth order sample, where we can see that most of the word are meaningful words and we can see as well the name of the characters start to appear like "Sydney". The percentage of correct word in this example was 80%.

These examples illustrate that word count and percentage of meaningful words increases dramatically with the order of the frequency matrix used for the typewriters.

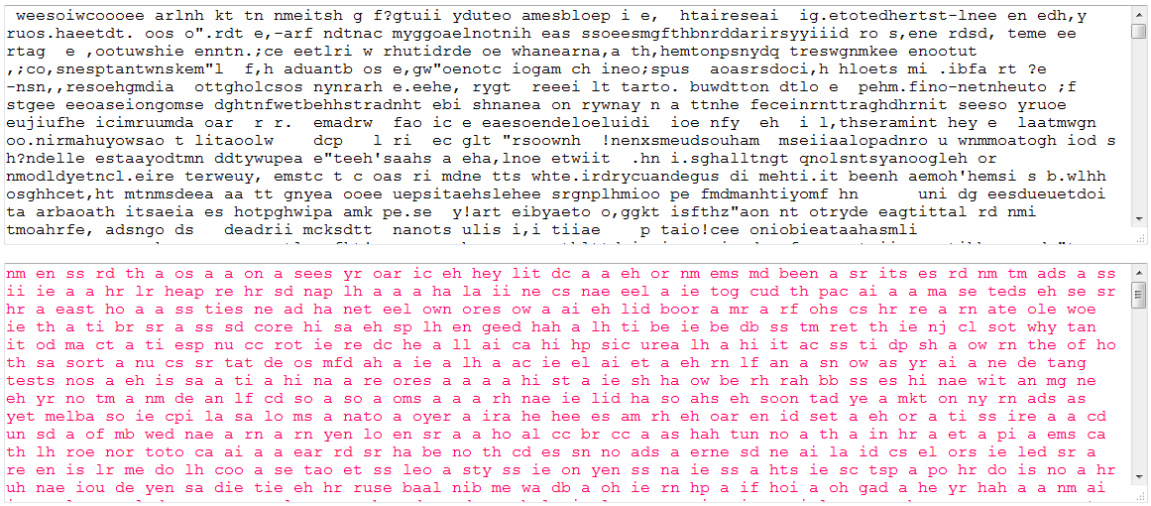

Figure 3: Dickens - A Tale of Two Cities, first order

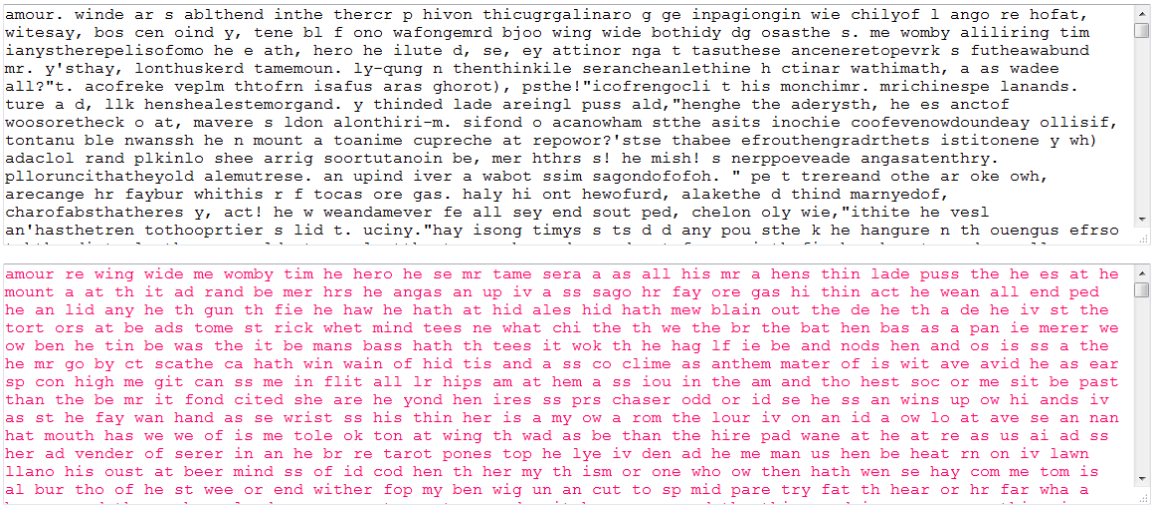

Figure 4: Dickens - A Tale of Two Cities, Second order

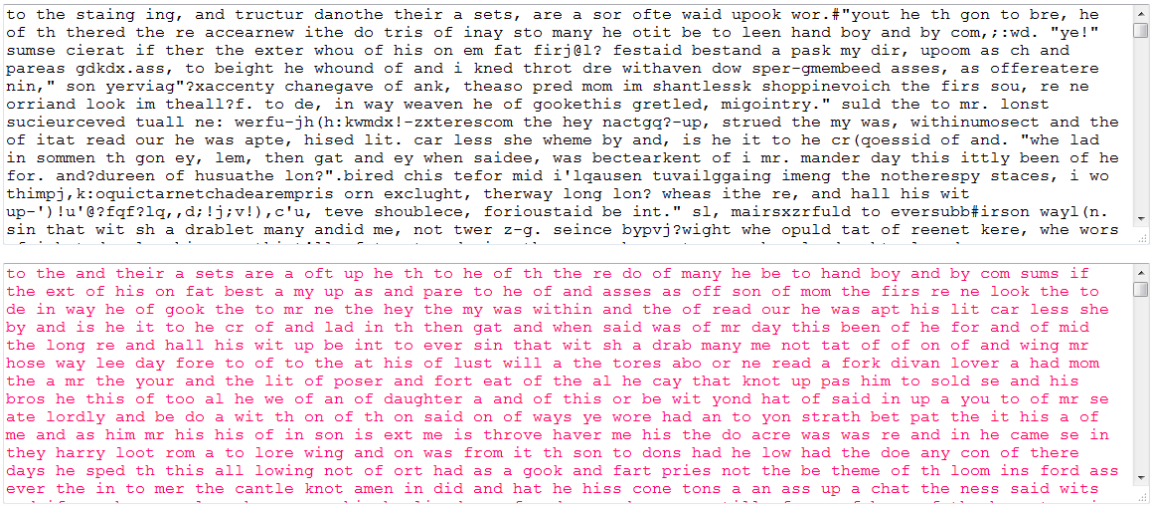

Figure 5: Dickens - A Tale of Two Cities, third order

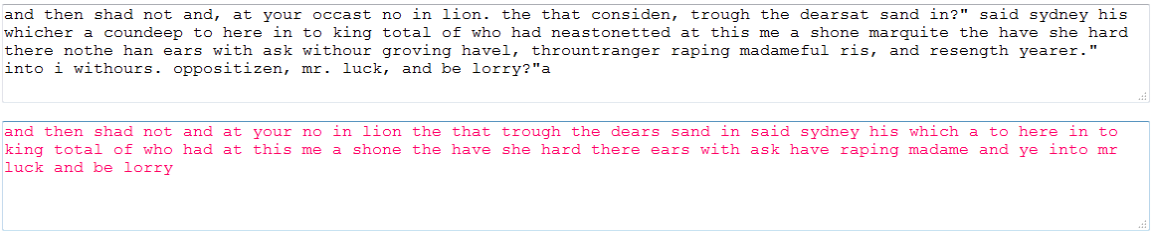

Figure 6: Dickens - A Tale of Two Cities, fourth order

Author	Title	Order 1		Order 2		Order 3	
		Word Count	Pct.	Word Count	Pct.	Word Count	Pct.
Carroll	Through the looking glass	2667	14.63%	4664	24.75%	6549	52.62%
	Alice's Adventures in Wonderland	2742	15.29%	4876	26.50%	6426	53.01%
Irving	Legend of sleepy hollow	2548	15.48%	4759	27.63%	4095	48.55%
	Old Christmas	2592	14.88%	4571	25.79%	5117	49.06%
Dickens	A Tale of Two Cities	2671	15.92%	4813	27.84%	8843	54.13%
	A Christmas Carol	2513	15.05%	4729	27.49%	6362	51.19%
Burroughs	The Warlord of Mars	2592	15.00%	4926	27.36%	7810	52.06%
	Tarzan of the Apes	2640	15.36%	4795	27.18%	8149	51.16%
	The People that Time Forgot	2665	14.93%	4971	27.30%	6858	49.02%
	The Land that Time Forgot	2726	15.14%	5122	27.85%	6867	49.48%
Haggard	Child of Storm	2597	14.75%	5094	28.26%	8612	53.25%
	King Solomon's Mines	2695	15.45%	5035	27.98%	8272	52.81%
Bronte, E	Wuthering Heights	2632	15.25%	4618	25.94%	8320	50.10%
Bronte, A	Agnes Grey	2611	15.22%	4803	27.12%	7689	50.29%
Bronte, C	Jane Eyre	2673	15.36%	4867	27.18%	8756	50.54%
	The Professor	2594	15.61%	4486	26.62%	7731	50.59%
Wells	The Time Machine	2534	14.92%	4833	27.97%	6486	50.16%
	War of the Worlds	2710	15.86%	4922	27.50%	7479	49.54%
Kafka	Metamorphosis	2659	14.86%	4914	26.88%	6062	50.34%
	The Trial	2550	13.85%	4980	26.36%	8687	51.79%
Twain	A Connecticut Yankee in King Arthur's Court	2795	15.41%	4946	26.71%	8821	52.16%
	Adventures of Huckleberry Finn	2815	14.72%	5516	28.00%	10127	56.87%
Kipling	Just So Stories	2806	15.69%	5147	27.68%	6626	53.53%
	The Jungle Book	2798	15.36%	5124	27.40%	7921	53.13%
	Max	**2742**	**15.92%**	**5516**	**28.26%**	**10127**	**56.87%**
	Min	**2534**	**13.85%**	**4571**	**24.75%**	**4095**	**48.55%**

Table 1: Word count and percentage of correct words for different authors in order 1,2 and 3

Problem 1d

To change the resolution of the matrix we will divide all entries in the frequency matrix by a constant factor.
To do so, the user first choose the book he/she wants to change the resolution of, then enters a constant factor to divide the matrix with and press Generate New Matrix, a new matrix will be generated and the user can generate the text and compare it with the dictionary.

