

# English Composition Analyser

by Pak Ho Wong

## Design and Overview

### *Brief introduction of the software*

This ‘English Composition Analyser’ was developed in Pascal language, using Lazarus.

As the name ‘English Composition Analyser’ suggests, it was, of course, developed for analysing English compositions. Users only need to press a few buttons for a detailed statistical report of the loaded English composition. This significantly reduces the time taken to grade English compositions as users no longer need to perform those time-consuming yet indicative routines including but not limited to counting the number of words and finding the spelling mistakes. Such kind of low-level operations will now be taken over by this software.

- Developed using Pascal language
- Developed using Lazarus
- Developed for analysing English compositions

### *Purpose of the software*

This ‘English Composition Analyser’ aims to lessen the burden of people who are having a hard time grading English composition.

Grading English composition is a tough and time-consuming job. Even though the scores can or maybe a bit subjective, still, it needs to base on some objective statistics such as the total number of words and the number of spelling mistakes. Therefore, the marker needs to flip the composition over and over again before they can sort those numbers out.

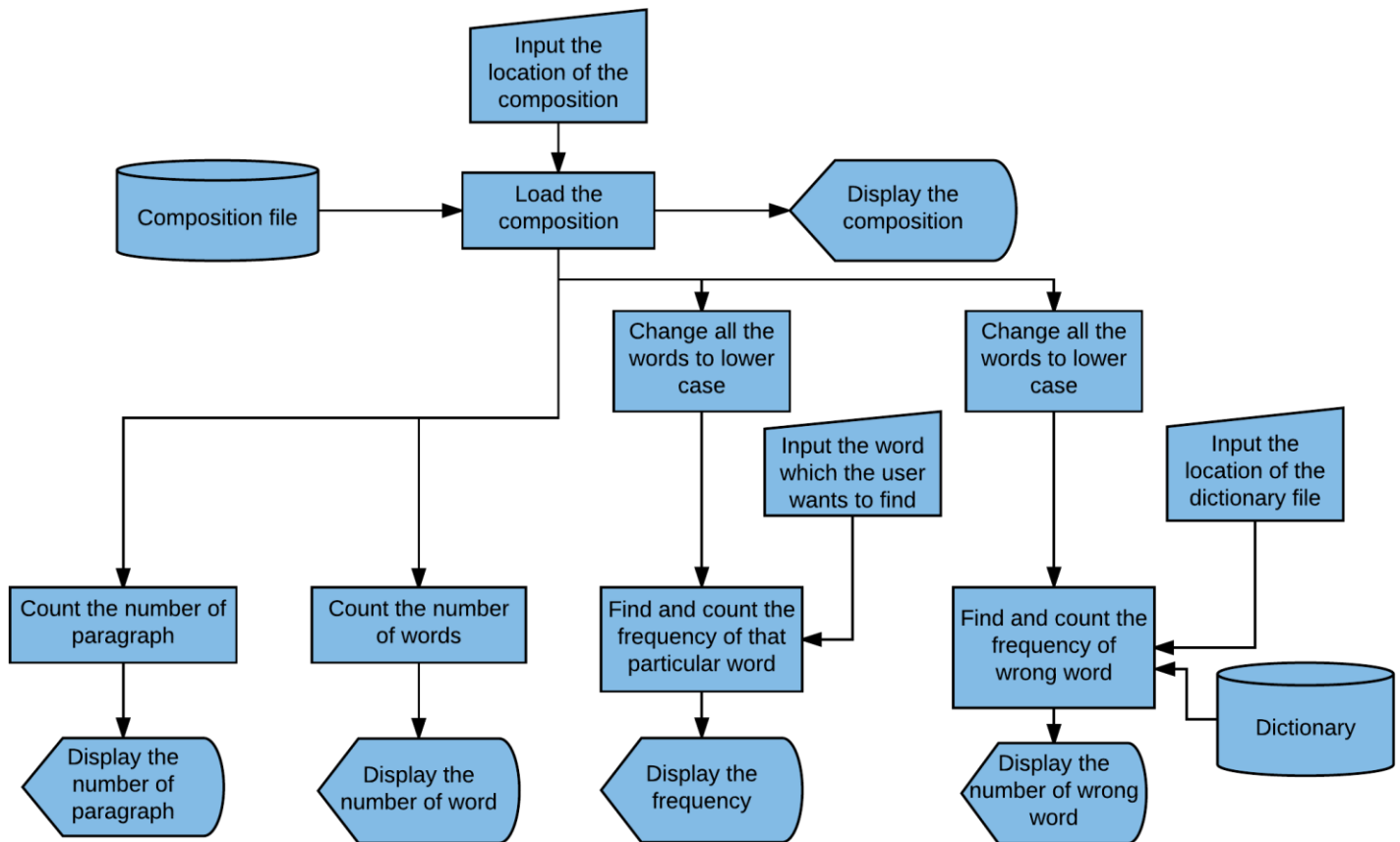
It may seems easy for a couple or two compositions, but imagine yourself as an English teacher – you need to grade about thirty compositions! It is an unmitigated problem if you continue to do the job manually.

For sake of getting rid of the troublesome, this software was developed to aid the English composition markers.

- Developed to aid English composition markers

### *Structure of the software*

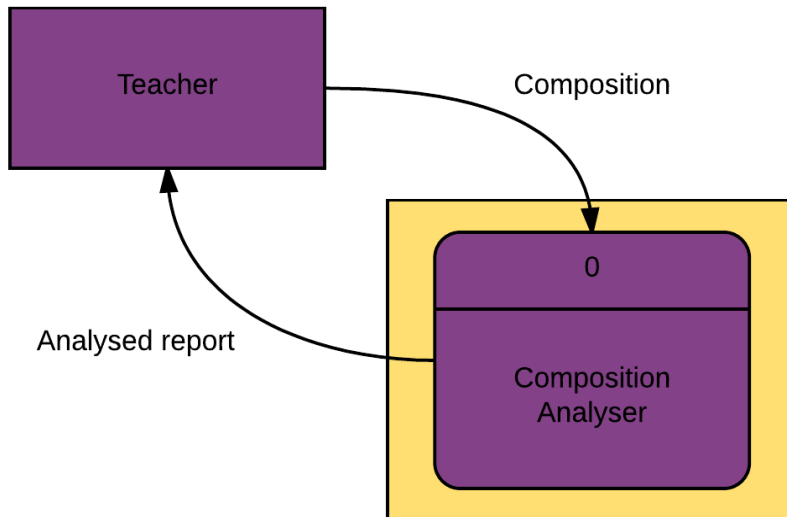
The following system flowchart illustrates the various process, files, database and associated manual procedures in this ‘English Composition Analyser’.



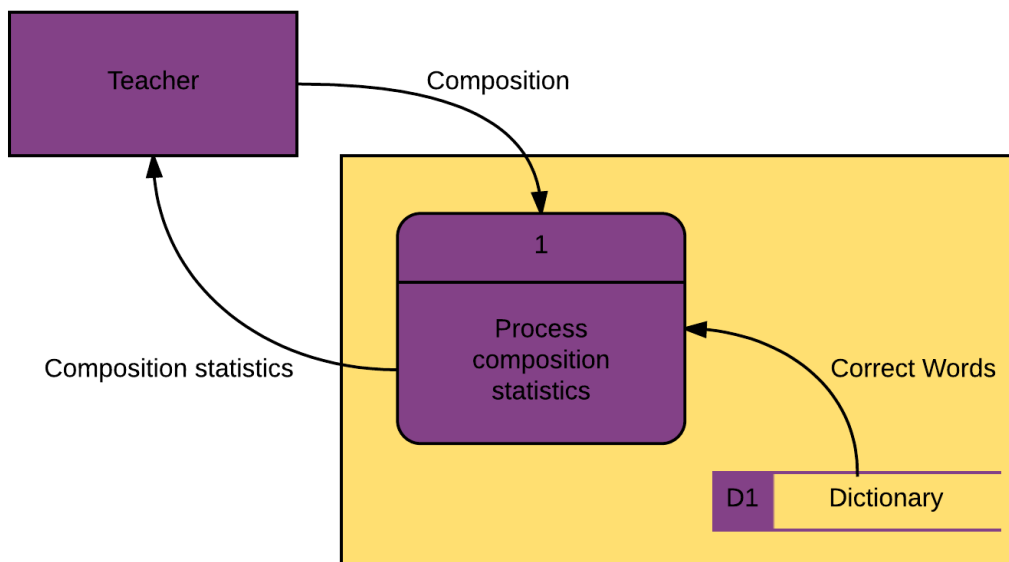
The ‘Composition file’ is the composition to be graded. It is in the format of text file, and is stored in the hard disk drive beforehand. Same as the ‘Composition file’, the ‘Dictionary’ is also in the format of text file and stored in the hard disk drive. It contains numerous words, and is used as a reference to check whether the words in the composition is correct in terms of spelling or not.

The following data flow diagram demonstrates all the main requirements of the software, namely the processes, the information or data flowing into, out of and within the software, and the way information or data is stored in the system.

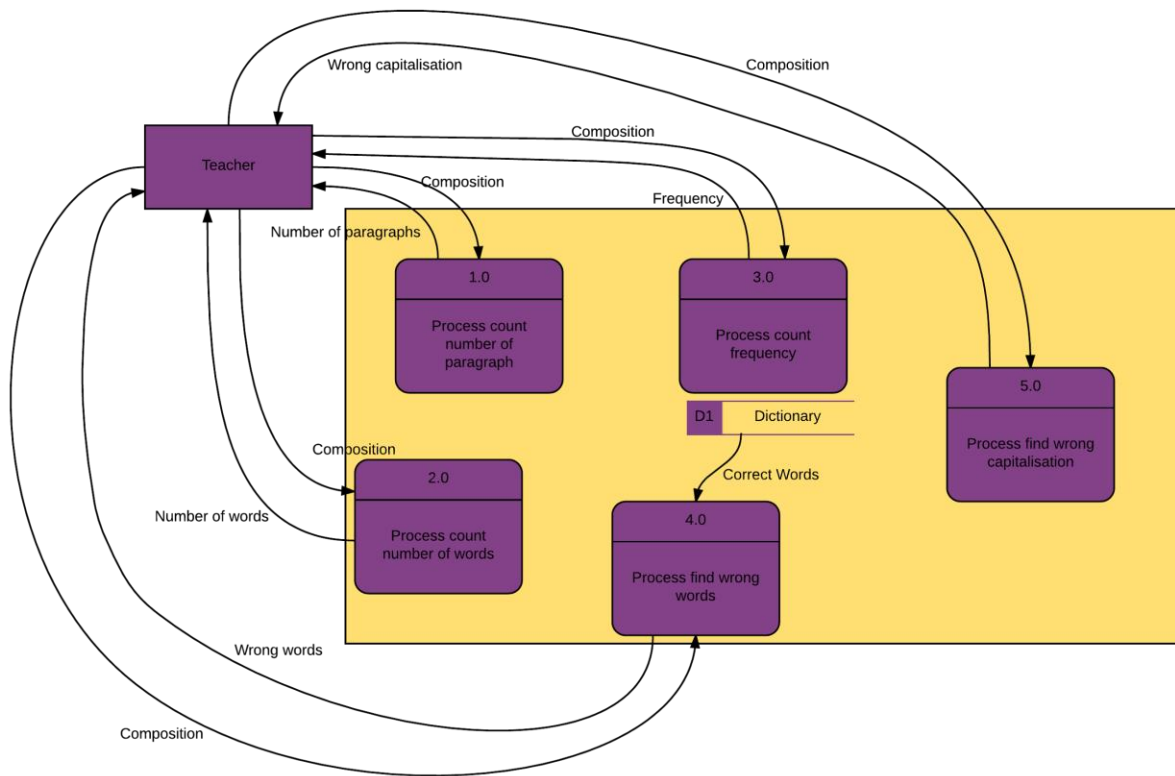
## LEVEL 0



## LEVEL 1



## LEVEL 2



As shown in the data flow diagram, there are a total of five process in this 'English Composition Analyser'. Their functions are count the total number of paragraphs, count the total number of words, count the frequency of a particular word or phrase, find the words which are spelt incorrectly, and find the letters which are capitalised wrongly or vice versa respectively.

### ***Statistical functions***

In this 'English Composition Analyser', there are a total of five statistical functions.

- 1.** Word Counter
- 2.** Paragraph Counter
- 3.** Frequency Counter
- 4.** Spelling Mistakes Finder
- 5.** Incorrect Capitalisation Finder

For the first statistical function, the ‘Word Counter’, it provides the user with the total number of words of the composition. This function is extremely useful as the statistic is of paramount importance in judging the quality of the composition. Without sufficient amount of words, it is certain that the composition cannot really express or illustrates anything meaningful. Hence, the grade is justifiable to be lower than average. Besides, according to the composition marking guidelines from the Hong Kong Education Bureau, the grade of compositions fewer than the required number of words cannot exceed ‘Level 4’. Thus, this statistical function is extremely useful in determining the grade of the composition.

For the second statistical function, the ‘Paragraph Counter’, it shows user the total number of paragraph in the composition. Same as the above one, this function is also overwhelmingly important in deciding the grade of a composition. It is because even with sufficient number of words, if the composition has too few paragraphs, it implies that the writer does not organise his ideas well. Therefore, the organisation marks would still be low. Thus, this statistical function is also useful in classifying masterpieces and poor works.

The third statistical function, the ‘Frequency Counter’, it is also undeniably useful in grading the composition. It is because with the use of famous idioms and quotes, the article will definitely sounds more persuasive and reasonable, and the score will be higher correspondingly. With this function, the marker can check whether there are any uses of idioms and quotes simply by typing the phrases into the appropriate place in this software. Thus, it is without doubt that this statistical function is very useful.

Apart from the above statistical functions, there is a ‘Spelling Mistakes Finder’ in this software too. This function can shows the user how many spelling mistakes were found in the composition with the aid of a dictionary containing numerous words. This function is especially useful because according to the marking guidelines, marks will be deducted whenever a wrong word was found. Therefore, this statistical function is essential when it comes to determining the actual grade of the composition.

Last but not least, the ‘Incorrect Capitalisation Finder’. This statistical function is also useful in grading a piece of composition. It is because even a piece of composition is very outstanding in terms of meaning, choice of words, and organisation, it cannot be treated as a masterpiece unless its capitalisation of letters are perfect too. Any incorrect capitalisation of letters will result in a deduction of the final score.

To conclude, the above five statistical functions are absolutely of supreme importance. It is unthinkable that any of these statistical functions being absent from the ‘English Composition Analyser’. Only with all these functions, the ‘English Composition Analyser’ could assist and provide the best grading experience for the marker.

### ***Special features***

Apart from the five essential statistical functions mentioned above, there are also several special features provided in this ‘English Composition Analyser’.

