Linnéuniversitetet
Linnaeus University

Project Description

About the Project

This document outlines the text analysis project. You will create, either in pairs or by yourself, a comprehensive text analysis programme in Python that processes large files efficiently and visualises the results using Matplotlib. The programme will demonstrate file I/O, data structures, statistical analysis, and data visualisation concepts.

First, all the features of the application are described and detailed. You should read this, but at the end we will detail what is needed to reach different grades (meaning, you do not need to implement everything in order to pass the project).

Learning Objectives

By completing this project, you will:

- Work with large files using memory-efficient line-by-line reading
- Apply Python data structures (lists, dictionaries, sets, tuples) to solve real problems
- Create interactive menu-driven applications
- Generate statistical analyses of text data
- Visualise data using Matplotlib
- Practice modular programming with functions
- Handle user input and file selection gracefully

Project Requirements

Core Features

1. **Menu System**: Create a console-based menu with the following options:
    - Load a text file
    - Display basic statistics (with visualisation)
    - Show word frequency analysis (with visualisation)
    - Display sentence analysis (with visualisation)
    - Display character analysis (with visualisation)
    - Export results
    - Exit programme
2. **File Loading**: Implement a simple file selection system that:
    - Lists available `.txt` files in the current directory
    - Allows users to type a filename
    - Handles file not found errors gracefully
    - Processes files line-by-line (never load entire file into memory)

3. **Basic Statistics**: Calculate and display:
    - Total number of lines
    - Total number of words
    - Total number of characters (with and without spaces)
    - Average words per line
    - Average characters per word
4. **Word Analysis**:
    - Most common words (top 10)
    - Word length distribution
    - Unique word count
    - Words appearing only once
5. **Sentence Analysis**:
    - Average words per sentence
    - Longest and shortest sentences
    - Sentence length distribution
6. **Character Analysis**:
    - Letter frequency distribution
    - Punctuation statistics
    - Case distribution (uppercase vs lowercase)
7. **Visualisations** (using Matplotlib - integrated into each menu option):
    - Basic statistics: Bar chart of text composition + pie chart of character types
    - Word analysis: Bar chart of most common words + histogram of word lengths
    - Sentence analysis: Histogram of sentence lengths + bar chart of common lengths
    - Character analysis: Bar chart of most common letters + pie chart of character types
    - **Note**: Pie charts should use labelling that avoids overlapping text on small segments

Technical Requirements

Memory Efficiency

- **CRITICAL**: Never load the entire file into memory
- Process files line-by-line using a `for line in file:` loop
- Build statistics incrementally as you read each line

Data Structures Usage

- **Dictionaries**: For counting words, characters, etc.
- **Lists**: For storing sentence lengths, word lengths
- **Sets**: For tracking unique words
- **Tuples**: For pairing data (word, count) for sorting

Code Organisation

- Use functions to break down the problem into manageable pieces
- Pass data between functions using parameters and return values, that is, do **not** use global variables
- Separate data processing from visualisation
- Keep the main programme loop clean and readable
- Each function should have a single, clear responsibility

Text Files

You can use any text files that you like, but do not forget to test it for *large* text files. We recommend that you visit sites like Project Gutenberg, Project Runeberg (Swedish) or similar to find free books in text format. You *need* to test your program with at least *one* text file that is larger than one megabyte. When using books from the previously mentioned sites, it can be a good idea to remove the "copyright" message (that basically says that it is free) in the beginning and the end of the book.

Load the text file into a capable text editor and use that to check the number of words there are in the book. Your program should come close to this, but five percent more or less is quite acceptable.

Grading

This project is graded A-F. It does not matter if you work on your own or in pairs, the project will be graded the same (so, you might actually benefit from doing it in pairs). In addition to functionality, as described below, the code standard will also be a factor when grading (see "Code Organisation" above).