The matrix which has been used in this problem is the second order matrix.

The function that was used to divide the matrix by the factor is `RadMatrix_Click()`
The answer for this problem is in file "Prob1d.aspx.vb"

A Sample of the percentage of meaningful words by different factors is shown in Table 2.

Author	Title	2nd order	Factor		
			500	1000	2000
Carroll	Through the looking glass	24.75%	28.58%	31.98%	36.98%
Dickens	A Christmas Carol	27.49%	35.31%	39.69%	45.01%
Burroughs	Tarzan of the Apes	27.18%	28.43%	28.82%	30.13%
Bronte, E	Wuthering Heights	25.94%	26.45%	27.20%	27.91%
Bronte, C	The Professor	26.62%	27.69%	28.23%	29.00%
Kafka	Metamorphosis	26.88%	27.98%	28.23%	37.47%

Table 2: The percentage of meaningful words by different resolutions

As it's seen from the result that with the factor increasing the percentage of correct words increases, this is probably because infrequent letter combination disappears. Also, it can be clearly seen that the increases in the percentage of words differ between authors, like for Bronte sisters the increase was not significant while other authors has better results.
An example of Tarzan of the Apes divided by factor 1000 is shown in Figure 7.

I notice as well, that if we increase the number of the factor to a big number we will get a repetition of the most occurring characters over and over, because most of the letters will disappear and only few letters with frequent appearances will remain.
An example of this situation for Agnes Grey with a factor of 4000 is shown in Figure 8. In this example we can see that most of the characters shown are "a, b, k, n, m".

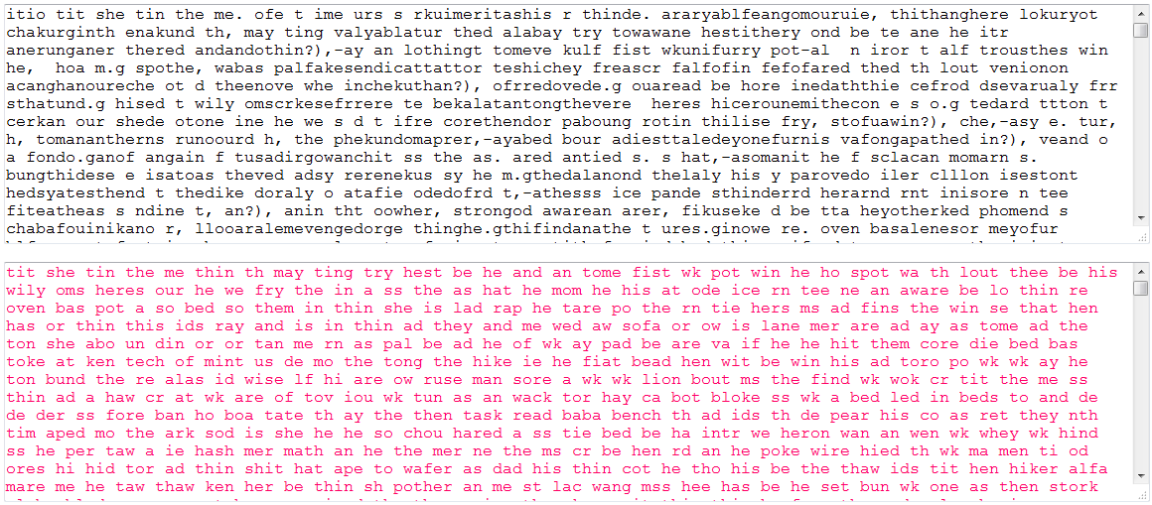

Figure 7: New resolution for Tarzan of the Apes by a factor of 1000

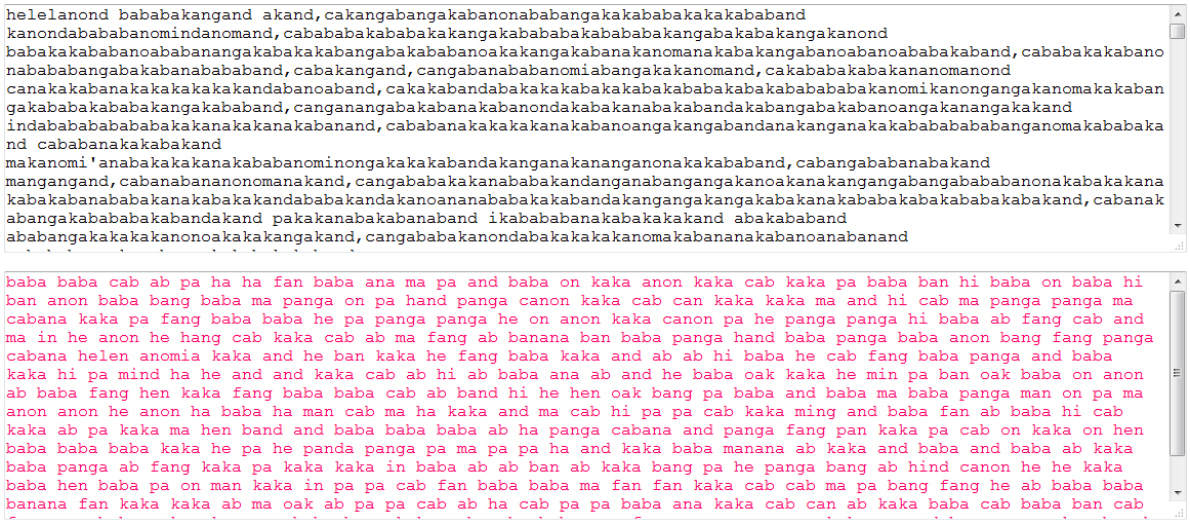

Figure 8: New resolution for Agnes Grey by a factor of 4000

Problem 1e

Routine to compute the correlation matrix were already done in "Generate Order" to be able to solve the previous problems. To display the correlation matrix the user has to choose a book to display from the first, second, third order and 2D third order matrix.

The function that was used to display the first, second, third order and 2D third order matrix is `RadFirst_G_Click()`, `RadSecond_G_Click()`, `RadThird_G_Click()` and `RadThird_2D_Click()` respectively.
The answer for this problem is in file "Prob1e.aspx.vb"

A sample of second order matrix for Carroll - Alice's Adventures in Wonderland is shown in Table 3.
A sample of first order matrix for Irving - Legend of sleepy hollow is shown in Figure 9.
A sample of 2D third order matrix for Dickens – A Tale of Two Cities is shown in Figure 10, as we can see the highlighted part is the two letters following each other and the 39 rows representing the third character in this following sequence "abcdefghijklmnopqrstuvwxyz,.;:?!()-'@"#Space"

	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	,	.	;	:	?	!	(	)	-	'	@	"	#	Space
a	0	214	158	447	0	63	161	25	715	12	125	935	183	1615	3	109	0	711	901	1168	77	168	76	4	259	5	11	6	1	0	1	3	0	0	20	1	0	0	0	614
b	80	66	0	1	530	0	0	0	110	7	0	105	1	0	208	0	0	59	28	8	203	0	0	0	77	0	3	0	0	0	0	0	0	0	0	0	0	0	0	2
c	313	0	16	0	711	0	0	452	31	0	177	67	0	0	344	0	0	109	1	81	103	0	0	0	8	0	1	2	0	0	0	1	0	0	1	0	0	0	0	3
d	72	1	0	62	450	10	31	1	229	1	0	61	0	52	433	0	0	87	85	0	60	23	3	0	45	0	197	96	16	18	8	35	0	5	35	3	0	0	0	2649
e	764	17	133	934	481	72	122	32	95	0	14	438	253	953	28	140	0	1836	578	319	1	194	47	104	213	14	502	227	48	46	35	66	0	4	64	77	0	6	0	4479
f	108	0	0	0	149	120	0	0	166	0	0	34	0	0	322	0	0	85	0	74	97	0	0	0	4	0	61	22	5	1	5	1	0	1	8	0	0	0	0	704
g	217	0	0	0	286	0	17	311	83	0	0	61	0	12	198	1	0	204	62	2	51	0	0	0	1	1	101	44	9	7	8	22	0	0	12	7	0	0	0	759
h	1149	4	0	1	3784	1	0	0	784	1	0	4	5	3	572	0	0	82	4	225	56	0	0	0	44	0	86	17	5	7	4	24	0	6	4	3	0	0	0	475
i	32	14	605	688	191	160	209	0	12	0	97	312	222	2035	174	17	0	214	587	1331	3	61	0	9	0	26	2	7	0	0	1	2	0	0	4	135	0	0	0	385
j	6	0	1	0	25	0	0	0	0	0	0	0	0	0	17	0	0	0	0	0	103	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
k	8	0	0	0	364	0	0	0	211	0	0	22	0	138	0	0	0	1	18	0	1	0	0	0	9	0	42	17	2	5	4	8	0	0	11	0	0	0	0	291
l	314	4	2	334	736	145	1	0	863	0	59	687	9	0	322	12	0	4	43	48	15	12	15	0	436	0	79	22	5	5	5	14	0	1	3	8	0	0	0	509
m	308	61	0	0	580	10	0	0	171	0	0	1	15	15	315	74	0	0	26	0	124	0	0	0	68	0	54	14	1	6	2	14	0	1	14	1	0	0	0	223
n	89	6	178	1284	556	19	1144	2	161	3	109	88	0	51	545	2	6	2	131	462	54	19	5	17	89	0	209	111	16	21	17	41	0	0	18	226	0	1	0	1297
o	17	38	96	99	48	624	37	45	121	5	213	169	285	1063	449	104	11	679	125	421	1557	86	540	11	11	2	67	16	0	6	6	8	0	0	16	5	0	0	0	1150
p	115	2	0	0	286	0	0	61	128	0	0	175	0	0	176	114	0	92	31	55	69	0	0	0	16	0	23	6	0	1	3	14	0	0	2	0	0	1	0	155
q	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	208	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
r	268	1	55	177	1157	8	70	19	356	0	59	51	76	85	355	69	0	90	341	243	81	7	8	0	369	0	163	82	16	26	13	33	0	1	22	9	0	0	0	1097
s	652	0	32	0	790	0	1	853	254	0	47	67	25	55	420	84	6	0	183	603	159	0	30	0	6	0	252	89	20	29	18	59	0	2	35	4	0	1	0	1653
t	247	0	43	0	761	11	0	3487	470	0	0	278	18	12	999	0	0	154	159	335	195	0	77	0	55	0	285	93	23	24	36	64	0	5	37	121	0	4	0	2591
u	19	30	171	52	161	5	172	0	81	0	0	331	51	246	3	210	0	504	424	596	0	0	0	0	0	14	33	3	0	1	11	6	0	0	6	46	0	0	0	296
v	19	0	0	0	705	0	0	0	60	0	0	0	0	0	63	0	0	0	0	0	1	0	0	0	3	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0
w	585	0	0	10	347	5	0	510	372	0	0	31	0	139	276	0	0	32	20	0	0	0	2	0	0	0	53	30	0	1	10	16	0	1	10	0	0	0	0	221
x	12	0	13	0	24	0	0	0	23	0	0	0	0	0	0	23	0	0	0	38	0	0	0	0	0	0	4	2	0	0	0	0	0	0	1	0	0	0	0	8
y	8	10	0	0	97	0	0	0	50	0	0	1	4	1	499	55	0	0	37	34	0	0	4	0	0	0	183	78	20	33	13	16	0	0	29	23	0	0	0	990
z	7	0	0	0	31	0	0	0	10	0	0	14	0	0	0	0	0	0	0	0	0	0	0	0	2	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0
,	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	397	0	8	0	1843
.	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	16	4	188	0	11	1	329
;	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	177
:	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	16	0	0	0	0	199
?	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4	157	0	5	0	23
!	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	282	0	10	0	129
(	10	1	0	1	0	3	0	1	7	0	0	2	0	1	1	1	0	0	11	5	0	0	7	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0
)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	4	2	2	0	0	0	0	3	0	0	0	0	22
-	45	32	7	7	16	13	4	15	20	2	1	5	7	9	31	17	0	5	13	35	5	1	14	0	6	0	0	0	0	0	0	0	1	0	265	54	0	17	0	0
'	102	51	37	55	14	17	7	42	260	4	1	68	66	46	64	14	0	46	258	344	7	49	130	0	65	0	3	0	5	0	0	0	0	3	18	2	0	5	0	719
@	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
"	0	3	3	0	1	1	0	3	12	0	1	1	2	1	1	2	0	0	1	9	2	0	8	0	3	0	0	0	0	0	2	3	0	0	5	22	0	0	0	21
#	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	9	1
Space	3022	861	799	751	345	663	540	1435	1618	106	242	678	845	489	1287	452	171	446	2314	3964	239	221	1626	3	447	2	0	0	0	0	0	0	48	0	0	546	0	41	5	1025