1. Spelling Mistakes Displayer
2. Grade Suggestion
3. The Clock
4. Comment
5. ‘Missing Composition’ Reminder
6. Progression Indicator
7. Report Preview
8. Report Generation

First of all, for the ‘Spelling Mistakes Displayer’, it will lists the words with spelling mistakes, and the sentences and paragraph which that particular word is from. This function can help the marker easier to locate the words with spelling mistakes and facilitate the marker to perform further actions.

The second feature is the ‘Grade Suggestion’ function. It will suggest a range of grades for the marker to determine the actual grade for the composition. This can be done because in the composition marking guidelines from the Hong Kong Education Bureau, there is a grade cap for compositions with lower than the required number of words. Moreover, as I mentioned above, the grade for compositions with only a few paragraphs or little to no use of idioms and quotes cannot receive high grades, and vice versa. Therefore, based on the statistics analysed by the five statistical functions, the software can roughly determines the range of grades of the compositions for the marker. This helps to save the marker’s precious time as the marker now have something to rely on.

The third among the total of eight features is ‘The Clock’. For ‘The Clock’, it seems like it is simply just a clock, but it does more than a common clock. It actually enables users to include the date and time of analysing the composition in the composition analytical report. This may seems useless but actually, this can help users to manage their time and work much better. This function is especially essential to users who need to analyse a batch of compositions. It is because with this feature, they can know exactly how much time they spent on a particular composition. So if they are lagging behind their schedule, they can speed up. Thus, this can effectively enables them to meet their targets and manage their work better. However, some people may overlook the importance of this feature. They think that

the operating system, such as Windows 7 or Mac OS, already provides a clock for the users. It is not that essential to add one more clock. But actually, as people always say, 'focusing too intently on one thing can make you blind to important details'. When the user is paying all the attention to this 'English Composition Analyser', the user may, indeed, overlooks the clock provided by the operating system as that clock is not located inside this software. Therefore, 'The Clock' is essential to users.

The fourth feature, 'Comment' enables users to give some comments and responds to the composition. The typed comments will then be printed in the composition analytical report. The user can open the report anytime to review the previous comments. This is a very meaningful and useful feature. It is because human's memory is limited. No one can remember every composition they read. However, a few words related to the composition can helps users to recall their memory. This helps to save time as users no longer need to read the composition once again. Instead, they just need to read a few lines of comments which they left in the first reading.

"Missing Composition" Reminder' is the fifth feature. What this feature does is that there will be a notification reminding the user to choose a composition (.txt document) after the user clicked the button to load a composition, and however, due to some unforeseen circumstances or mistakes, that user closes the file browser, leaving the software with no composition. This feature is useful, as it prevents the user from analysing nothing, so as not to waste the precious time of the user as well as prevents the user from getting some wrong and useless statistics.

Apart from the above features, the sixth one is the 'Progression Indicator'. This feature allows users to keep track of their own progression on analysing the composition. By getting more statistics, the user can increase the progression. A hundred per cent of progression rate means that the user has got all the statistics needed to grade the composition and also has given a comment on the composition. This is very helpful to users as it helps them not to go astray. It is because, with the indicator, they can know what to do next and how far are they from getting all the statistics they need to grade a composition.

For the seventh feature, 'Report Preview', it allows users to have a glimpse of the final composition analytical report before they actually save it. The user can then choose to amend the content of the report. For example, users can choose to include the date and time of analysing the composition in the analytical report. This feature can obviously let users have more control over the format as well as the content of the analytical report, allowing the software to be more user-friendly.

Last but not least, 'Report Generation' is the eighth feature. With this feature, users can generate a composition analytical report based on the analysed statistics and then save the report in the format of text documents (.txt). This feature is extremely important as it allows users to recall the previous analysed data and statistics of compositions without needing to analyse them once again. This undeniably stops users from wasting time on doing

meaningless things. Furthermore, this feature allows users to customise the name of the report. Thanks to that, users can classify different reports, so that users can know exactly which composition is the report related to, and not being mistaken. Therefore, the feature is essential to the software and is extremely important.

To conclude, the above special features are unbelievably useful and helpful to the user, and these certainly complements the original software and allowing it to become a user-friendly and comprehensive software.

## **Input**

### ***Plain text format***

Composition:

- Two 'space' ( ' ') marks the beginning of a new paragraph.
- No line is skipped between two paragraphs.
- Both uppercase and lowercase are acceptable.

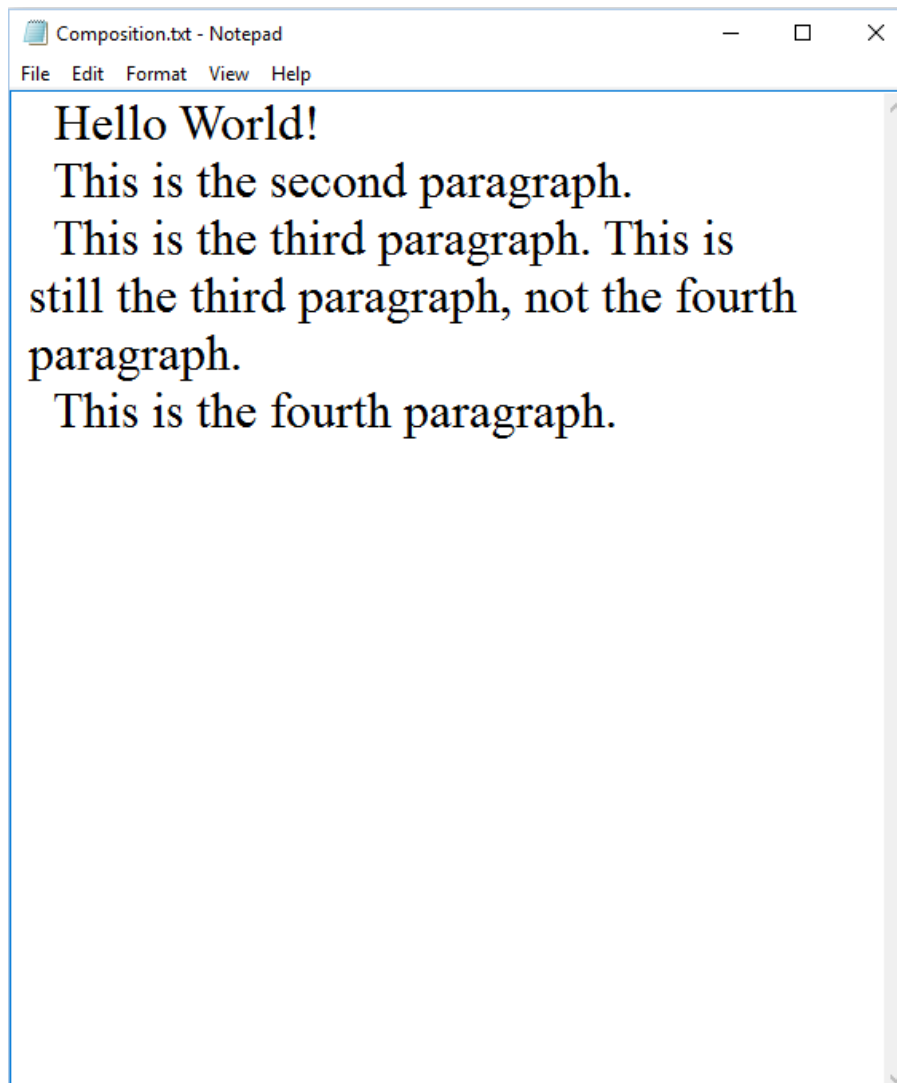
Dictionary:

- One single word per line. Hyphenated words count as one.
- No spaces before words.
- No line should be skipped.
- Both uppercase and lowercase are acceptable.

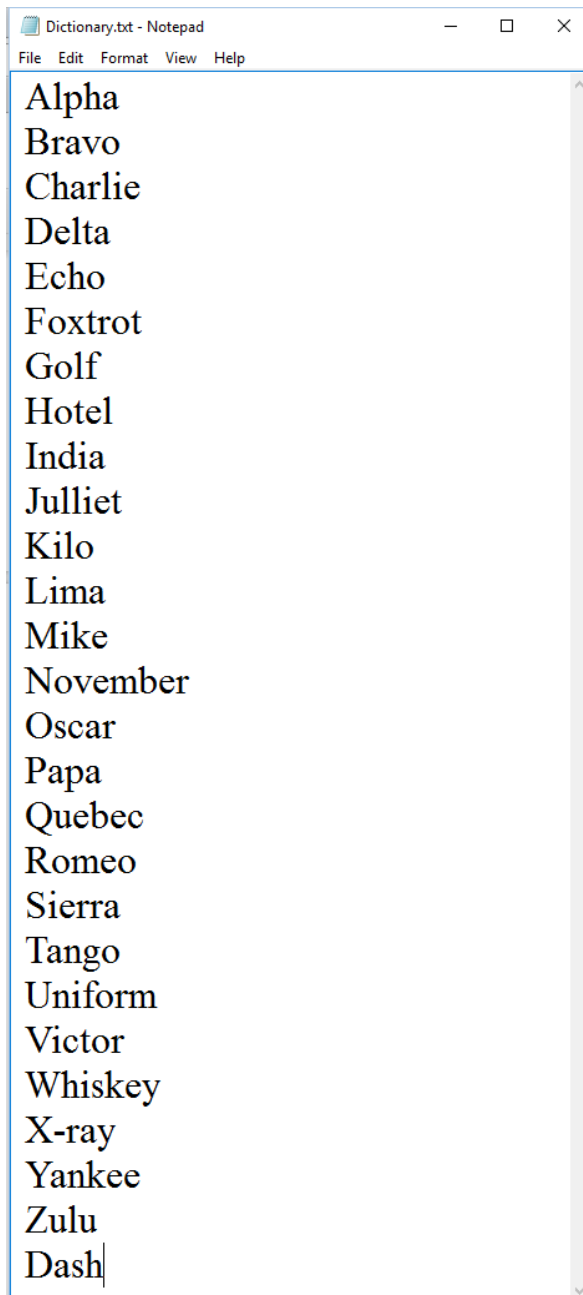
### ***Samples***

Composition:





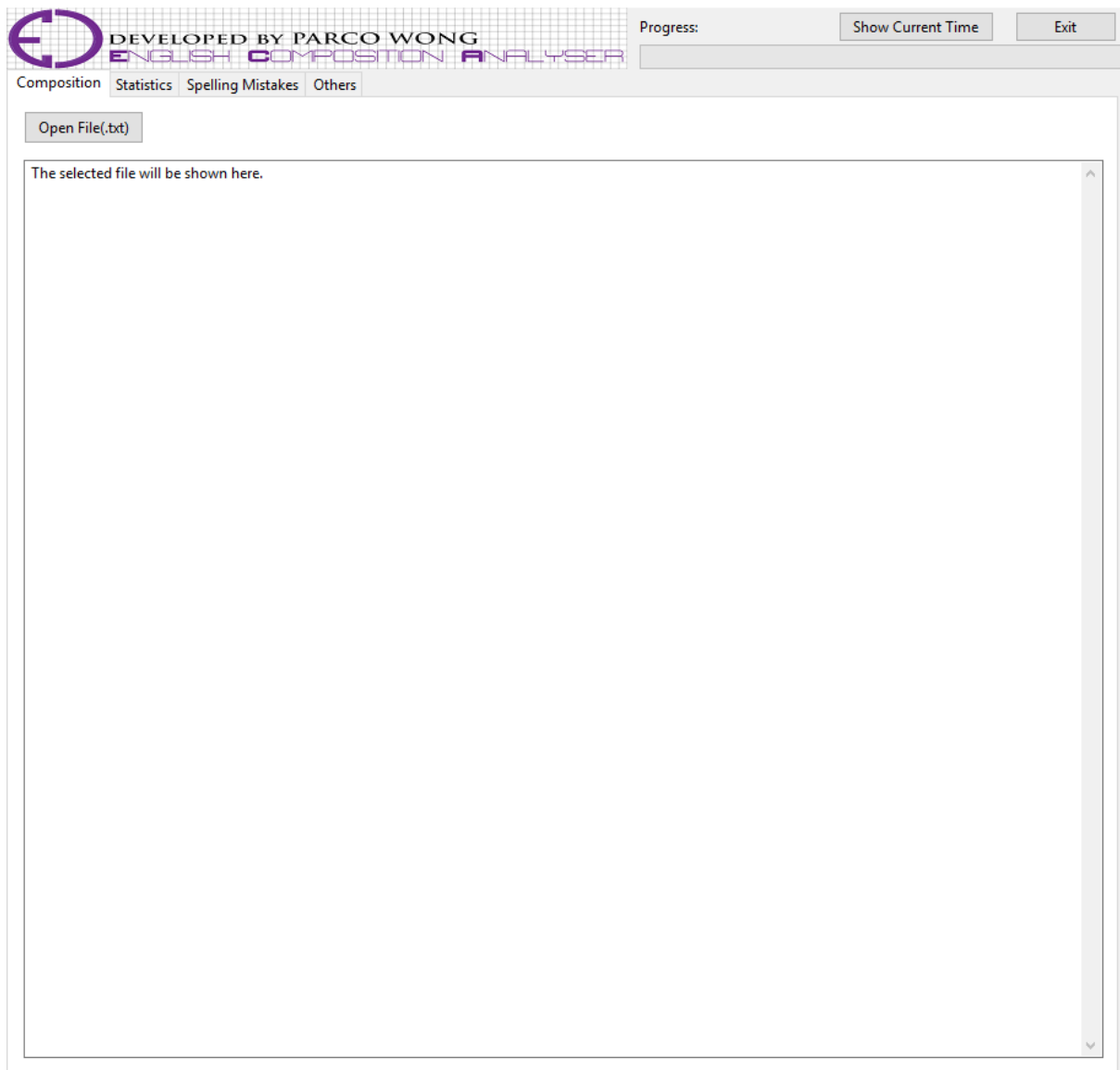
Dictionary:



## **User Interface**

There are a total of 4 pages in this 'English Composition Analyser'. Below are the screen captures of the four different pages.

Composition (first page):



Statistics (second page):

Number of Words:

Number of Paragraphs:

Frequency:

Type the word or phrase here.

Frequency Checker

Number of Spelling Mistakes:

Spelling Checker

(Click to open the dictionary .)

Number of Incorrect Capitalisations:

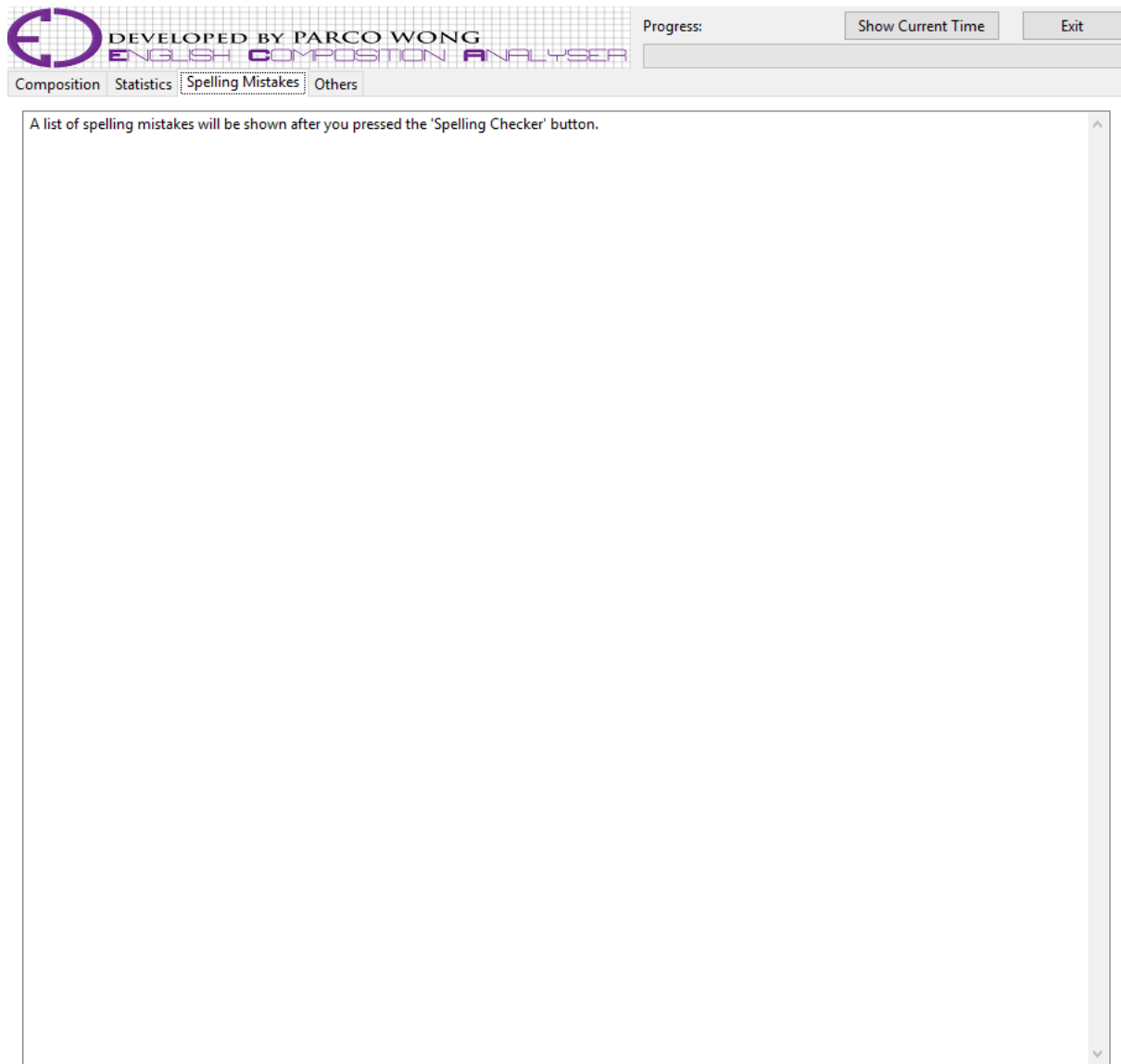
Capital Letter Check

Grade:


Grade Suggestion

---

Spelling Mistakes (third page):



Others (fourth page):



DEVELOPED BY PARCO WONG  
ENGLISH COMPOSITION ANALYSER

Progress: 

Show Current Time

Exit

Composition Statistics Spelling Mistakes Others

Type your comment here.

Add Comment

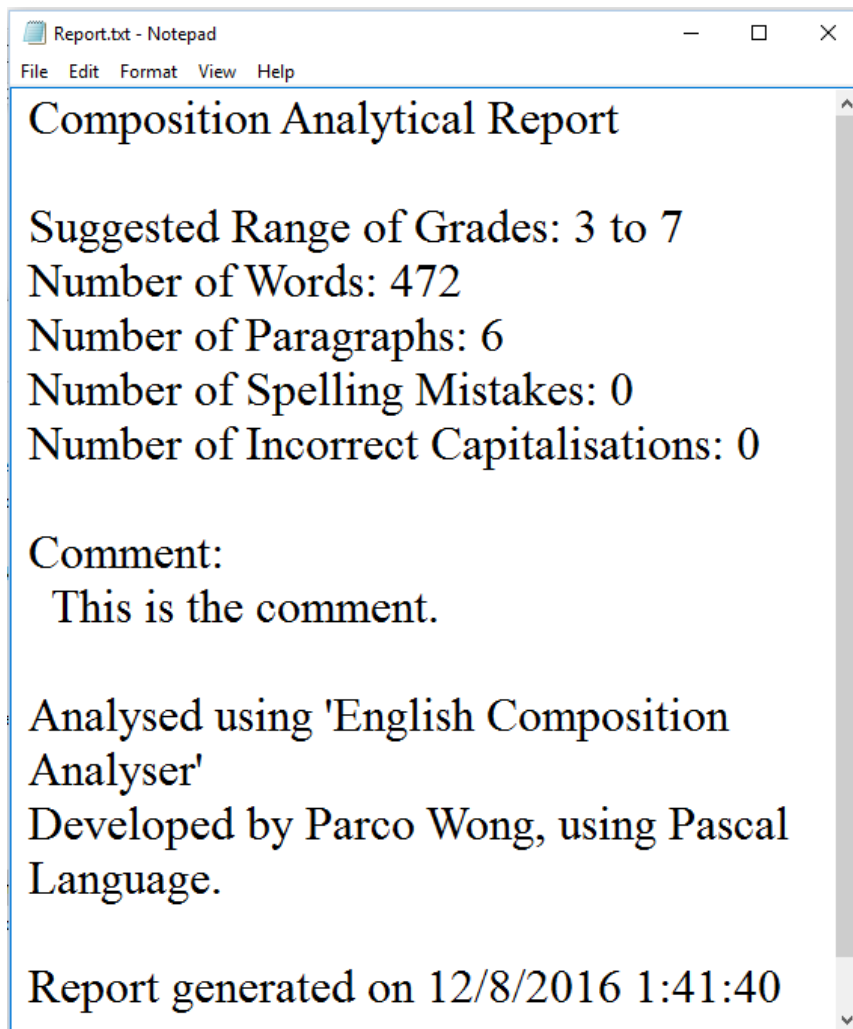
Report Preview Print Report ☒ Generate report with current time.

Report Preview

## Output

### *Sample report*

The following is a sample of the composition analytical report generated by pressing the 'Print Report' button in the 'Others' page. The report is in the format of text document (.txt).

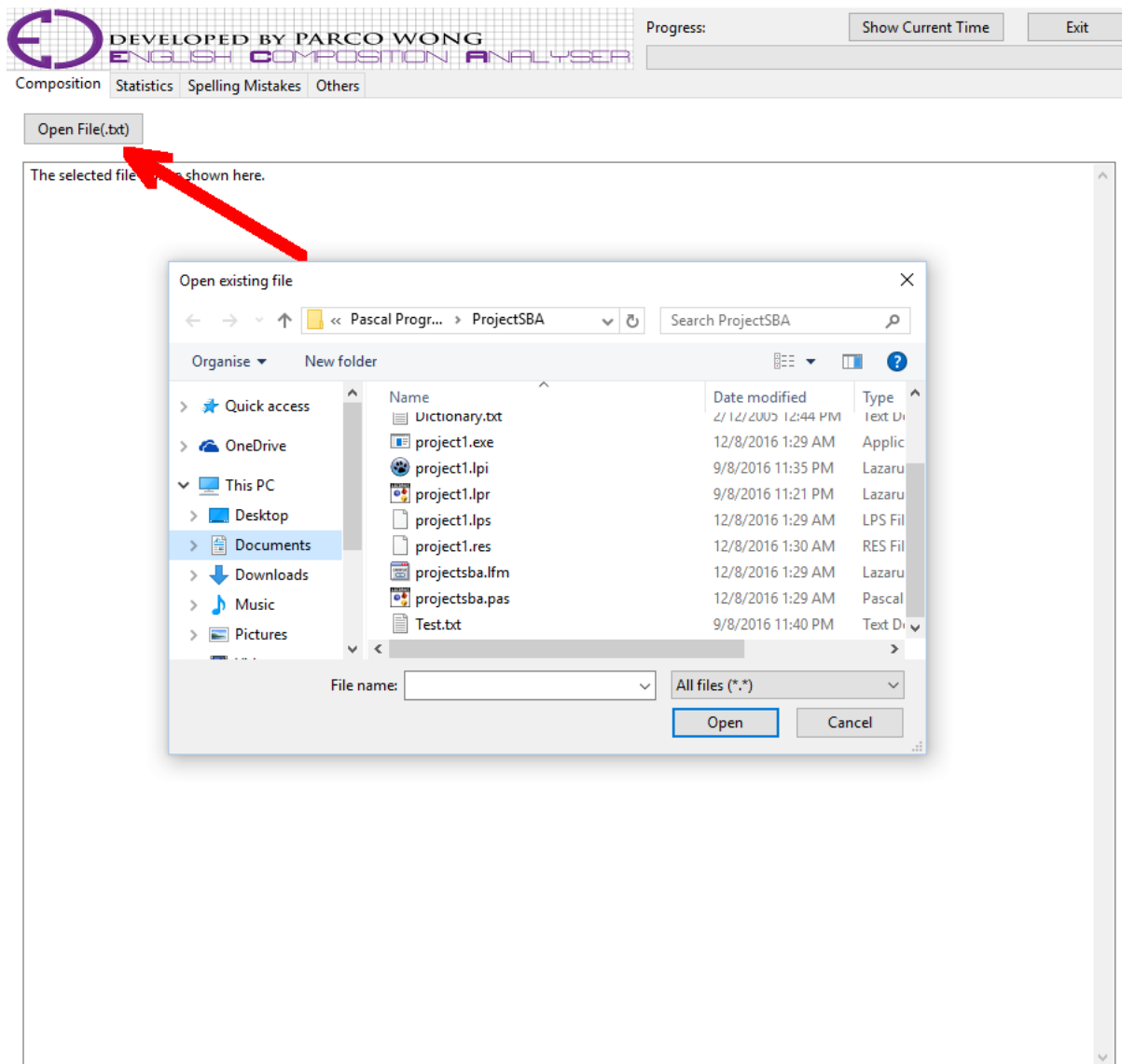


## **Implementation**

### ***Functions and features***

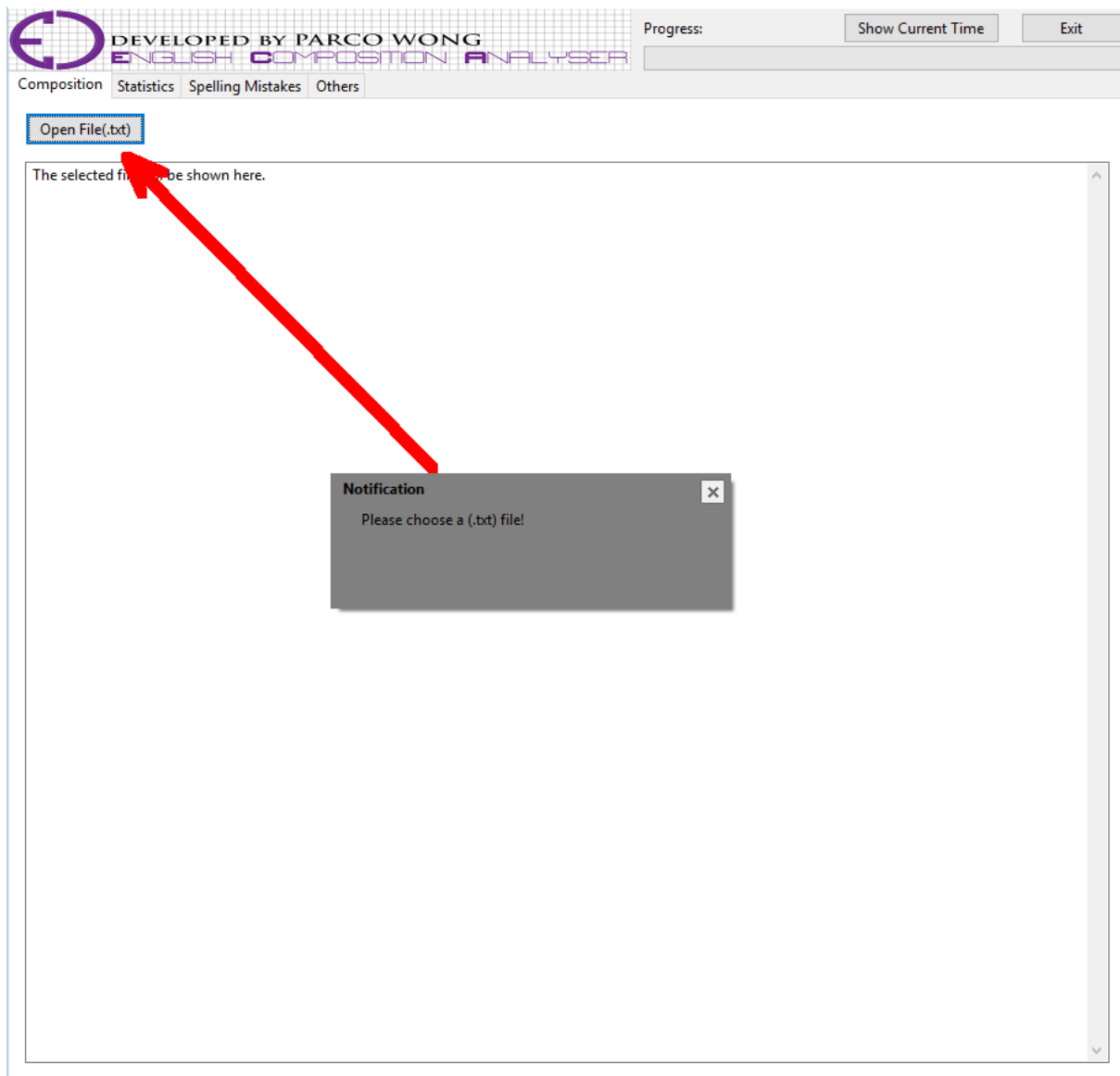
As mentioned above, this 'English Composition Analyser' has a total of 5 statistical functions and 8 special features. Below are the demonstrations of these 13 different functions and features, as well as some basic operations of the software.

Open File:




By clicking the 'Open File' button located on the 'Composition' page, the user can open a file browser so as to select the composition they want to load. For the composition, the file must be in the format of text documents (.txt).





However, if the user close the file browser without selecting any composition (.txt document), there will be a kindly notification for the user, reminding the user to choose a file before proceeding to further analysis.

DEVELOPED BY PARCO WONG  
ENGLISH COMPOSITION ANALYSER

Progress: 100%

Show Current Time

Exit

CompositionStatisticsSpelling MistakesOthers

Open File(.txt)

For the first statistical function, the Word Counter, it provides the user with the total number of words of the composition. This function is extremely useful as the statistic is of paramount importance in judging the quality of the composition. Without sufficient amount of words, it is certain that the composition cannot really express or illustrates anything meaningful. Hence, the grade is justifiable to be lower than average. Besides, according to the composition marking guidelines from the HongKong Education Bureau, the grade of compositions fewer than the required number of words cannot exceed Level Four. Thus, this statistical function is extremely useful in determining the grade of the composition.