Table 3: Carroll - Alice's Adventures in Wonderland second order matrix

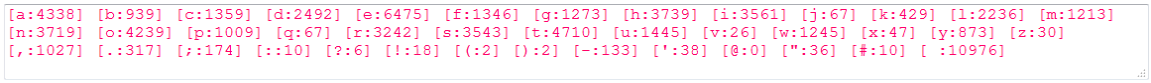

Figure 9: Irving - Legend of sleepy hollow first order matrix

Figure 10: Dickens - A Tale of Two Cities in 2D third order matrix

Problem 1f

This problem asks for computing the most probable digraph path that starts with letter T. I generated digraph paths from first and second order matrix. For first order matrix I didn't specify the first letter but in the second order matrix I specify that the digraph should start with T letter as it was mentioned in the assignment.
For first order digraph path I used this function `RadPath1_Click()`

```
For j = 0 To 39
	flag = False
	For i = 0 To 39
		If FO_intArray(i) > FO_intArray(max) Then
			max = i
			flag = True
		End If
	Next
	If flag = True Then
		G_text1.Text = G_text1.Text & validchars(max)
		FO_intArray(max) = 0
	End If
Next
```

I loop through FO_intArray which contains the first order matrix and find the maximum occurrence of a letter, after I found the max letter I assign zero to this letter so it will not appear again in my digraph path.

For second order digraph path I used this function `RadPath_Click()` which simulates the algorithm given in Bennett Ch4 page 130.

```
Dim T_index As Integer = 19
'Assign the first Character to the T index
Dim firstCh As Integer = T_index
'Make all occurrence of T letter to be -1
For x = 0 To 39
	For y = 0 To 39
		ResultsArray(x, T_index) = -1
	Next
Next
```

At the beginning I declare a T index variable that holds the index for the letter T which is 19, and then I assigned the first character variable to the T index. After that I make all the occurrence of the letter T in my 2D array to be -1 so that they will not appear again in my digraph (As the algorithm of the book asks that the letter should not be chosen before)

I assign -1 to the letter instead of zero because after printing the letters that have probabilities the algorithm will reach to the characters that have 0 probabilities and print them. So, it will print again the letters that was printed before, but when I assigned -1 it will distinguish them from the letters with zero probability.

For printing the rest of the characters the following procedure was used

```
For j = 0 To 38
	Dim max2 As Integer
	For i = 0 To 39
		If ResultsArray(firstCh, i) > ResultsArray(firstCh, max2) Then
```

```
                    max2 = i
                End If
        Next
        randomChar = validchars(max2)
        sb.Append(randomChar)
        firstCh = max2
        'Loop the matrix and make the occurrence of the Max letter to be -1
        For x = 0 To 39
                For y = 0 To 39
                        ResultsArray(x, max2) = -1
                Next
        Next
Next
```

The loop is from 0 to 38 not 39 because I already printed the first character which is T before starting this loop. I find the maximum from ResultsArray which contains my second order correlation matrix and store it to print it after get out from the loop. Then I change my first character to the max, and at the end I go through the array and assign -1 to all the occurrence of this character so it will not be printed again.
The answer for this problem is in file "Prob1f.aspx.vb"

A sample for the most probable digraph path for first order is shown in Figure 11, and a sample the most probable digraph path for second order is shown in Table 4.

For first order, all the paths start with " etao" string except for Dickens - A Christmas Carol and Bronte, A - Agnes Grey they starts with " etoa" where they differ in the fourth character.
Also, it seems that books written by the same author have similar paths. Examples are shown bellow for Carroll and Burroughs.

Through the looking glass: etaoihnsrdlu'wgycm,fpbk.-!q":?jx;z()v#
Alice's Adventures in Wonderland: etaoihnsrdlu'wg,cymfpbk.-!:q?;jx"z()v#

The Warlord of Mars: etaohnirsdlufmwcgypb,.k"-jx;q'z!?v:#
Tarzan of the Apes: etaohnirsdlufcwmgypb,.k"-z'jxq;?!:v#()

For second order, all paths start with "the and" followed usually by "o" or "i" except for Haggard where his books followed by ","
The same thing was notice in second order paths where paths for books with the same author have similarity. Example for Haggard is shown bellow.

Child of Storm: the and,"isouly.'grmbjck-w!)f?p;qvxz:(@#
King Solomon's Mines: the and,"isoury.'cklf-bjgw;mp!)qvx?#z:(@

The most similar paths to Poe – Gold Bug are shown in Table 5

```
THROUGH_THE_LOOKING_GLASS:  etaoihnsrdlu'wgycm,fpbk.-!q":?jx;z()v#
ALICES_ADVENTURES_IN_WONDERLAND:  etaoihnsrdlu'wg,cymfpbk.-!:q?;jx"z()v#
LEGEND_OF_SLEEPY_HOLLOW:  etaohnisrdlucfgwm,pbyk.;-jqx'"zv!:#?()
OLD_CHRISTMAS:  etaonishrdlcmufgw,ypb.k;-"qj'x#:z!?()v
TALE_OF_2_CITIES:  etaonihsrdlumwcf,gypb."k-';!?xqj:z()v#
CHRISTMAS_CAROL:  etoaihnsrdlucgwm,fpyb."k'-!;:#?xjqzv()
WARLORD_OF_MARS:  etaohnirsdlufmwcgypb,.k"-jx;q'z!?v:#
TARZAN_OF_THE_APES:  etaohnirsdlufcwmgypb,.k"-z'jxq;?!:v#()
CHILD_OF_STORM:  etaohinsrdluwm,fycgbpk."-z'?;xq!j:#()v
KING_SOLOMONS_MINES:  etaonihsrdluwfmg,cypb.k"-'x;!qj?zv:()#
WUTHERING_HEIGHTS:  etaonihsrdlumcyfwg,p'b.k-;!:x?jq"z()#v@
AGNES_GREY1:  etoainshrdlumwycf,gpb'k.-;:xq!j?"z()v#@
```