For the second statistical function, the Paragraph Counter, it shows user the total number of paragraph in the composition. Same as the above one, this function is also overwhelmingly important in deciding the grade of a composition. It is because even with sufficient number of words, if the composition has too few paragraphs, it implies that the writer does not organize his ideas well. Therefore, the organization marks would still be low. Thus, this statistical function is also useful in classifying masterpieces and poor works.

The third statistical function, the Frequency Counter, it is also undeniably useful in grading the composition. It is because with the use of famous idioms and quotes, the article will definitely sounds more persuasive and reasonable, and the score will be higher correspondingly. With this function, the marker can check whether there are any uses of idioms and quotes simply by typing the phrases into the appropriate place in this software. Thus, it is without doubt that this statistical function is very useful.

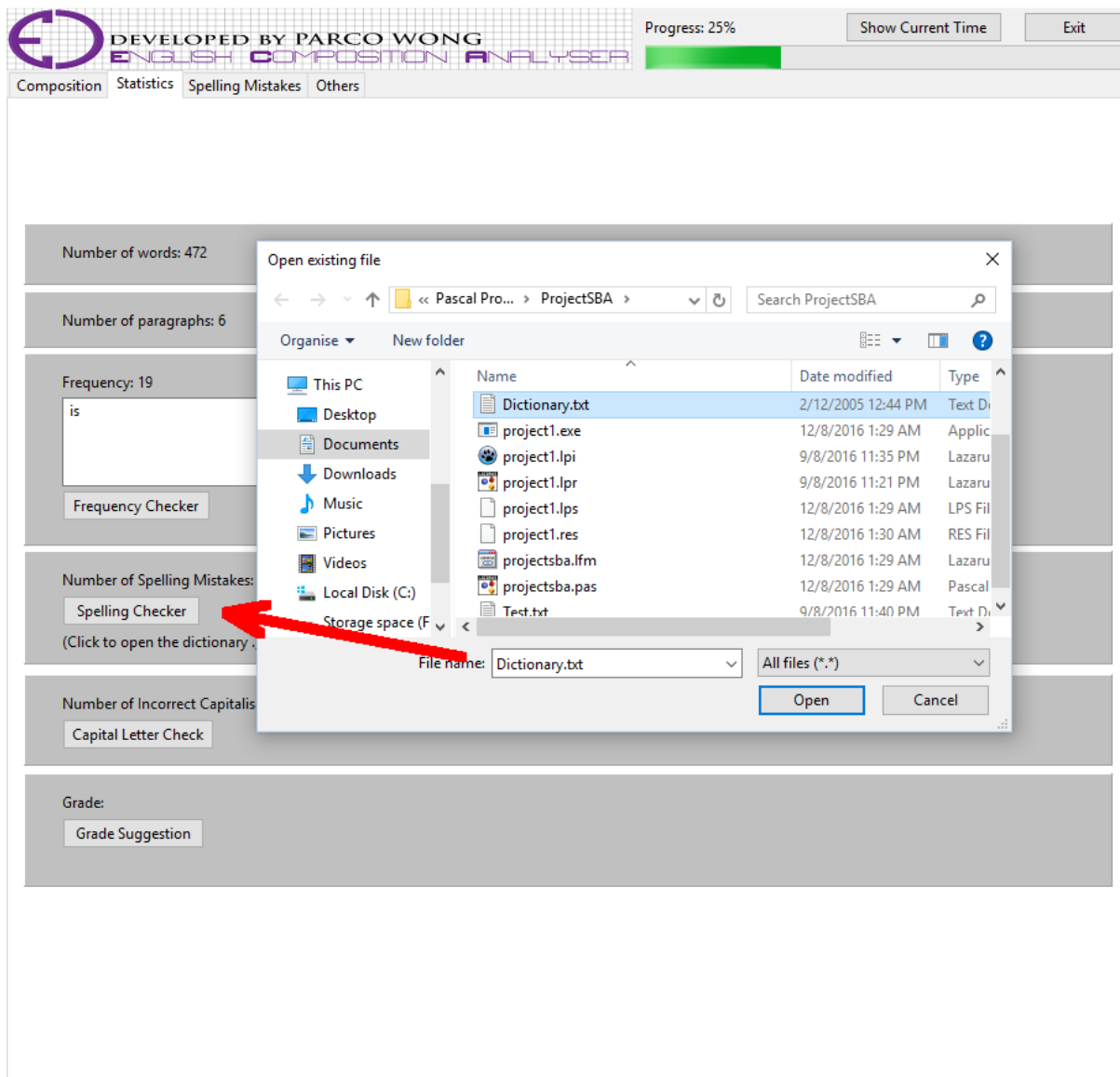
Apart from the above statistical functions, there is a Spelling Mistakes Finder in this software too. This function can shows the user how many spelling mistakes were found in the composition with the aid of a dictionary containing numerous words. This function is especially useful because according to the marking guidelines, marks will be deducted whenever a wrong word was found. Therefore, this statistical function is essential when it comes to determining the actual grade of the composition.

Last but not least, the Incorrect Capitalization Finder. This statistical function is also useful in grading a piece of composition. It is because even a piece of composition is very outstanding in terms of meaning, choice of words, and organization, it cannot be treated as a masterpiece unless its capitalization of letters are perfect too. Any incorrect capitalization of letters will result in a deduction of the final score.

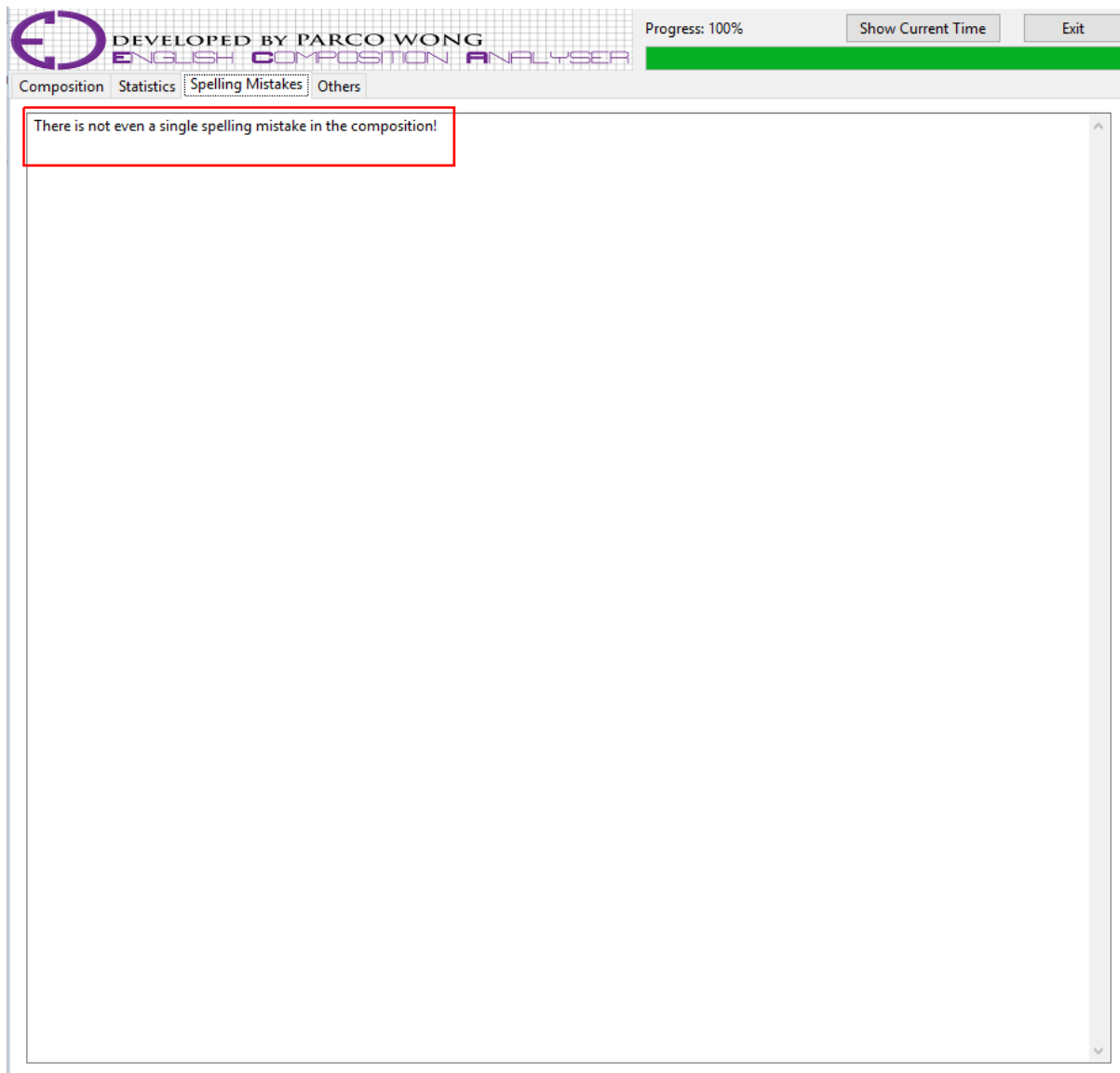
To conclude, the above five statistical functions are absolutely of supreme importance. It is unthinkable that any of these statistical functions being absent from the English Composition Analyser. Only with all these functions, the English Composition Analyser could assist and provide the best grading experience for the marker.

On the contrary, if the user selected a composition (.txt document), all the content of that composition will be shown in the box below the 'Open File' button, so that the user can read that composition.

Spelling Mistakes Checker:

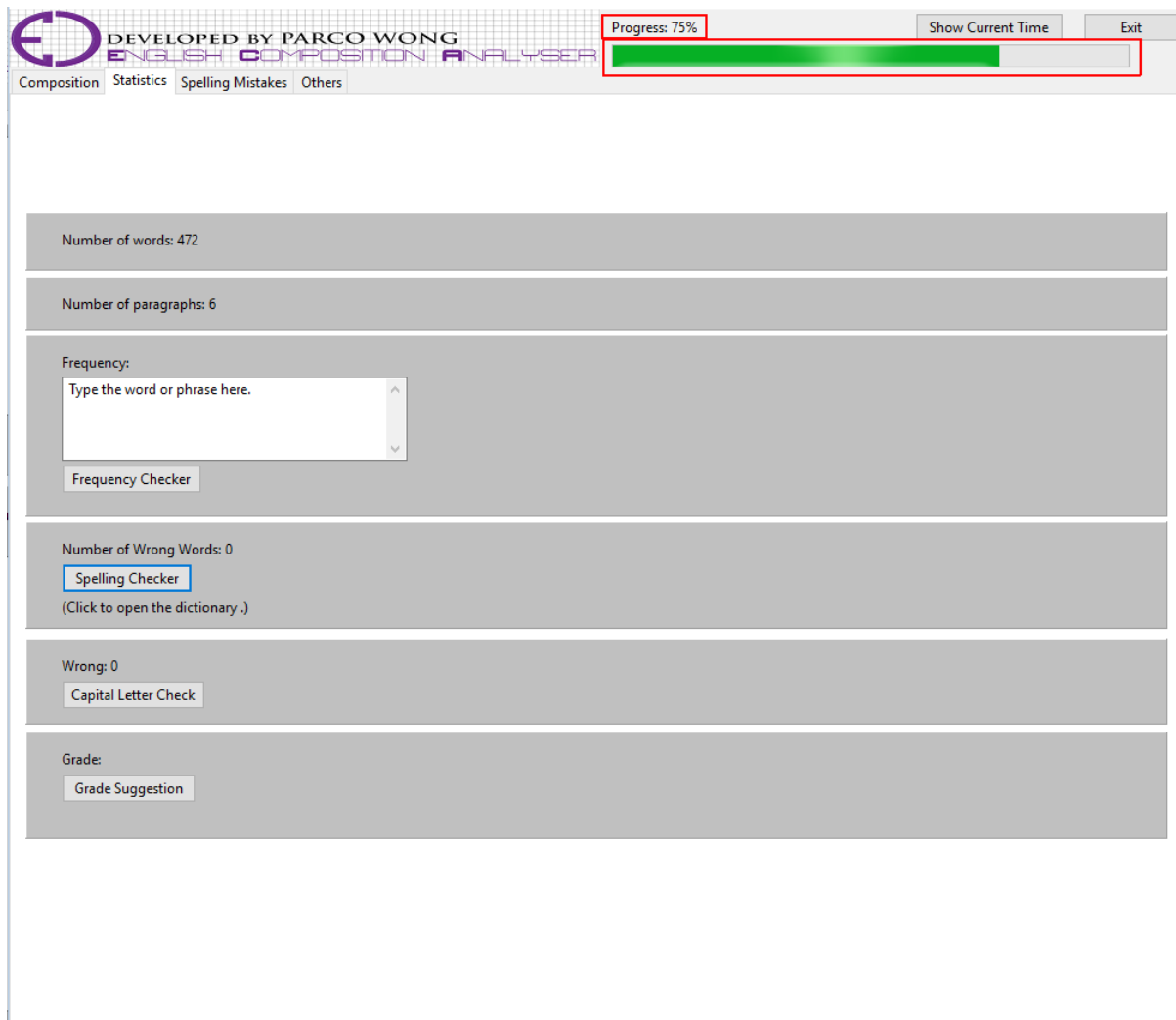


After selecting the composition (.txt document), the user can proceed to the 'Statistics' page. In this page, the user can check the number of spelling mistakes in the composition with the aid of the 'Spelling Mistakes Checker'. Simply by pressing the button, there will be a file browser popping out. The user can then select a dictionary for references. Same as the composition, the dictionary should also be in the format of text document (.txt). Moreover, if the user failed to select a dictionary (.txt document), again, there will be a kindly notification for the user, reminding the user to choose a dictionary (.txt document).

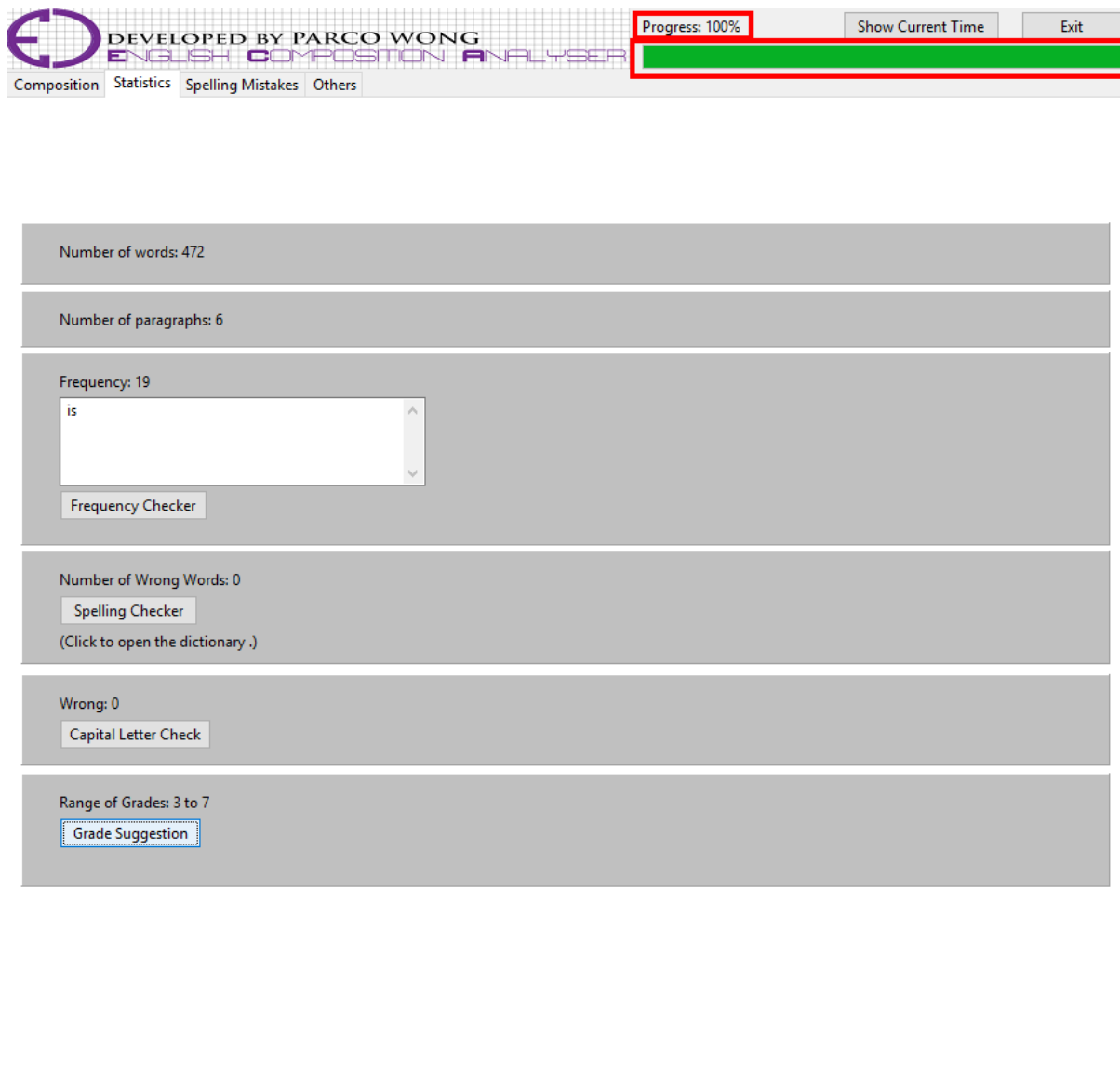


After choosing a dictionary (.txt document), the number of spelling mistakes will be shown above the 'Spelling Checker' button. Furthermore, a list of spelling mistakes will be shown under the 'Spelling Mistakes' page, in order to provide a more detailed analysis for the user.

Progression Indicator:

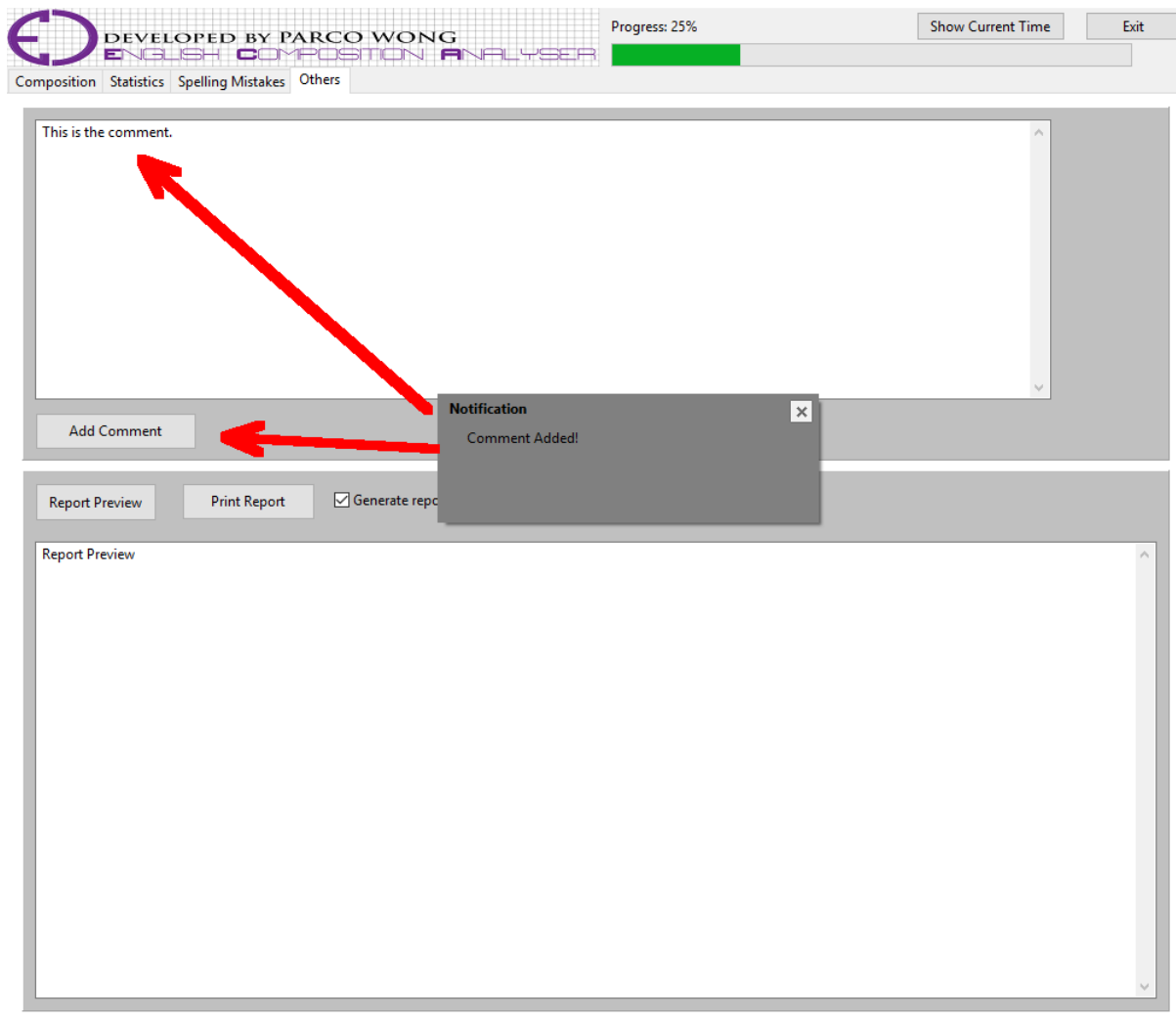


There is a progression indicator located on the upper-right corner of the software. On completing several analyses or using some particular special features, the rate of progression will increase.



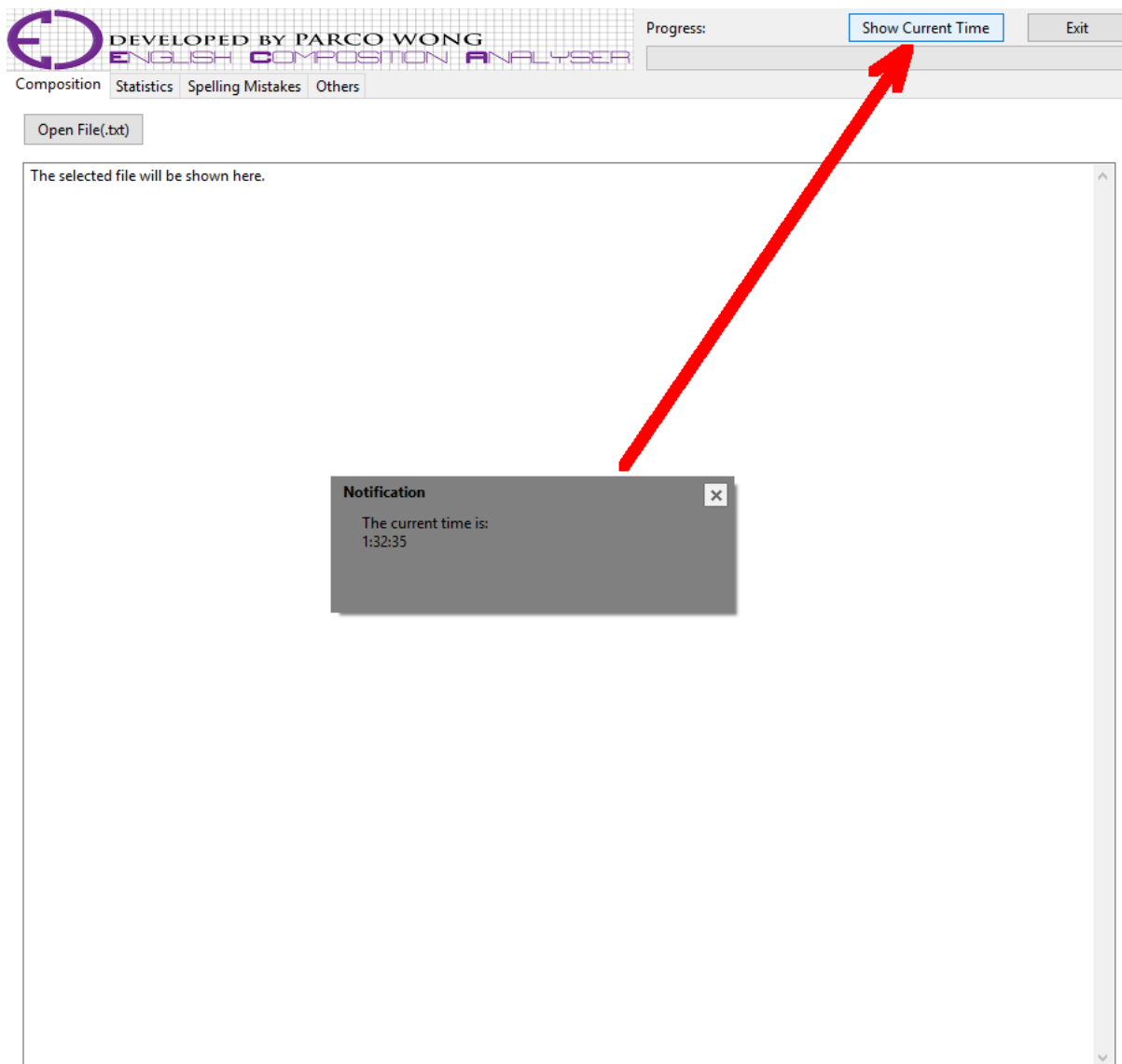
A hundred per cent of progression rate indicates that the user has used all analytical functions and several special features, and has got enough statistics and data to grade the composition.

Comment:



The user can leave a few lines of comment in under the 'Others' page. After typing the comment, the user can click the 'Add Comment' button to save the comment. After saving the comment, it will be printed in the composition analytical report – of course, the comment can be amended even after the user clicked the 'Add Comment' button. Simply by pressing the button once again, the comment will be updated.

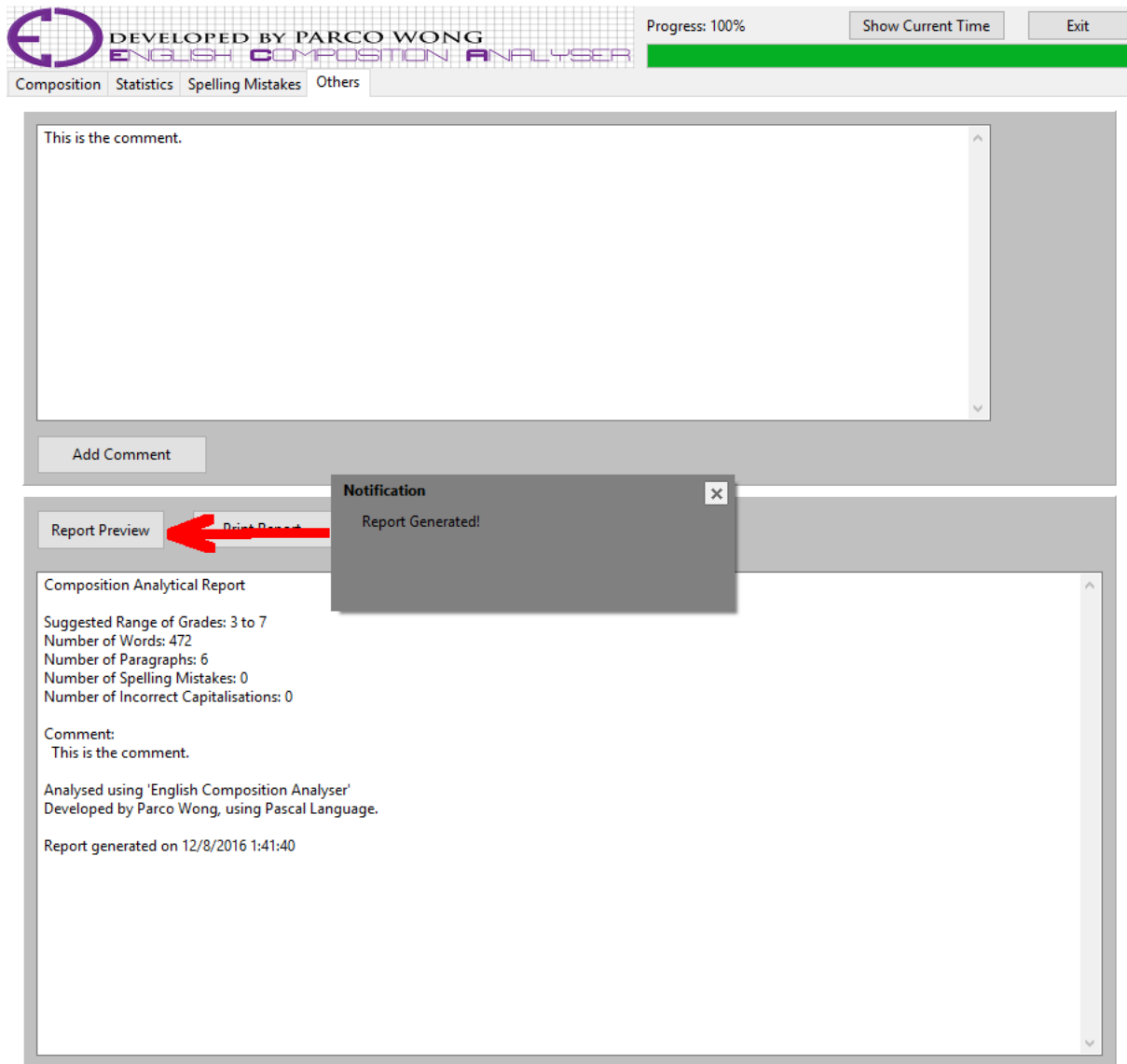
The Clock:



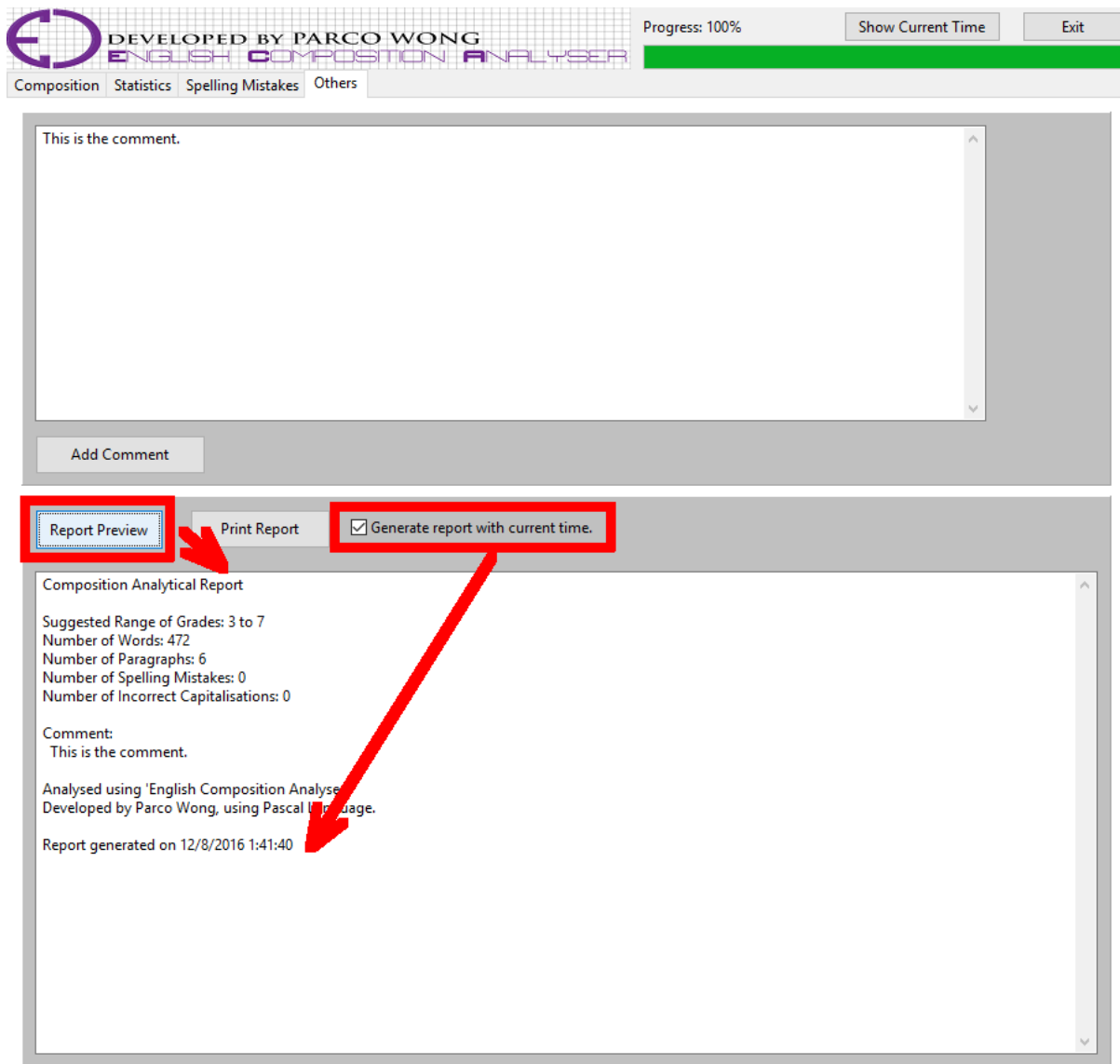
The time can be shown by clicking the ‘Show Current Time’ button. The time of day is written in the 24-hour notation in the form of hh:mm:ss, where hh is the number of hours, mm is the number of minutes, and ss is the number of seconds.

Report Preview:

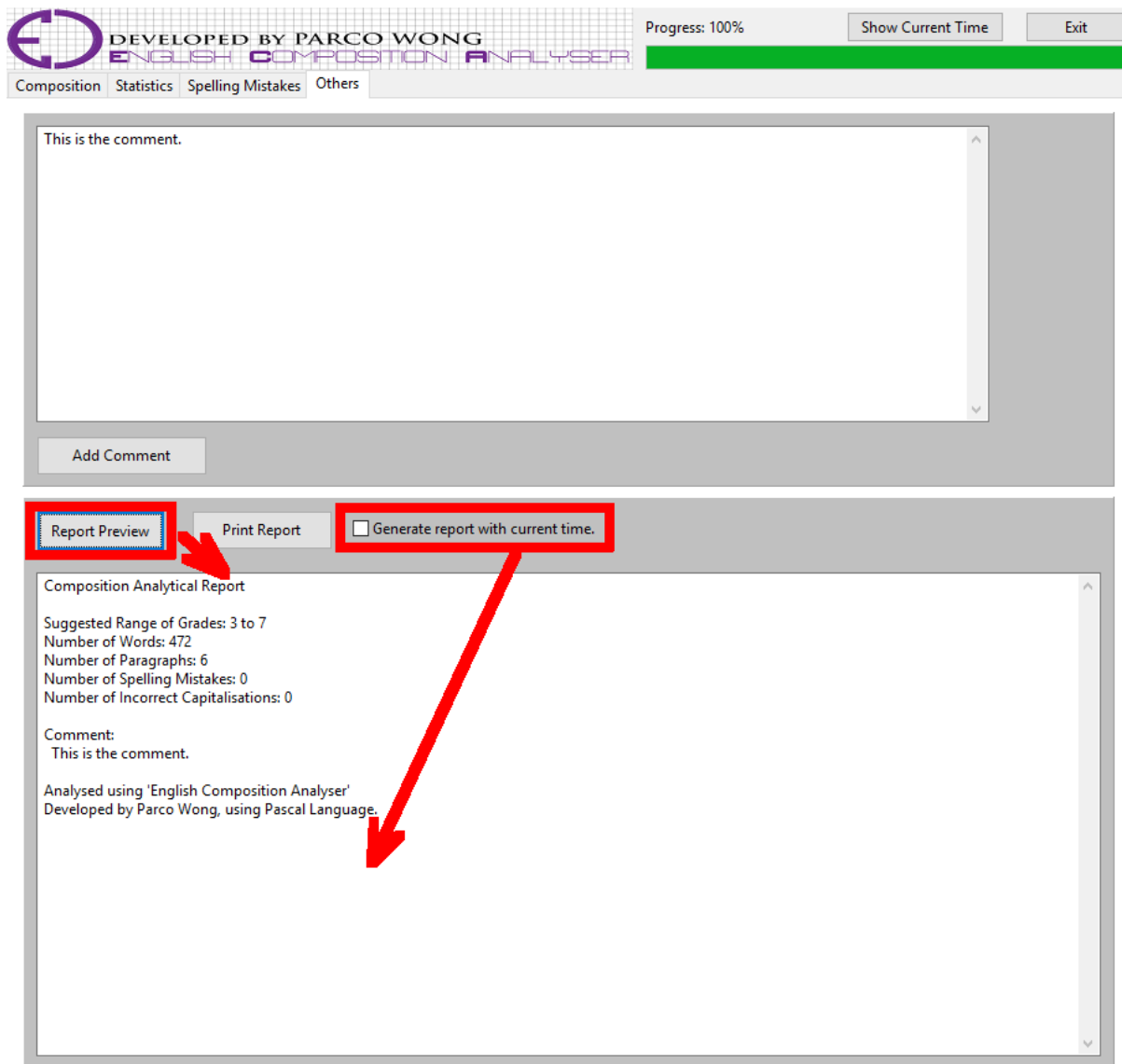




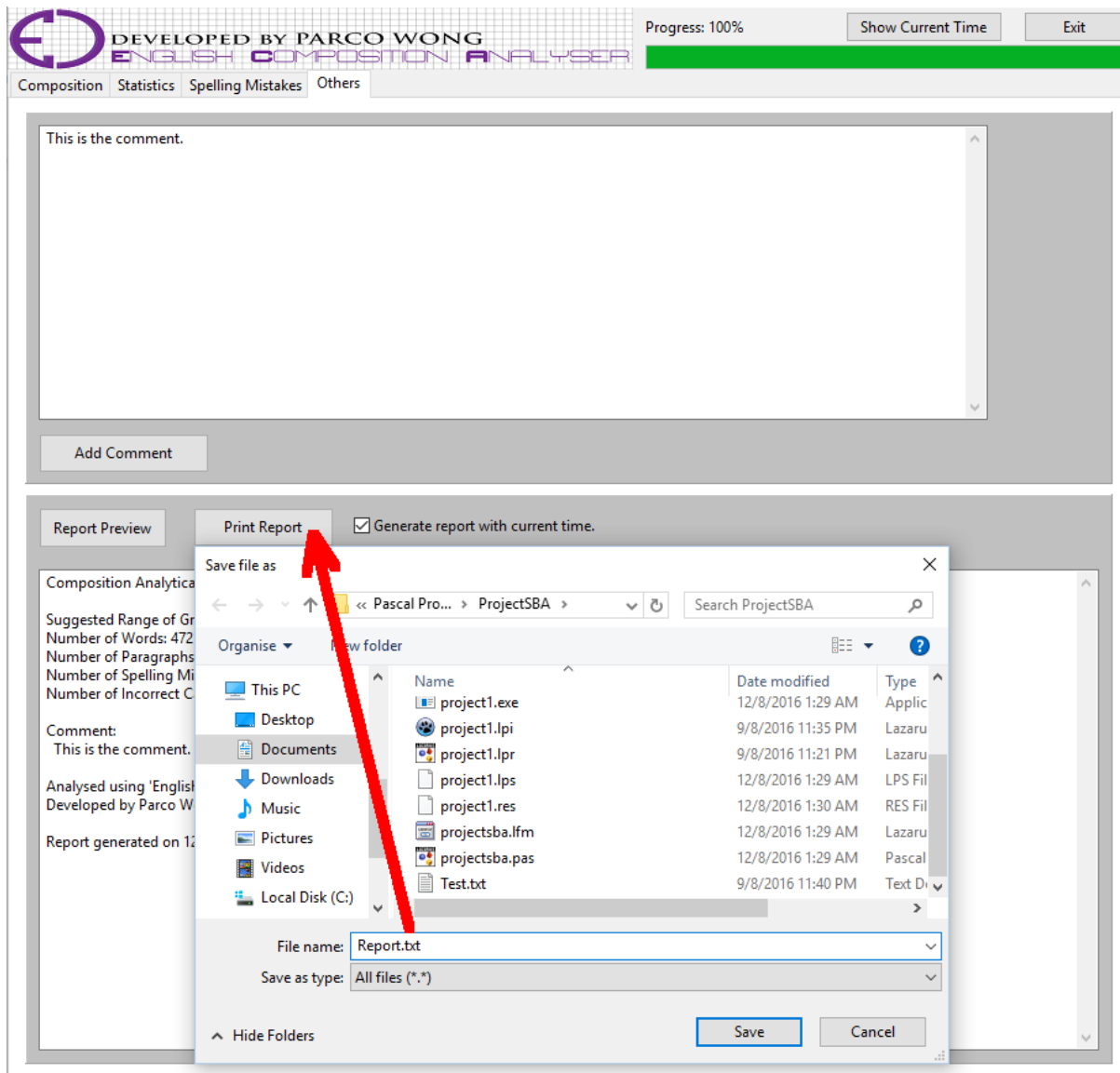
The user can preview the report to-be-printed by clicking the 'Report Preview' button. A notification telling user that the report has been generated will be popped after that.



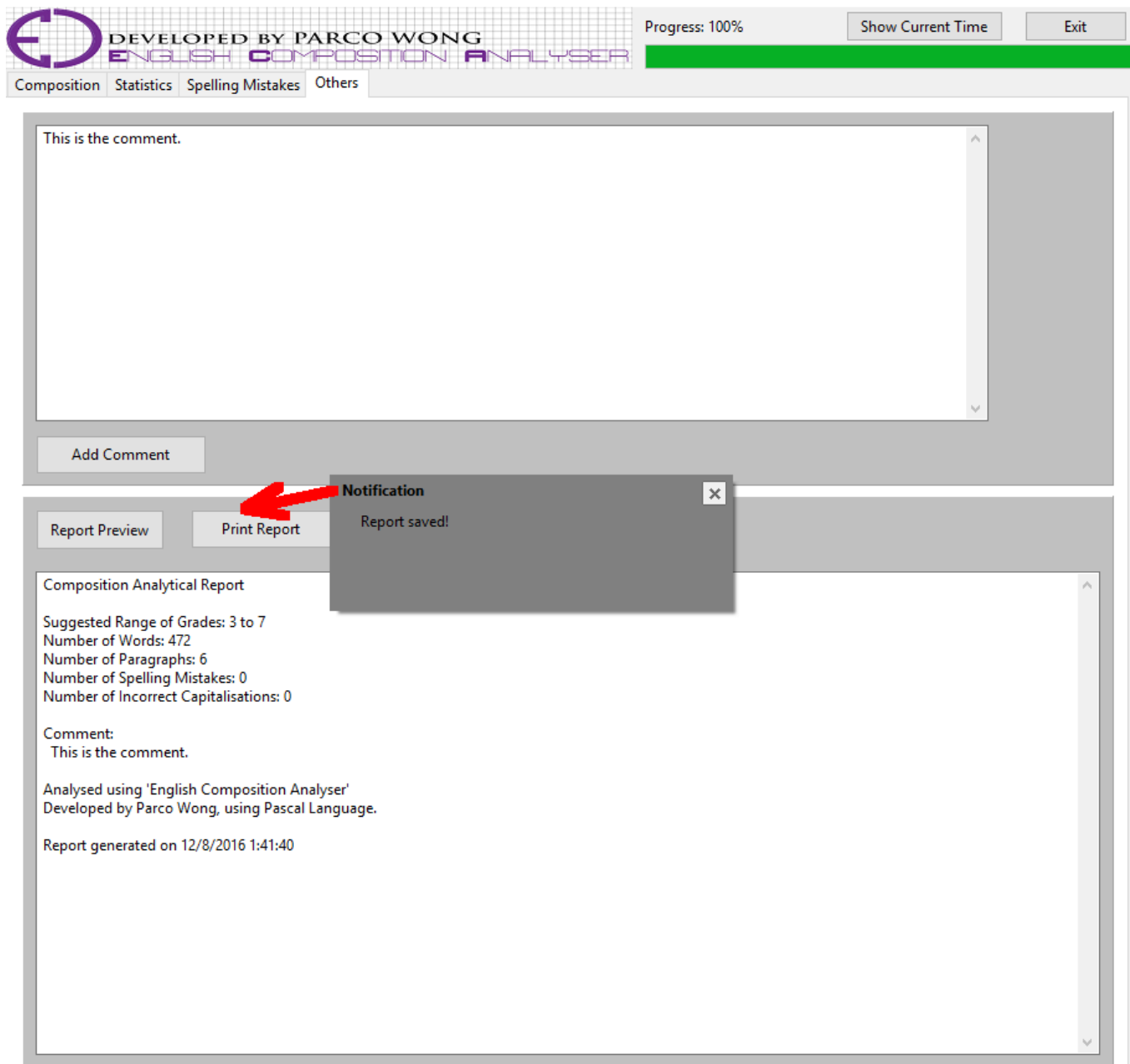
The user can choose to include the date and time of analysing the composition in the report by checking the box labeled as 'Generate report with current time'.