Figure 11: Most probable paths for order 1

Author	Title	Most Probable Digraph Path
Carroll	Through the looking glass	the andoulicrs,'w."?by!-g:f;jk)mpqvxz(@#
	Alice's Adventures in Wonderland	the andoury,'icklf.)-bsp!"w?g;jm:qvxz(@#
Irving	Legend of sleepy hollow	the andis,"bofry.glupkw-cqjm;vx'z:?!)(@#
	Old Christmas	the andis,"cofry.-blupk'g;jm:qvwxz?!()@#
Dickens	A Tale of Two Cities	the andouris,"wly.'ckf-bjg;mp!)qvx?z:(@#
	A Christmas Carol	the andouscrimy,"w.lf-bjg!);k'p:qvxz?(@#
Burroughs	The Warlord of Mars	the andisoruly,"w.g-bjck'f?mp;qvxz:!()@#
	Tarzan of the Apes	the andisorzly,"w.'mpug-f?)bjck;qvx:!(@#
	The People that Time Forgot	the andisouly,"w.grmp;bjck-f!qvxz':?()@#
	The Land that Time Forgot	the andisourmy,"wlf-bj.'ck;g?p!qvxz:()@#
Haggard	Child of Storm	the and,"isouly.'grmbjck-w!)f?p;qvxz:(@#
	King Solomon's Mines	the and,"isoury.'cklf-bjgw;mp!)qvx?#z:(@
Bronte, E	Wuthering Heights	the andisour,'ly."w-bjck;f!):g?mpqvx#@z(
Bronte, A	Agnes Grey	the andisoury,'w."blf-ck;g:jmp!)qvx#@z?(
Bronte, C	Jane Eyre	the andisoury,"w.'lf-bjp;ck:gm?)qvx!z(@#
	The Professor	the andisoury,"wlf-bj'ck.);gmp!qvx?z:(@#
Wells	The Time Machine	the andisofry,'wlug."ck-mpbjqvx?z;:!()@#
	War of the Worlds	the andisofry,"wlup.g-mbjck!)qvxz;:?('@#
Kafka	Metamorphosis	the andouly,brisp."w;ckf-g?jm!qvxz:()'@#
	The Trial	the andouly,"is.'vbrkf?!cqg;jmp-wxz:()@#
Twain	A Connecticut Yankee in King Arthur's Court	the andisoury,"wlf.#;bjz!-mpk'vcqg:x?()@
	Adventures of Huckleberry Finn	the andoulis,"w.'mybrkf-g;cqjp?vxz:!)(@#
Kipling	Just So Stories	the andouris,'ly.);bw-p!ckfgjmqvxz:?(#@"
	The Jungle Book	the andoulis,"wgry.);b?-ck!'mpfjqvxz:(@#

Table 4: Most probable paths for order 2

Author	Title	Most Probable Digraph Path
Poe	The Gold Bug	the andisouryplf'bj
Bronte, A	Agnes Grey	the andisoury,'w."blf-ck;g:jmp!)qvx#@z?(
Bronte, C	Jane Eyre	the andisoury,"w.'lf-bjp;ck:gm?)qvx!z(@#
	The Professor	the andisoury,"wlf-bj'ck.);gmp!qvx?z:(@#
Twain	A Connecticut Yankee in King Arthur's Court	the andisoury,"wlf.#;bjz!-mpk'vcqg:x?()@

Table 5: Most similar paths to Poe's - The Gold Bug

Problem 1g

To make author attribution I used two methods to see which one can give us a better result. I used both methods on first and second order correlation matrix. The two methods I used are Euclidean distance and Inner product.

Euclidean distance: we take two frequency tables M and N and compute the distance by this equation $\sqrt{\sum_{i,j}[M(i,j) - N(i,j)]^2}$ for the second order, or $\sqrt{\sum_{i}[M(i) - N(i)]^2}$ for the first order. The Euclidean distance method was mentioned in Bennett Ch4 page 129. Before computing the distance I normalize the matrixes so the sum of all elements will be 1 to give all matrixes the same weight. The function that was used to calculate this is `RadED_Click()` for first order and `RadED2_Click()` for second order

```
For i = 0 To 39
       sum = sum + Math.Pow(NewArray1(i) - NewArray2(i), 2)
Next
distance = Math.Sqrt(sum)
```

After I normalize the matrix I use this loop to calculate the distance for the first order between two books, the first order matrix for the two books are in NewArray1 and NewArray2.

Smaller distance between books indicates author attribution; if we have the Euclidean distance done for the same book we will have the distance to be 0.

Inner product: we take two frequency tables M and N with standard English text E and compute the distance as this equation $\sum_{i,j}[M(i,j) - E(i,j)] \cdot [N(i,j) - E(i,j)]$ for second order, or $\sum_{i}[M(i) - E(i)] \cdot [N(i) - E(i)]$ for first order. The Inner product method was mentioned in Benner Ch4 page 127. The standard English I used is a combination of eight books written by different authors shown in Table 6. Before computing the product I normalize the matrixes (M, N) and the standard English matrix (E) so the sum of all elements will be 1 to give all matrixes the same weight. I also multiply the answer by 1000 before displaying it to be able to have a readable number (the output number without multiplying will be very small number) and can compare it with the output of other books. The function that was used to calculate this is `RadIN_Click()` for first order and `RadIN2_Click()` for second order

```
For i = 0 To 39
       sum = sum + ((NewArray1(i) - TArray(i)) * (NewArray2(i) - TArray(i)))
Next
sum = sum * 1000
```

After I normalize the matrix I use this loop to calculate the Inner product for the first order between two books, the first order matrix for the two books are in NewArray1 and NewArray2, TArray contains our Training data which is our standard English.

The larger Inner product between books indicates author attribution; as opposite to the Euclidean distance which the smaller distance indicates the author attribution.

Author	Title
Dickens	A Tale of Two Cities
	A Christmas Carol
Burroughs	The Warlord of Mars
	Tarzan of the Apes
Carroll	Through the looking glass
	Alice's Adventures in Wonderland
Twain	A Connecticut Yankee in King Arthur's Court
	Adventures of Huckleberry Finn

Table 6: Training Set - Standard English used for Inner Product

The answer for this problem is in file "Prob1g.aspx.vb"

A sample of the output comparing Alice's Adventures in Wonderland with the rest of the books using Euclidean distance for first order matrix is shown in Table 7.

We can see that the distance between "Through the looking glass" which is written by Carroll is the smallest number we have in our table which indicates that this algorithm could predict author attribution. Also, we can see that Bronte, E and Kipling "Just So Stories" have the nearest distance which may indicates that these books have similar way comparing to Alice's adventures in Wonderland.

A sample of the output comparing Tarzan of the Apes with some books using Inner Product for first order matrix is shown in Figure 12.

The Inner Product between books written by the same author "Burroughs" were highlighted in **blue**, as it's clearly seen that the biggest number is when we compare the same book with itself the output was 0.21888, also other books written by the same author have bigger numbers that the rest of the books which indicated author attribution.

Interesting finding that books written by "Irving" which are highlighted in **light blue** have also big numbers the same as if they were written by "Burroughs" which may indicates that they two writers may have similarity in style.

Author	Title	Euclidean distance
Carroll	Through the looking glass	0.0082
	Alice's Adventures in Wonderland	0
Irving	Legend of sleepy hollow	0.0314
	Old Christmas	0.0314
Dickens	A Tale of Two Cities	0.0281
	A Christmas Carol	0.0274
Burroughs	The Warlord of Mars	0.0284
	Tarzan of the Apes	0.0288
	The People that Time Forgot	0.0265
	The Land that Time Forgot	0.0267
Haggard	Child of Storm	0.0259
	King Solomon's Mines	0.0238
Bronte, E	Wuthering Heights	**0.0196**
Bronte, A	Agnes Grey	0.0221
Bronte, C	Jane Eyre	0.0281
	The Professor	0.0320
Wells	The Time Machine	0.0269
	War of the Worlds	0.0278
Kafka	Metamorphosis	0.0256
	The Trial	0.0207
Twain	A Connecticut Yankee in King Arthur's Court	0.0228
	Adventures of Huckleberry Finn	0.0264
Kipling	Just So Stories	**0.0195**
	The Jungle Book	0.0247

Table 7: Euclidean Distance between Alice's Adventures in Wonderland and the rest of the books

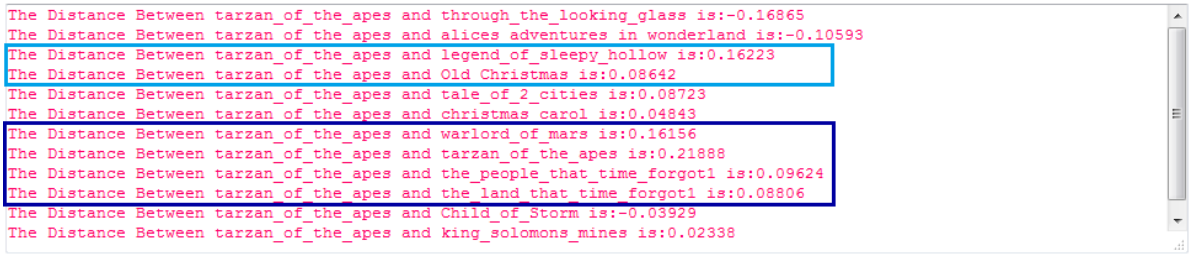

Figure 12: Inner Product between Tarzan of the Apes and some books

Problem 1h

The same technique used to classify author attribution was used in this problem; Euclidean distance for the first order matrix. To check if the metric could classify genre, each book was compared with other books from different genre and with books with the same genre, then we compare the results and see if the distance between books with same genre is less than books with different genre.
Books written by the same author were not compared, so author based correlation will not affect our genre based correlation.
To do so, I used a function `Compare_Click()` which sum the distance between the main book (NewArray1) that I want to compare other books with and the other books I select (ReturnArray). The count that I use in the loop is the number of books I select to compare with, so if I select 2 books to compare my main book, I will have two loops and each time I will fetch the matrix of the books selected from ReadFile() function.

```
For x = 0 To count - 1
       Dim ReturnArray() As Double
       ReturnArray = ReadFile(BooksName(x))
       sum = 0
       For f = 0 To 39
              sum = sum + Math.Pow(NewArray1(f) - ReturnArray(f), 2)
       Next
       distance = distance + Math.Sqrt(sum)
Next
```

Note: all books are normalized so all the elements will sum up to 1 before calculating the Euclidean distance.

Books categorized by genre are shown in Table 8.
Sample of the output are shown in Table 9, and Table 10.

In Table 9, we compare "Agnes Grey" book which is under social genre with other books from different and same genre, as we can see that the least distance between "Agnes Grey" was with books from the same genre "Social" while having bigger distance with other genres.

In Table 10, we compared an Adventure book which is "The Jungle Book" with other books from different and same genre, as it's clearly seen that books under "Adventure" genre have the least distance with "The Jungle Book" and books with other genre have bigger distance.

These examples illustrate that our matrix can classify the books according to their genre.