Of course, the user can also choose to uncheck the box so as not to print the date and time of analysing the composition in the composition analytical report.

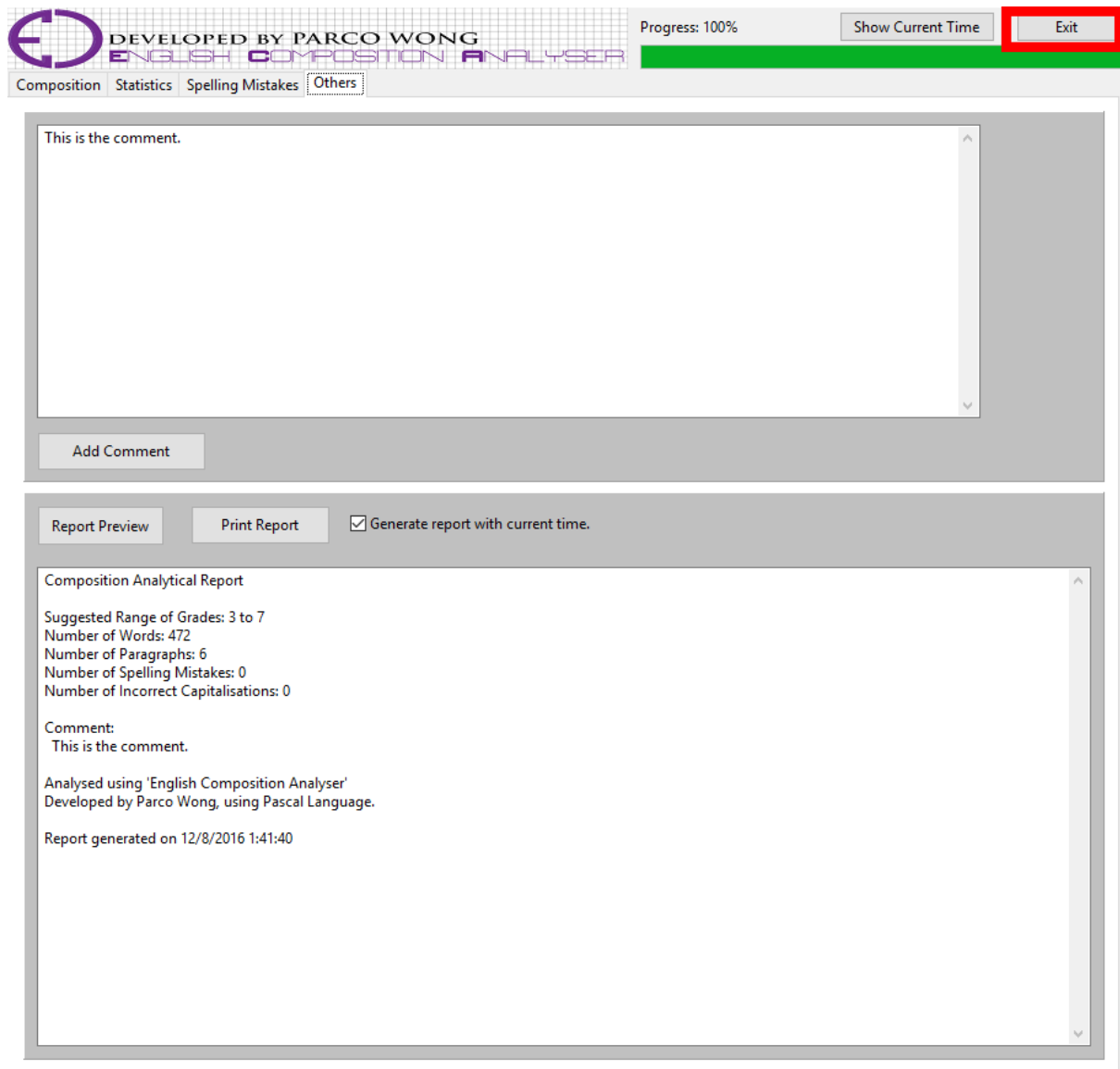


After previewing the report, the user can save that report in a text document (.txt) by clicking 'Print Report' button. After clicking the button, a file browser will be popped out to let the user choose where to save the report. The user can even name the report in accordance to their own will.



A notification will be shown after saving the report.

Exit:



Finally, after finish using this 'English Composition Analyser', the user can click the 'Exit' button to close the application.

### ***Data structures and algorithm***

Below, I will explain the algorithm used in the 'English Composition Analyser' in order to make the statistical functions and special features possible.

'Word Counter' and 'Paragraph Counter':

```
10  Number_of_Word <- 0
20  Number_of_Paragraph <- 0
30  if (open the composition file) then
40      filename <- file location;
50  End if
60  if (filename <> null) then
70      Display the composition
```

```

80      Open the composition for further process
90      hasComment <- FALSE
100     repeat
110         Load the composition line by line
120         Temp <- Loaded composition
130         check_para <- copy the first two units of Temp
140         if (check_para = two space) then
150             Number_of_Paragraph <- Number_of_Paragraph + 1
160         End if
170         repeat
180             Find the position of space in Temp
190             if (position of space = 0) then
200                 Number_of_Words <- Number_of_Words + 1
210             else
220                 Number_of_Words <- Number_of_Words + 1
230             End if
240         until (No more space in the sentence)
250     until (end of file)
260     Close file
270     Number_of_Words := Number_of_Words - Number_of_Paragraph * 2
280     Print the Number_of_Words
290     Print the Number_of_Paragraphs
300 else
310     Show the reminder telling the user to choose a file
320     Set the location of the notification
330 End if

```

In this algorithm, the program first load the composition line by line. Then copy the first two characters of the loaded line. This is very important as if that two characters are space, that marks the start of a new paragraph, so the counter for paragraph should add one. Then, the program finds the location of space in the sentence. It is because, in the algorithm, spaces are used to separate words. So whenever the program finds a space, the counter for the number of words is added one. However, even if there is no more presence of space, still, the counter for number of words needs to add one before the program load a brand-new line. It is because, for the last word of the paragraph, there will not be any space after the full stop.

The program will do the above process repeatedly until the end of the document. Then, the program will show the statistics in the user-interface.

#### ‘Frequency Checker’:

```

10  if (filename <> null) then
20      Open the composition for further process
30      word <- Lower-case version of the user-typed word
40      Initialise the Number_of_Words
50      repeat
60          Load the composition line by line

```

```

70      Temp <- Non-capitalised copy of the loaded document
80      repeat
90          Find the position of the user-typed word in Temp
100         if (the word is found) then
110             Copy the word
120             Copy the character before that word
130             Copy the character after the word
140             Delete all words before the located word in Temp
150             if (the word = the copied word, and there is no
other characters before and after the copied word) then
160                 Number_of_Words <- Number_of_Words + 1
170             End if
180         End if
190     until (No more space in the sentence)
200 until end of file
210 Close file
220 if (Number_of_Words <> 0) then
230     Print the Number_of_Words
240 End if
250 else
260     Show a reminder, telling the user to choose a file
270     Set the location of the notification
280 End if

```

In this algorithm, the program first convert the user-typed words as well as the composition into lower-case letters in order to make the following counting process easier to do. Then, the program search for the user-typed word in the composition. Whenever the program finds one, it will copy the preceding and succeeding character and checks if they are space or not. It is because, if they are space, that implies that the word is a stand-alone word, rather than just some characters inside another word, such as 'is' in the word 'this'. If the word is a stand-alone word, then, the counter for the frequency will add one.

The program will do the above process repeatedly until the end of the document. Then, the program will show the statistics in the user-interface.

#### 'Wrong Word Checker':

```

10  if (filename <> null) then
20      Open the file
30      WrongList <- Create a list of string
40      Initialise the Number_of_Paragraph
50      Initialise the Number_of_Correct_Words
60      Initialise the number of wrong words as the number of words.
70      if (Open the dictionary) then
80          filename2 <- Location of the dictionary
90      End if
100  if (filename2 <> null) then

```



```

110      Open the dictionary
120      repeat
130          Load the dictionary line by line
140          Temp <- Non-capitalised copy of the loaded document
150          check_para <- copy the first two words of Temp
160          if (check_para = two space) then
170              Number_of_Paragraph := Number_of_Paragraph + 1
180              Delete the first two space of the new paragraph
190          End if
200          repeat
210              Find the position of space in the sentence
220              if (the position <> 0) then
230                  Based on the position of space, copy the word before it
240                  Copy the last character of the word
250                  if (that character is a punctuation) then
260                      Delete the punctuation after the word
270                  End if
280              else
290                  Copy the remaining word of Temp
300                  Copy the last character of the word
310                  if (that character is a punctuation) then
320                      Delete the punctuation after the word
330                  End if
340              Open the dictionary
350              repeat
360                  Load the words in the dictionary
370                  Non-capitalised the dictionary.
380                  if (the word copied = the word in dictionary) then
390                      Number_of_Correct_Words := Number_of_Correct_Words + 1
400                  End if
410              until end of file(dictionary) or (the word copied = the word in
dictionary)
420              if NOT (word2 = word) then
430                  Add the wrong word into the WrongList
440              End if
450              Close file(dictionary)
460              Delete one word in Temp
470          until (No more space in the sentence
480          until end of file(composition)
490          Number_of_Wrong_Words <- Number_of_Wrong_Words -
Number_of_Correct_Words
500      Print the Number_of_Wrong_Words
510      Close file(composition)
520      if (Number_of_Wrong_Words > 0) then
530          Show the WrongList
540      End if
550      if NOT (clicked the button) then

```

```

560         Increase the progression indicator by 25%
570         Show the percentage
580         (clicked the button) <- TRUE
590     End if
600 else
610     Show a reminder, telling the user to choose a file
620     Set the location of the notification
630 else
640     Show a reminder, telling the user to choose a file
650     Set the location of the notification
660 End if

```

In this algorithm, the program will first initialise the number of wrong words as the total number of words. Then, whenever the program finds a correct word, the number of wrong words will deducts itself by one. The final result will then be the correct amount of wrong words.

So, in the program, it first finds the location of space, which is used for separating words. Then, it copies the words before the space. After that, the copied word will be used to compare with all the words in the dictionary (.txt document) one by one in accordance to the alphabetical order. If the copied word matches a word in the dictionary, that implies that the copied word is a correct word. So, the number of wrong words will deducts itself by one, as I mentioned above.

The program will do the above process repeatedly until the end of the document. Then, the program will show the statistics in the user-interface.

## **Testing and Evaluation**

### *Test plan*

Case	Target Function	Content	Expected Result (Passing Criteria)	Testing Objective
1	Word Counter	Hello World!	2	To test whether the software can count words or not.
2	Word Counter	HelloWorld!	2	To test whether the software can count words with no spacing or not.
3	Word Counter	Hello__World! (A ' _ ' is used to replace one 'space' for easier recognition)	2	To test whether the software can count words with two 'space' in

				between or not.
4	Word Counter	你好， 世界！	4	To test whether the software can count Chinese words or not.
5	Word Counter	Hello World! 1 2 3	5	To test whether the software can count numbers or not.
6	Word Counter	Halo World!	2	To test whether the software can count wrong words or not.
7	Paragraph Counter	__Hello World! __This is the second paragraph. (A ‘_’ is used to replace one ‘space’ for easier recognition)	2	To test whether the software can count paragraph starts with three space or not.
8	Paragraph Counter	__Hello World! _This is the second paragraph, (A ‘_’ is used to replace one ‘space’ for easier recognition)	2	To test whether the software can count paragraph starts with one space or not.
9	Paragraph Counter	__Hello World!__ (A ‘_’ is used to replace one ‘space’ for easier recognition)	1	To test whether the software will be misled by the paragraph ends with two space or more or not.
10	Paragraph Counter	__你好， 世界！ (A ‘_’ is used to replace one ‘space’ for easier recognition)	1	To test whether the software can count paragraph starts with Chinese characters or not.
11	Paragraph Counter	__Hello World!__This is the second sentence. (A ‘_’ is used to replace one ‘space’ for easier recognition)	1	To test whether the software will be misled by sentences separated by two 'space' in between or not.
12	Paragraph Counter	__Hello World! __This is the second paragraph. (A ‘_’ is used to replace one ‘space’ for easier recognition)	2	To test whether the software can count paragraph or not.
13	Frequency Counter	Hello World (Supposed the composition to-be-analysed contains two ‘Hello World’, three ‘the’, and one ‘The’)	2	To test whether the software can count frequency or not.
14	Frequency Counter	The (Supposed the composition to-be-	4	To test whether the software will treat words

		analysed contains two ‘Hello World’, three ‘the’, and one ‘The’)		which consist of the same characters but start with different cases as another word or not.
15	Frequency Counter	Hello World! (Supposed the composition to-be-analysed contains two ‘Hello World’, three ‘the’, and one ‘The’)	2	To test whether the software will treat the same phrase with a punctuation cohering after as another phrase or not.
16	Frequency Counter	Hello World!! (Supposed the composition to-be-analysed contains two ‘Hello World’, three ‘the’, and one ‘The’)	2	To test whether the software will treat the same phrase with more than one punctuations cohering after as another phrase or not.
17	Frequency Counter	tHe (Supposed the composition to-be-analysed contains two ‘Hello World’, three ‘the’, and one ‘The’)	4	To test whether the software will treat words which consist of the same characters but having different cases as another word or not.
18	Frequency Counter	HelloWord! (Supposed the composition to-be-analysed contains two ‘Hello World’, three ‘the’, and one ‘The’)	2	To test whether the software will treat phrases with same words but without spacing as another phrase or not.
19	Frequency Counter	你好，世界！ (Supposed the composition to-be-analysed contains two ‘Hello World’ , one ‘你好，世界!’ , three ‘the’ , and one ‘The’ )	1	To test whether the software can count Chinese words or not.
20	Spelling Mistakes Checker	Halo World	0	To test whether the software can count words with spelling mistake or not.
21	Spelling Mistakes Checker	hello World	0	To test whether the software will treat the word with different cases as wrong word or not.
22	Spelling Mistakes	Hello World!	0	To test whether the software will treat the

	Checker			word with a punctuation cohering after as wrong word or not.
23	Spelling Mistakes Checker	你好，世界！	0	To test whether the software will treat Chinese words as wrong word or not.
24	Spelling Mistakes Checker	HelloWorld!	0	To test whether the software will treat words without spacing as wrong word or not.
25	Spelling Mistakes Checker	Hello World!!	0	To test whether the software will treat the word with one or more punctuations cohering after as wrong word or not.
26	Incorrect Capitalisation Checker	__This is the first paragraph._this is the second sentence. (A ‘ _ ’ is used to replace one ‘space’ for easier recognition)	1	To test whether the software can find characters with incorrect capitalisation or not.
27	Incorrect Capitalisation Checker	__this is the first paragraph._This is the second sentence. (A ‘ _ ’ is used to replace one ‘space’ for easier recognition)	1	To test whether the software will treat the first character in a new paragraph with incorrect capitalisation as an incorrect capitalisation or not.
28	Incorrect Capitalisation Checker	__This is the first paragraph?_this is the second sentence!_this is the third sentence. (A ‘ _ ’ is used to replace one ‘space’ for easier recognition)	2	To test whether the software can find characters with incorrect capitalisation or not.
29	Incorrect Capitalisation Checker	__This is the First Paragraph._This is the second sentence. (A ‘ _ ’ is used to replace one ‘space’ for easier recognition)	2	To test whether the software can find characters with incorrect capitalisation or not.
30	Grade Suggestion	A composition with less than two hundred and fifty words.	Maximum grade: 4	To test whether the software suggest an appropriate grade or not
31	Grade Suggestion	A composition with exactly two hundred and fifty words.	Maximum grade: 7	To test whether the software suggest an appropriate grade or not

32	Grade Suggestion	A composition with more than two hundred and fifty words.	Maximum grade: 7	To test whether the software suggest an appropriate grade or not
33	Grade Suggestion	A composition with more than five spelling mistakes.	Maximum grade: 5	To test whether the software suggest an appropriate grade or not
34	Grade Suggestion	A composition with exactly five spelling mistakes.	Maximum grade: 5	To test whether the software suggest an appropriate grade or not
35	Grade Suggestion	A composition with less than five spelling mistakes.	Maximum grade: 7	To test whether the software suggest an appropriate grade or not
36	Grade Suggestion	A composition with more than five incorrect capitalisations.	Maximum grade: 5	To test whether the software suggest an appropriate grade or not
37	Grade Suggestion	A composition with exactly five incorrect capitalisations.	Maximum grade: 5	To test whether the software suggest an appropriate grade or not
38	Grade Suggestion	A composition with less than five incorrect capitalisations.	Maximum grade: 7	To test whether the software suggest an appropriate grade or not
39	Grade Suggestion	A composition with less than three paragraphs.	Maximum grade: 4	To test whether the software suggest an appropriate grade or not
40	Grade Suggestion	A composition with exactly than three paragraphs.	Maximum grade: 4	To test whether the software suggest an appropriate grade or not
41	Grade Suggestion	A composition with more than three paragraphs.	Maximum grade: 7	To test whether the software suggest an appropriate grade or not
42	Grade Suggestion	A composition with more than three paragraphs, but with more than five spelling mistakes.	Maximum grade: 5	To test whether the software suggest an appropriate grade or not.
43	Comment	你好，世界！	你好，世界！	To test whether the software can add a comment written in Chinese or not.
44	Comment	Hello World!	Hello World!	To test whether the software can add a comment or not.
45	Incorrect Capitalisation Checker	__This is the first paragraph.this is the second sentence. (A ‘_’ is used to replace one ‘space’ for	1	To test whether the software can find incorrect capitalisations

		easier recognition)		if there is no space before that character or not.
--	--	---------------------	--	--

## ***Evaluation***

After conducting the above tests, the overall performance of this ‘English Composition Analyser’ is unquestionably overwhelmingly impressive, with only little and acceptable bugs, and this software can utterly meet the objectives. Such conclusion is drawn because the software returns most of the results exactly the same as the expected results and the passing criterias.

Below, the bugs and failed tests will be brought to light.

First of all, the software cannot recognise words without space in between. This is reflected in Cases 2, 18, 24 and 45. This is due to the reason that the program algorithm uses space to separate words. Thus, if there is no space in between, the software will treat those words as one. Therefore, the software failed those tests. However, that is acceptable and justifiable as in reality, space is used to separate words. Thus, there is nothing wrong with the algorithm to recognise words based on the presences of space. It is a sheer absurdity ‘resolve’ this ‘bug’. If this software accepts concatenated words, and count them as two words, then how can it recognises the word ‘therefore’ from words ‘there’ and ‘fore’?

Secondly, the software cannot recognise Chinese words in some cases. This is reflected in Cases 19 and 23. In the ‘Frequency Checker’ and the ‘Spelling Mistakes Checker’, even though the inputted Chinese words are exactly the same as the words in the composition (.txt document), still, the software cannot recognise that, But for Case 4, the ‘Word Counter’ demonstrates it is capable of recognising Chinese characters. However, due to the fact that the four Chinese characters for testing only have one single space in the middle. In other words, two pairs of two concatenating Chinese characters. This misled the software, and therefore the software only count them as two words rather than four. This is because as I mentioned above, the algorithm uses spaces to separate words. Apart from that, the other test cases involving Chinese characters all passed the tests. This shows that actually, the software can recognise Chinese words. The reason why Cases 19 and 23 failed the tests is due to the fact that in those functions, the words inputted is used directly in calculations and processing. On the contrary, in Cases 4, 10 and 43, the Chinese characters are not directly used. The space before and after the characters are the things that are used directly indeed. Same as the above ‘bug’, this is also acceptable and justifiable. It is because for a normal text document, it only accepts words and characters in the format of ‘Unicode’. Which means, the composition actually does not allows Chinese characters. The text document containing Chinese characters are in fact generated by me with some special method (the method will not be discussed here). Thus, there is no need for the software to recognise Chinese characters. Moreover, this software is named the “‘English’ Composition Analyser’. Hence,

it is clear that Chinese character will never be used as there will not be any Chinese compositions.

Lastly, the software cannot recognise the incorrect capitalised letters which are not located after a full stop, exclamation mark or question mark with space. This is reflected in Cases 27 and 29. This is due to the reason that in the program, the algorithm only checks whether the character two units after a full stop, exclamation mark and question mark is capitalised or not. Therefore, incorrect capitalisation located at the start of a paragraph or only one unit after the full stop, exclamation mark and question mark cannot be found out. This bug is going to be resolved, and further details will be explained later, in the ‘Amelioration’ section.

- The overall performance of this ‘English Composition Analyser’ is impressive, with only little and acceptable bugs
- Reasons for failing several tests:
  - The software cannot recognise words without space in between
  - The software cannot recognise non-English characters in some cases
  - The software cannot recognise the incorrect capitalised letters which are not located after a full stop, exclamation mark or question mark with space
- However, the first and the second of the above reasons are acceptable, and are not really bugs
- Although the last reason is a loophole of the software, it is not a fatal and serious bug

## **Conclusion and Discussion**

### ***Amelioration***

Even though this ‘English Composition Analyser’ is already very comprehensive, still nothing in the world is perfect. Even the flagship softwares from famous world-leading enterprises have bugs and errors which need improvements and ameliorations. And so as this ‘English Composition Analyser’. Based on the above results of the ‘Testing and Evaluation’, I came up with three plans and proposal of amelioration.

First of all, the ‘Testing and Evaluation’ reveals that the ‘Incorrect Capitalisation Checker’ of this ‘English Composition Analyser’ does not check whether the first character of each paragraph is capitalised or not. This, in fact, makes the result of the incorrect capitalisation checking not as accurate as the users and I thought it would be. Therefore, the first improvement of the ‘English Composition Analyser’ should be solving this problem. This can be done in a simple way. In the algorithm, whenever a sentence or paragraph is loaded, the software will check whether that is the start of a new paragraph or not by checking if the first two characters are spaces or not. Then, if it is a new paragraph, the counter for counting the number of paragraph will be increased by one, and the first two



spaces will be deleted. The amelioration can makes use of this characteristic by adding a function to check whether the first letter is capitalised or not after deleting the two spaces. If it is not capitalised, then the number of incorrect capitalisation will increase itself by one.

The second amelioration also on the ‘Incorrect Capitalisation Checker’. Since the function currently only checks whether the character two units after a full stop, exclamation mark and question mark is capitalised or not. So the user never knows if the characters in the middle of sentences are wrongly capitalised or not. For example, in the sentence ‘I love playing computer Games.’, the letter ‘G’ should not be capitalised. However, the software cannot tell the difference between ‘I love playing computer Games.’ and ‘I love playing computer games.’ currently. Hence, the software should be improved to solve this problem. This correction is a little complicated, but it is feasible, First, after finding the incorrect capitalisations in the first character of each paragraph, rather than increasing the number of incorrect capitalisation, store the results in a temporary variable, and so as the incorrect capitalisations after the full stop, question mark, and the exclamation mark. Then, load the composition from the beginning once again. This time, except from the punctuations and spaces, check all characters, no matter they are they first character of a paragraph or the start of a new sentence or not, if they are capital letter or not. If it is capitalised, add one to the number of incorrect capitalisation. Finally, deducts the number of incorrect capitalisation with the value stored in the temporary variable. The final value in the counter for counting incorrect capitalisation will then be the correct value.

Lastly, the third amelioration is to make some amendments to the algorithm of the ‘Spelling Mistakes Finder’. It is because, currently, the algorithm is to copy words from the composition (.txt document) one by one, and compare them with all the words in the dictionary (.txt document). This method of searching is linear search, which all the words are searched in an alphabetical order. Although this may sounds great, actually, this makes the searching process too long and slow. It is in fact, redundant to search all words. Therefore, it should be improved. The method of improving is to use a different search method, so the software will not search all words one after another. Instead, the software will only finds words start with the same alphabet. For example, the software will only compares the words starting with ‘G’ if the word in the composition is ‘game’. Since the software no longer need to compares all the words in the dictionary to the words in the composition one by one, therefore, the speed is believed to be much faster than that of the current search.

- Ameliorations:
  - Check whether the first character of each paragraph is capitalised or not
  - Check whether the characters in the middle of the sentence are capitalised or not
  - Make the finding of spelling mistakes faster

### ***Further development***

Apart from the improvements pointed out in previous section, there are also a few possible further developments.

First of all, it is better to add a function to print the report on a piece of paper with the aid of a printer. This function can facilitate the distribution of the report. It is because, for example, after an English teacher use this 'English Composition Analyser' to analyse his or her students' composition, The teacher may wants to give the reports to students as reference. However, in a traditional school which does not have the practice of e-learning, it is hard to do so. Therefore, a 'Print' function is needed in order to facilitate the distribution of the report.

Secondly, it is also a good idea to include a function to suggest words to replace the original words with spelling mistakes. Currently, this software can only lists out all the wrong words, without suggesting any appropriate replacements. If this software can suggest some appropriate words to replace the words with spelling mistakes, it can saves the time of the user and greatly increase the efficiency of grading compositions as users no longer need to do that on their own.

- Further development:
  - Add a function to print the report on a piece of paper with the aid of a printer
  - Add a function to suggest words to replace the original words with spelling mistakes

## ***Conclusion***

As the above 'Testing and Evaluation' suggests, the overall performance of this 'English Composition Analyser' is unquestionably overwhelmingly impressive, with only little and acceptable bugs,

The 'English Composition Analyser', is a remarkable accomplishment, as I completed a lot of functions and features, All the statistical functions and special features mentioned above except the 'Incorrect Capitalisation Finder' was finished with an outstanding performance. They can be operated in almost all circumstances which will be encountered by users during their use of this 'English Composition Analyser' as reflected by the testing results. For example, the 'Words Counter', it is fully functionable and is able to count the number of words correctly and precisely no matter how many words are there in the composition. It can even counts the number of words of compositions which are not written in English if the format of the composition is appropriate. Furthermore, for the 'Paragraph Counter', it also performs very well. Same as the 'Word Counter', it is still functionable, and can provide precise statistics even when composition written in languages other than English is inputted into the software. Judging by the above examples, the software is truly an astonishing, and flawless accomplishment.

Even for the ‘Incorrect Capitalisation Finder’, still, it is usable and is basically finished and completed. It can find out all incorrect capitalised characters located after a full stop, exclamation mark or question mark with space. It just cannot check whether the first character of a new paragraph, is capitalised or not, and whether the letters in the middle of a sentence is in the correct case or not. Still, to a large extent, it is functionable. Therefore, it deserves a merit, and should be counted as one of the great accomplishments, but just not as outstanding and brilliant as the other statistical functions and special features.

To conclude, what I accomplished in this project is five statistical functions, namely the 'Word Counter', 'Paragraph Counter', 'Frequency Counter', 'Spelling Mistakes Finder', and the 'Incorrect Capitalisation Finder', and eight special features, namely the 'Spelling Mistakes Displayer', 'Grade Suggestion', 'The Clock', 'Comment', '"Missing Composition" Reminder', 'Progression Indicator', 'Report Preview', and 'Report Generation', with twelve of the thirteen accomplishments are achieved with an eye-popping impressive performance.

- The ‘English Composition Analyser’, is a remarkable accomplishment
- Most of the statistical functions and special features works perfectly well

## Reflection

In this project, I have learnt a lot of things.

First of all, I learnt to use charts and diagrams to present my ideas and designs. For example, I used the ‘Data Flow Diagram’ (DFD) to show the system components including processes, data flow, and storage. By using this diagram, my program logic is shown clearly, and it is easier for other people to understand what I mean. Moreover, in the project, I also used a ‘System Flowchart’ to show system components including processes, files, databases and associated manual procedures. Thanks to this high level view chart, I can show my program logic explicitly.

Secondly, I learnt to manage my time and work. It is because, while doing this project, I was required to complete everything on-time. Therefore, I tried to use a 'Gantt Chart' to carefully plan my schedule. Through repetition of using the chart, I finally master the skill of time-management, Therefore, by doing this project, my time-management skill is honed.

[illegible]

[illegible]

## **References**

1. **'New Senior Secondary Information and Communication Technology: Elective D2 (Software Development)', published by Pearson, 2015 Edition**
2. **'Software Development', (<http://hayathisolutions.com/software-development.aspx>), Retrieved 2016-06-07**
3. **'System Requirements Engineering', written by P. Loucopoulos and V. Karakostas, 1995**
4. **'Software Testing in The Real World', Edward Kit, (1992)**
5. **'Dynamics of Software Development', Jim McCarthy, (1995)**
6. **'Effective Software Project Management', Robert K. Wysocki, (2006)**
7. **'More About Software Requirements: Thorny Issues and Practical Advice', Karl E. Wiegers, (2005)**
8. **'Software Systems Development', (<http://www.onswaziline.com/services/software/index.php>), Retrieved 2016-06-07**

*~END~*