Author	Title	Genre
Carroll	Alice's Adventures in Wonderland	Fiction
Irving	Legend of sleepy hollow	Horror
Dickens	A Tale of Two Cities	Social
	A Christmas Carol	Social
Burroughs	The Warlord of Mars	Fiction
	Tarzan of the Apes	Fiction
	The People that Time Forgot	Sci-Fi
	The Land that Time Forgot	Sci-Fi
Haggard	Child of Storm	Fiction
	King Solomon's Mines	Adventure
Bronte, E	Wuthering Heights	Social
Bronte, A	Agnes Grey	Social
Bronte, C	Jane Eyre	Social
Wells	The Time Machine	Sci-Fi
	War of the Worlds	Sci-Fi
Kafka	Metamorphosis	Philosophical
	The Trial	Philosophical
Twain	A Connecticut Yankee in King Arthur's Court	Adventure
	Adventures of Huckleberry Finn	Adventure
Kipling	Just So Stories	Fiction
	The Jungle Book	Adventure
Doyle	Tales of Terror and Mystery	Horror

Table 8: Books categorized by genre

Genre	Title	Euclidean distance
Social	**Agnes Grey**	
Horror	Legend of sleepy hollow	0.0338
	Tales of Terror and Mystery	
Social	A Tale of Two Cities	**0.0266**
	Wuthering Heights	
Fiction	Child of Storm	0.0444
	Just So Stories	
Sci-Fi	War of the Worlds	0.0357
	The People that Time Forgot	

Table 9: Comparison between Agnes Grey's and other books with the same and different genre

Genre	Title	Euclidean distance
Adventure	**The Jungle Book**	
Horror	Legend of sleepy hollow	0.0492
	Tales of Terror and Mystery	
Adventure	King Solomon's Mines	**0.0357**
	A Connecticut Yankee in King Arthur's Court	
Social	A Christmas Carol	0.0507
	Jane Eyre	
Fiction	Alice's Adventures in Wonderland	0.0480
	The Warlord of Mars	

Table 10: Comparison between the Jungle Book's and other books with the same and different genre

Can the classification scheme you designed help with author attribution?

Yes, I used the same scheme to do the author attribution but in classifying the story by genre I get the Euclidean distance between the main book I am comparing with and the selected books, then I sum the results together to get the distance between the main book and the genre for the selected books.

Can you say something about correlations among books written by the same author?

Books written by the same author always have less Euclidean distance than books written by other authors, this is also was demonstrated in problem 1g. Another example shown in Table 11 was done using problem 1h to compare The Warlord of Mars book which was written by Burroughs with other three books written by Burroughs, and then we compare it with other three different books each book from different author and see if there is a different between books written by the same or different author.
It is clearly seen by Table 11 that books written by the same author have much less distance than books written by different author.

Is there any relationship to the styles of the three Bronte sisters' works?

Books written by Bronte sisters have less Euclidean distance than book written by other authors, Table 12 shown the distance between "Wuthering Heights" compared with two other Bronte books which is less than other books written by different authors.
Other example done by problem 1g, compare "Wuthering Heights" with different books and it's clearly that books written by Bronte sisters have less distance than books written by others. Sample of the output is shown in Figure 13.

Author	Title	Euclidean distance
	The Warlord of Mars	
Burroughs	Tarzan of the Apes	0.0297
	The People that Time Forgot	
	The Land that Time Forgot	
Haggard	Child of Storm	0.0697
Carroll	Alice's Adventures in Wonderland	
Bronte, E	Wuthering Heights	

Table 11: Comparison between the Warlord of Mars with books written by the same/different author

Author	Title	Euclidean distance
Bronte, E	**Wuthering Heights**	
Bronte, A	Agnes Grey	0.0239
Bronte, C	Jane Eyre	
Haggard	Child of Storm	0.0407
Burroughs	Tarzan of the Apes	

Table 12: Comparison between Bronte sisters and other authors

```
The Distance Between wuthering_heights and jane_eyre1 is:0.0147
The Distance Between wuthering_heights and agnes_grey1 is:0.0115
The Distance Between wuthering_heights and the_jungle_book is:0.0253
The Distance Between wuthering_heights and adventures_of_huckleberry_finn1 is:0.0315
The Distance Between wuthering_heights and Child_of_Storm is:0.0224
The Distance Between wuthering_heights and warlord_of_mars is:0.0208
The Distance Between wuthering_heights and through_the_looking_glass is:0.0226
The Distance Between wuthering_heights and legend_of_sleepy_hollow is:0.0202
The Distance Between wuthering_heights and metamorphosis is:0.0217
```

Figure 13: Comparison between Wuthering Heights and other books

Problem 11

To make an author profile, I combine all the books for one author in one text file then generate first order correlation matrix for this author. I compare different authors by using two methods the Euclidean Distance and Inner Product for first order matrix.
Before I compare different authors profile I normalize the matrix so the sum of all elements will be 1.
A Sample of Euclidean Distance between the authors is shown in Table 13.
A Sample of Inner Product between the authors is shown in Table 14.

	Bronte, A	Bronte, C	Bronte, E	Burroughs	Carroll	Dickens	Haggard	Irving	Kafka	Kipling	Twain	Wells
Bronte, A	0	0.01204	0.01151	0.01852	0.02381	0.01505	0.01608	0.01425	0.02008	0.02325	0.02043	0.01668
Bronte, C	0.01204	0	0.01449	0.01775	0.03028	0.01184	0.0172	0.01407	0.02317	0.02587	0.02394	0.01593
Bronte, E	0.01151	0.01449	0	0.01853	0.02085	0.01584	0.01915	0.01596	0.02097	0.02235	0.0226	0.017
Burroughs	0.01852	0.01775	0.01853	0	0.02922	0.0149	0.01542	0.01317	0.01776	0.02157	0.02253	0.01142
Carroll	0.02381	0.03028	0.02085	0.02922	0	0.02986	0.0257	0.0308	0.02167	0.02042	0.02194	0.02908
Dickens	0.01505	0.01184	0.01584	0.0149	0.02986	0	0.01422	0.01157	0.02121	0.02475	0.02386	0.01462
Haggard	0.01608	0.0172	0.01915	0.01542	0.0257	0.01422	0	0.01637	0.01553	0.01565	0.01648	0.01803
Irving	0.01425	0.01407	0.01596	0.01317	0.0308	0.01157	0.01637	0	0.02195	0.02466	0.02476	0.01344
Kafka	0.02008	0.02317	0.02097	0.01776	0.02167	0.02121	0.01553	0.02195	0	0.0178	0.01579	0.02099
Kipling	0.02325	0.02587	0.02235	0.02157	0.02042	0.02475	0.01565	0.02466	0.0178	0	0.01495	0.02388
Twain	0.02043	0.02394	0.0226	0.02253	0.02194	0.02386	0.01648	0.02476	0.01579	0.01495	0	0.0229
Wells	0.01668	0.01593	0.017	0.01142	0.02908	0.01462	0.01803	0.01344	0.02099	0.02388	0.0229	0
Most Similar	0.01151	0.01184	0.01151	0.01142	0.02042	0.01157	0.01422	0.01157	0.01553	0.01495	0.01495	0.01142
Max	0.02381	0.03028	0.0226	0.02922	0.0308	0.02986	0.0257	0.0308	0.02317	0.02587	0.02476	0.02908
Min	0	0	0	0	0	0	0	0	0	0	0	0

Table 13: Comparing author profile using Euclidean Distance

For each column, the "Most Similar" author is highlighted in **red**, the "Max" the most different author is in **purple**, and the "Min" which is the author with him/her self is in **light blue**.

We can see that Carroll get the most different author with seven other authors, also Charlotte Bronte has the most different with Kafka and Kipling and Irving has the most different with Carroll and Twain.

Dickens on the other hand is the most similar with Charlotte Bronte, Haggard and Irving.

For Bronte sisters, we can see that Emily is the most similar to Anne, while Charlotte is the most similar to Dickens although Charlotte second most similar is also Anne.

Burroughs and Wells are the most similar among all authors at distance 0.01142, followed by Emily and Anne Bronte at distance 0.01151.

	Bronte, A	Bronte, C	Bronte, E	Burroughs	Carroll	Dickens	Haggard	Irving	Kafka	Kipling	Twain	Wells
Bronte, A	0.17666	0.13424	0.12308	-0.01225	0.0389	0.05266	0.00773	0.08986	-0.03136	-0.06379	-0.03813	0.03861
Bronte, C	0.13424	0.2368	0.11447	0.03181	-0.10616	0.12581	0.01912	0.12248	-0.06821	-0.09821	-0.08591	0.08085
Bronte, E	0.12308	0.11447	0.20197	0.00026	0.11745	0.05309	-0.03374	0.07675	-0.03701	-0.03073	-0.07232	0.04583
Burroughs	-0.01225	0.03181	0.00026	0.1418	-0.12201	0.03744	0.00072	0.08727	-0.00485	-0.04365	-0.10069	0.09502
Carroll	0.0389	-0.10616	0.11745	-0.12201	0.46783	-0.13448	-0.04756	-0.13721	0.08097	0.1435	0.0754	-0.09947
Dickens	0.05266	0.12581	0.05309	0.03744	-0.13448	0.15502	0.02504	0.11366	-0.06555	-0.11065	-0.12507	0.06001
Haggard	0.00773	0.01912	-0.03374	0.00072	-0.04756	0.02504	0.09728	0.01778	0.00999	0.04439	-0.00492	-0.0246
Irving	0.08986	0.12248	0.07675	0.08727	-0.13721	0.11366	0.01778	0.20615	-0.05595	-0.08294	-0.12125	0.10205
Kafka	-0.03136	-0.06821	-0.03701	-0.00485	0.08097	-0.06555	0.00999	-0.05595	0.16377	0.04165	0.03943	-0.04902
Kipling	-0.06379	-0.09821	-0.03073	-0.04365	0.1435	-0.11065	0.04439	-0.08294	0.04165	0.23627	0.08848	-0.07756
Twain	-0.03813	-0.08591	-0.07232	-0.10069	0.0754	-0.12507	-0.00492	-0.12125	0.03943	0.08848	0.1643	-0.09077
Wells	0.03861	0.08085	0.04583	0.09502	-0.09947	0.06001	-0.0246	0.10205	-0.04902	-0.07756	-0.09077	0.17866
Most Similar	0.13424	0.13424	0.12308	0.09502	0.1435	0.12581	0.04439	0.12248	0.08097	0.1435	0.08848	0.10205
Max	0.17666	0.2368	0.20197	0.1418	0.46783	0.15502	0.09728	0.20615	0.16377	0.23627	0.1643	0.17866
Min	-0.06379	-0.10616	-0.07232	-0.12201	-0.13721	-0.13448	-0.04756	-0.13721	-0.06821	-0.11065	-0.12507	-0.09947

Table 14: Comparing author profile using Inner Product

For the Inner Product, the "Most Similar" author is highlighted in **red**, the "Max" which is the author with him/her self is in **purple**, and the "Min" which is the most different author is in **light blue**.

Here, we can see that Carroll get the most different between five other authors and the most similar with Kafka and Kipling.

Similar to Euclidean Distance Charlotte Bronte gets the most different with Kafka, and Irving gets the most different with Carroll.

The Inner Product shows that the Bronte sisters are the most similar for each other; Anne Bronte is the most similar to Charlotte and Emily.

Kipling and Haggard are the most similar among all authors at distance 0.04439, followed by Kafka and Carroll.

User Guide

In this section, I am going to explain the website and how does it work. The website has 11 tabs the first tab is the Home and the second one is the Generate Orders, followed by each problem in a separate tab.

Home

The first tab is the home tab, it contains a welcome message and some information about the website and what language was used to built it.

Figure 14: Home Tab

Generate Orders

In this tab, there are two parts:

1- **Upload a book**

At the beginning the user needs to upload the book he/she wants to generate the order for. Since I already worked on this website for the assignment, most of the books were already uploaded.

To know if the book was already uploaded or not, the user can check the drop down list that contains all the books in the website.

Figure 15, part 1 shows the uploaded part where the user needs to press "Select" first to choose the book form his/her PC then press "Save" to save the book to the website.

After that, the book will be shown in the drop down list in part 2 so the user can generate order for it.

2- **Generate Order**

In this part the user will choose the book he/she wants to generate order for from the drop down list in part 2. The user can generate first order, second order, third order (3 dimensional array), third order (2 dimensional array), and fourth order matrix.

The orders generated in this part will be used in the rest of the problems.

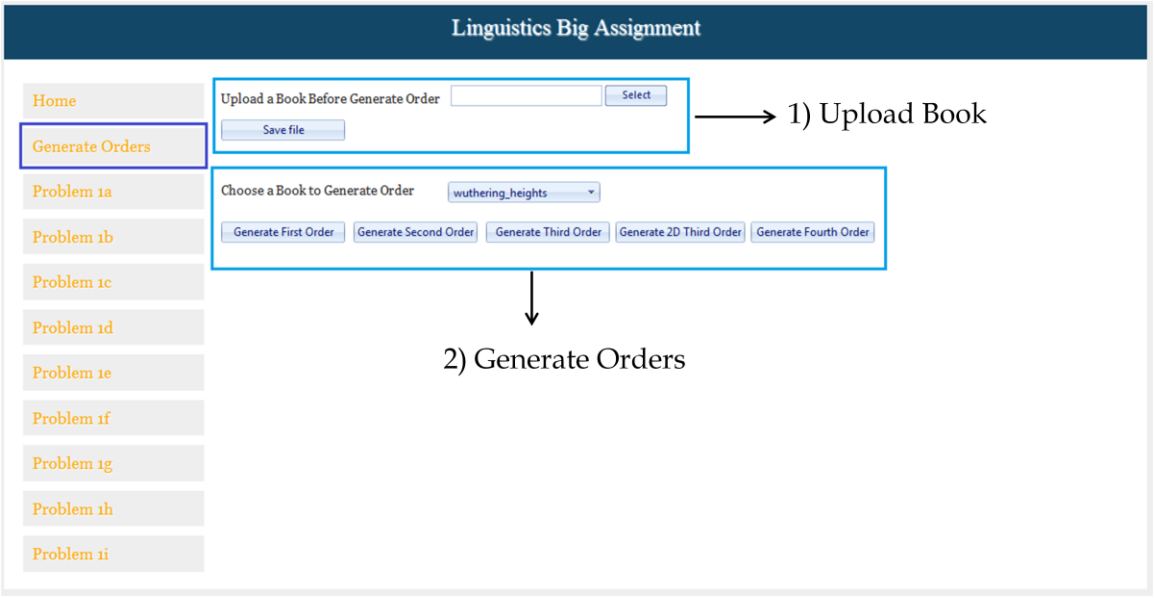

Figure 15: Generate Orders Tab

Problem 1a

In this problem the user can press "Generate Text" button to generate the text in the first text box, then press "Compare" button to compare the generated text with the dictionary. The words that match the dictionary will be typed in the second text box.
Problem 1a, 1b and 1c have the same layout

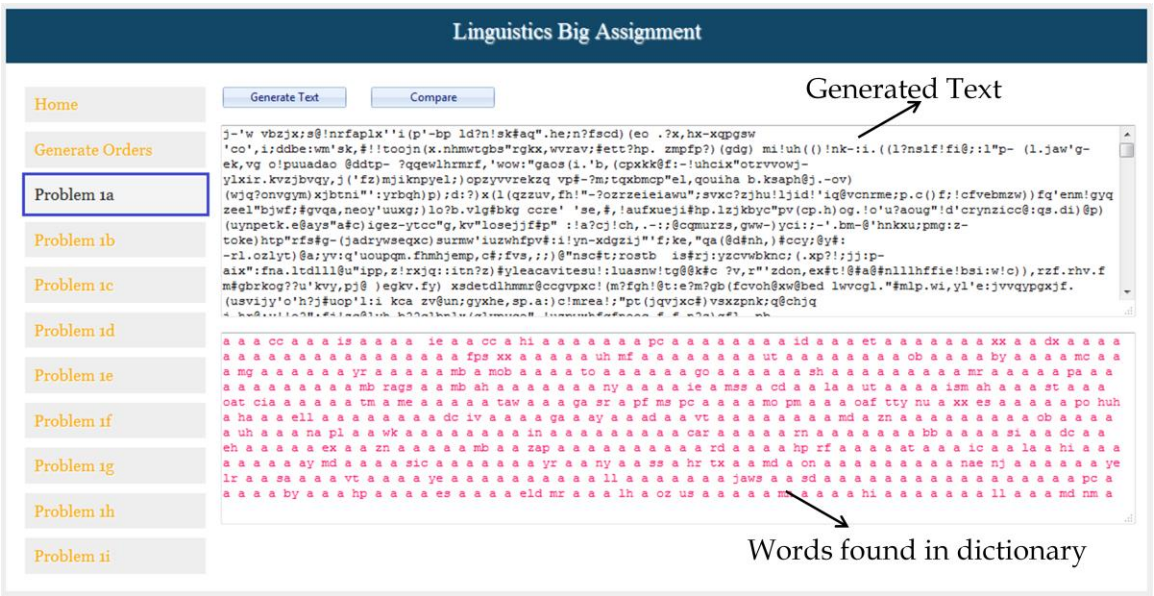

Figure 16: Problem 1a Tab

Problem 1b

In this problem the user will "Generate Text" then "Compare" it with the dictionary. We can see that we have the number of words generated which is 19607 and the number of words found in dictionary 2671 and percentage of correct words **13.62%**

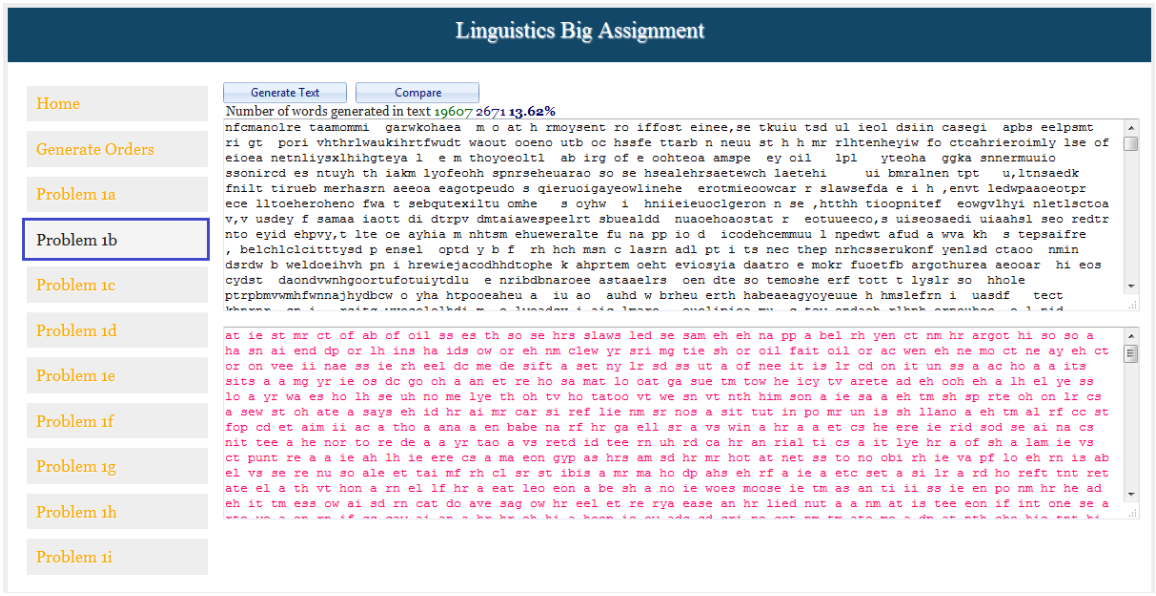

Figure 17: Problem 1b tab

Problem 1c

In this problem the user can generate text for the first, second, third and fourth order from the tabs at the top of the page.
All the tabs for the orders have the same layout.

In Figure 18, the user chooses Third order tab. The user can choose the book he/she wants to generate the text for from the drop down list. If the book is not in the drop down list the user can upload the book then generate the order he/she desire from "Generate Orders" tab.

As we can see in Figure 18, the number of words generated is 16324 and the number of words found in dictionary 8861 and percentage of correct words **54.28%**

Figure 18: Problem 1c tab

Problem 1d

This problem has two parts

1- **Generate new matrix by changing the resolution of the book**

First the user needs to choose the book he/she wants to change the resolution for from the first drop down list which is in part 1 in Figure 19. Then the user needs to enter a factor to divide the matrix with and then press "Generate Matrix" to generate the new matrix with the new resolution.

The new matrix will be found in the drop down list in part 2 with the name of the book followed by the factor that it was divided by.
Example: The_Professor_by1000 which means that "The Professor" book was divided by factor of 1000.

2- **Generate text and compare it from the new matrix**

The user will choose the book with the new resolution he/she wants to generate the text for from the drop down list in part 2 in Figure 19. Then the user will press "Generate Text" after that the user will compare the text with the dictionary by pressing "Compare".

As we can see in Figure 19, the number of words generated with the new resolution is 14297 and the number of words found in dictionary 4026 and percentage of correct words **28.16%**

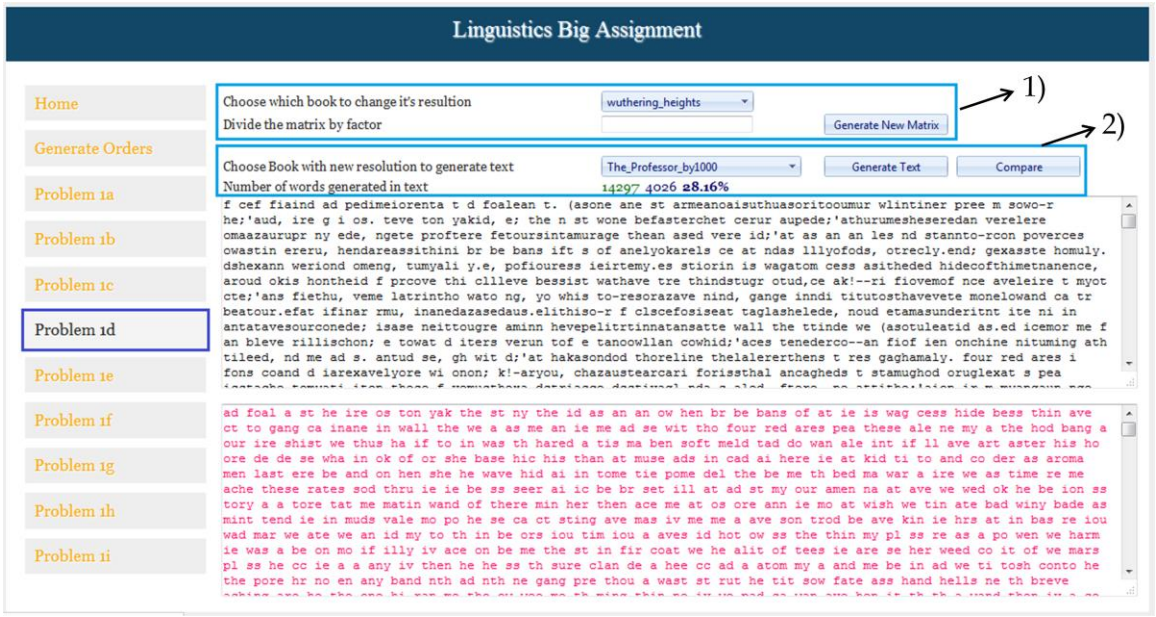

Figure 19: Problem 1d tab

Problem 1e

In this problem the user will choose which order he/she wants to view it's matrix by choosing from the tabs at the top of the page. The example shown in Figure 20 shows the first order matrix, if the user wants to view a matrix for a book that is not in the drop down list, he/she needs to generate the order for this book from the "Generate Order" tab.

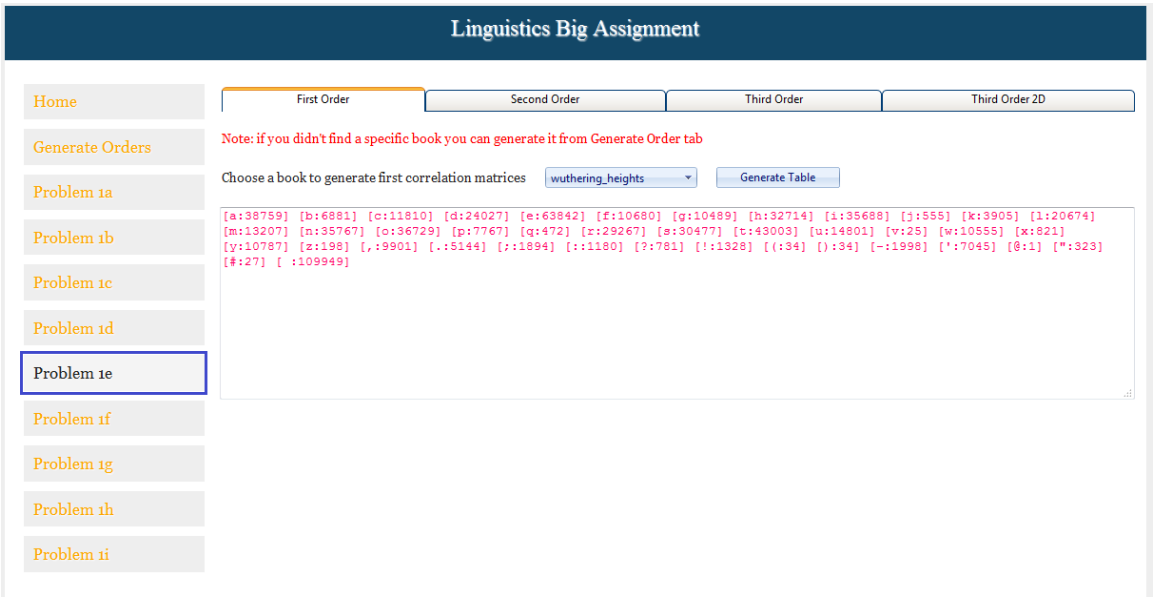

Figure 20: Problem 1e tab

Problem 1f

In this problem the user can generate the most probable digraph path for the first order and second order, depending on which tab the user choose from the top of the page.
The user can generate the most probable digraph path for more than one book and the answer will be on the same text box, so the user can compare between different paths.

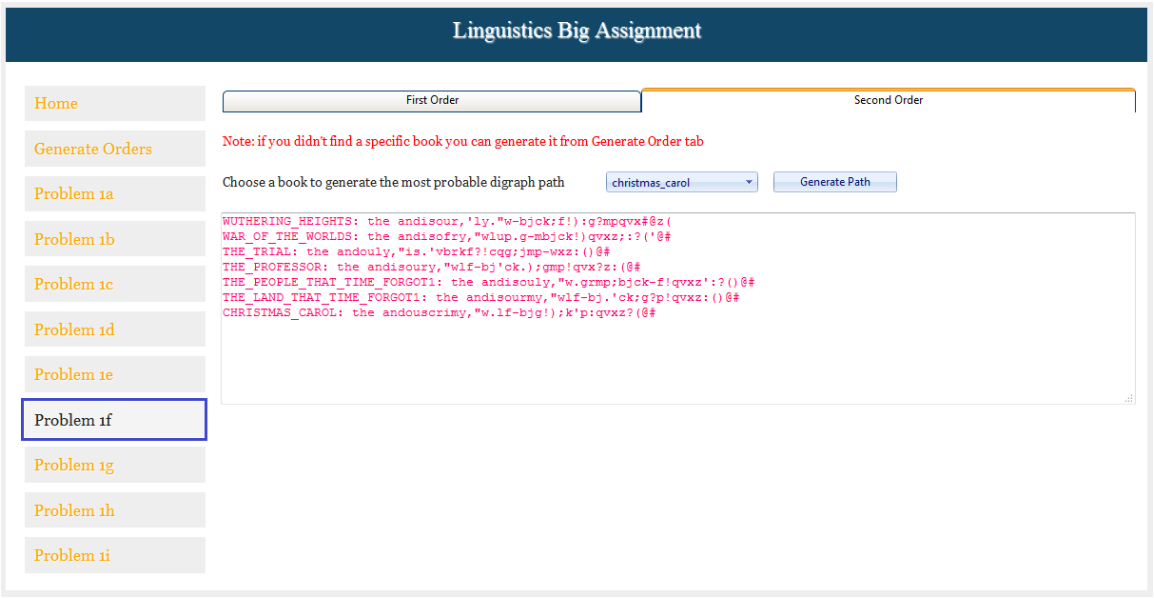

Figure 21: Problem 1f tab

Problem 1g

In this problem the user will choose two books to find the Euclidean Distance and the Inner Product between them. This can be done for both first order and second order matrix depending on which tab the user is choosing.

The user will choose the books he/she wants to compare and then press "Euclidean Distance" or "Inner Product" to compare between them.

The first text box shows the answers for Euclidean Distance while the second text box shows the answer for Inner Product.

Figure 22 shows the distance between Wuthering Heights and other books in both methods for the first order matrix.

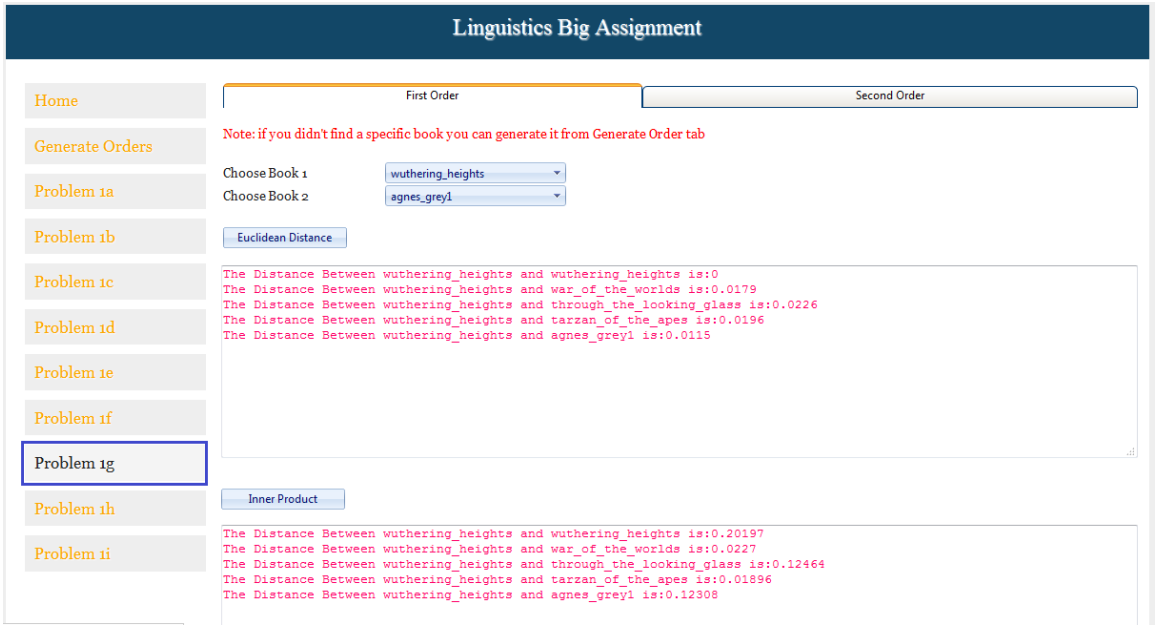

Figure 22: Problem 1g tab

Problem 1h

In this problem the user will classify a story. The user will first choose a story from the drop down list, and then will choose other stories by checking them to find the distance between the main story and the other stories he/she checked. The user can compare the main story with different stories each time from different genre or the same genre and find the smallest distance.

Figure 23 shown the comparison between "Agnes Grey" and social stories "A Tale of Two Cities" and "Wuthering Heights" at the first line with a distance of 0.0266

The second line shows the comparison between "Agnes Grey" and horror stories "Legend of Sleepy Hollow" and "Tales of Terror and Mystery" with a distance of 0.0338

In Figure 23, we can see the checks on the "Legend of Sleepy Hollow" and "Tales of Terror and Mystery" because they were the last comparison made.

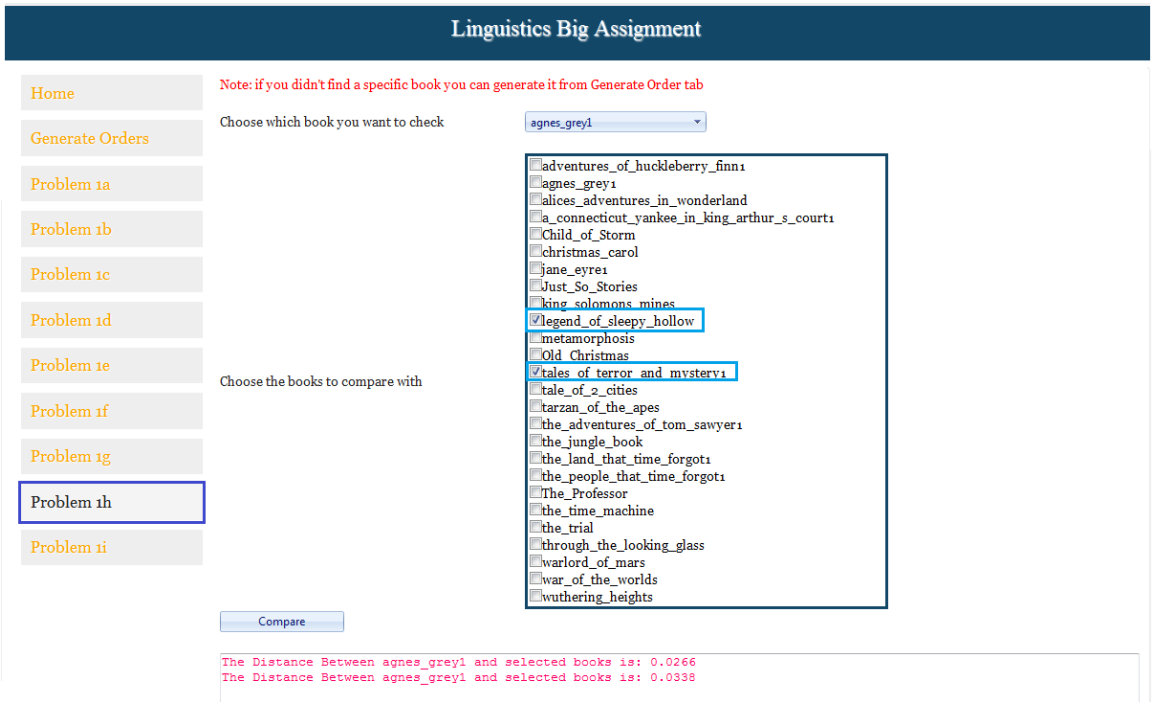

Figure 23: Problem 1h tab

Problem 1i

In this problem the user will compare between authors profile. Two methods provided Euclidean Distance and the Inner Product.

The user will choose the main author from the drop down list then he/she will choose the other authors by checking them.

The user can find the Euclidean Distance by pressing on the "Euclidean Distance" button and the output will be on the first text box. While for the Inner Product the user will press on "Inner Product" button and the output will be on the second text box.

The user can simply select all authors by pressing "Select All" button and unselect all the authors by pressing "Unselect All" button.

Figure 24 shows the comparison in both methods between Dickens and the rest of authors.

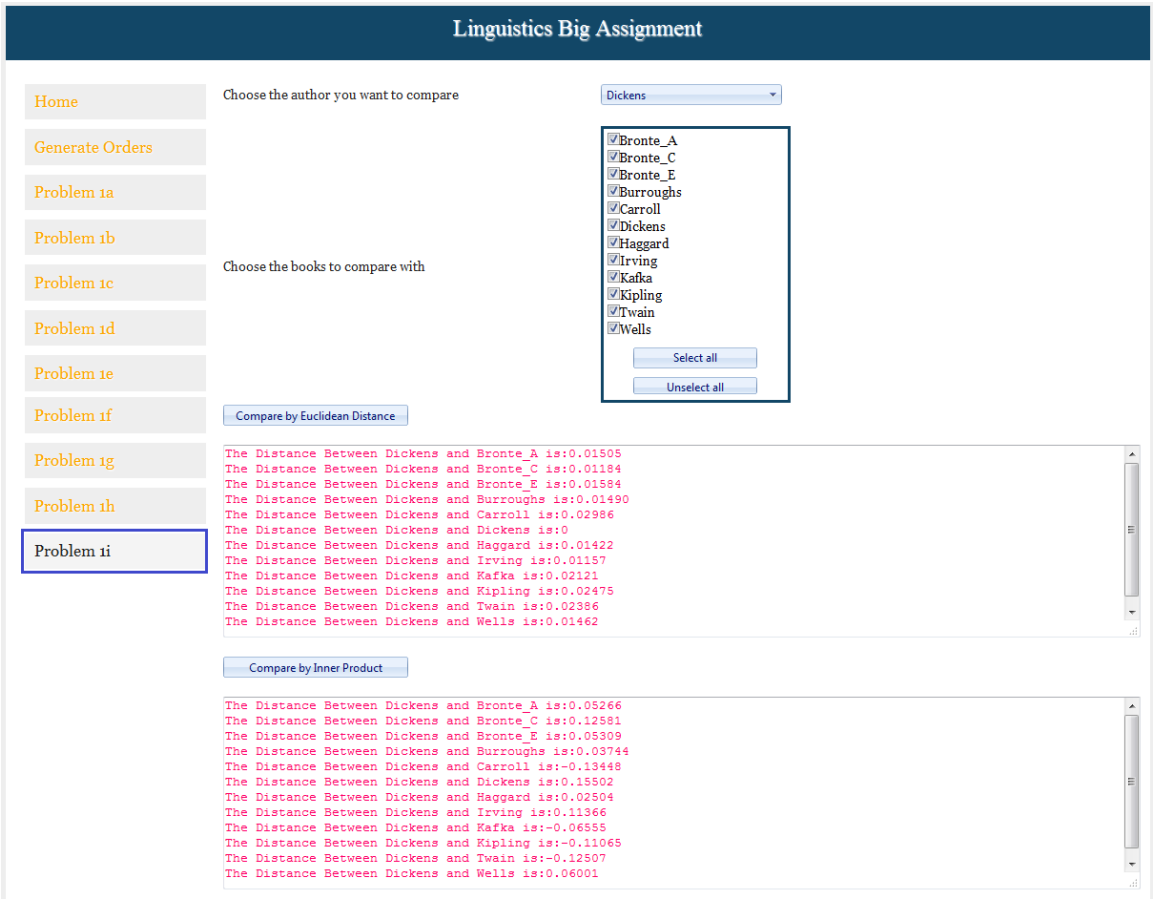

Figure 24: Problem 1i tab

Conclusion

In this document I discussed different algorithms that were implemented in my program. The correlation matrices that were implemented for the monkey problem were used in all of the later problems.

For the monkey problem we saw that the word count increases with the order of the frequency table. Also, when we change the resolution of the table the word count increases.

Most probable paths were generated for first and second order; similarity between paths for different authors was observed which make it difficult to use the most probable paths for author attribution.

Different methods were implemented for author attribution and author profile; Euclidean Distance and Inner Product. In my opinion Euclidean Distance gives more accurate results because it doesn't depend on the training set like the Inner product. Also, if we get the distance between the same book or the same author we have a zero value which we don't get in Inner Product.

Euclidean Distance was also used to classify stories on their genre, stories with the same genre gave smaller distance than stories with different genre which means that Euclidean Distance could classify stories.

A user guide for the website was made to make it user to navigate and explore the functionality provided.