Grades D-E

For passing the project, the project must be able to list all text files in the current directory as well as loading one. In addition, the program needs to be able to show basic statistics number of lines, words and characters. Statistics need to be visualised using Matplotlib. In essence, a minimum viable product.

Grade C

In addition to what is needed for grades D and E, a project for grade C needs to show analysis for at least one of "word", "sentence" and "character" (see example output below) and visualise these using Matplotlib. The statistics also need to be exportable, that is, exported to a file that compiles all the produced statistics.

Grade B

For grade B, the project needs to implement all the above plus the analysis not covered by grade C (that is, all of "word", "sentence" and "character" analysis), complete with visualisations and inclusion in the exported summary of the statistics.

Grade A

For the top grade, the project needs to add a useful feature to the project. This could be (but not limited to) readability scores (Flesch Reading Ease or Lix), comparative analysis between text files, N-gram analysis (find common 2-word or 3-word phrases).

Example Execution of the Program

This shows the output (without the visualisations) of an example implementation of the program. Your program can, but does not need to, look similar.

```
Welcome to the Text Analyser!
This programme analyses text files and provides statistical
insights.

==================================================
           TEXT ANALYSER
==================================================
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
No file loaded

Enter your choice (1-7): 1

--- File Selection ---
Available text files:
  1. crime_and_punishment.txt
  2. The_Shadow_over_Innsmouth.txt
  3. dracula.txt
  4. requirements.txt

Enter filename or number from list above: 1

Analysing "crime_and_punishment.txt"...
Analysis complete! Processed 22060 lines.
Successfully loaded and analysed "crime_and_punishment.txt"

==================================================
           TEXT ANALYSER
==================================================
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
Current file: crime_and_punishment.txt
```

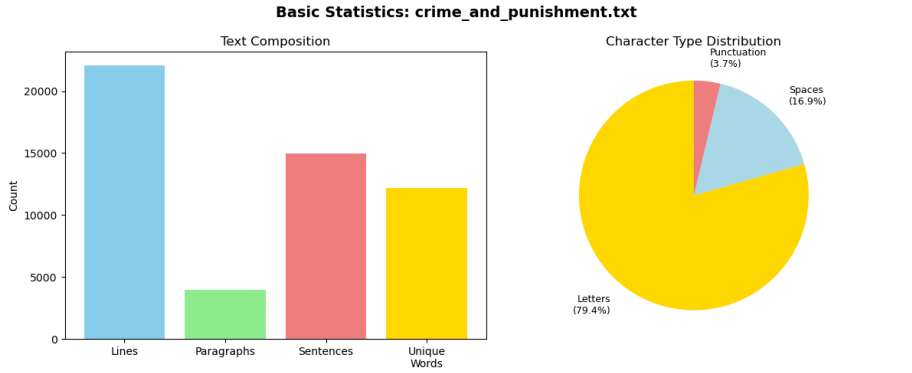
```
Enter your choice (1-7): 2

--- Basic Statistics for "crime_and_punishment.txt" ---
Lines: 22,060
Paragraphs: 3,968
Sentences: 14,940
Words: 203,505
Unique words: 12,212
Characters (with spaces): 1,135,110
Characters (without spaces): 949,458
Average words per line: 9.2
Average word length: 4.4 characters
Average words per sentence: 13.6

Generating basic statistics visualisation...

Press Enter to continue...

==================================================
           TEXT ANALYSER
==================================================
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
Current file: crime_and_punishment.txt

Enter your choice (1-7): 3

--- Word Analysis for "crime_and_punishment.txt" ---
Top 10 most common words:
   1. the              7,754 times (  3.8%)
   2. and              6,715 times (  3.3%)
   3. to               5,219 times (  2.6%)
   4. he               4,578 times (  2.2%)
   5. a                4,523 times (  2.2%)
   6. of               3,773 times (  1.9%)
   7. i                3,502 times (  1.7%)
   8. you              3,476 times (  1.7%)
   9. in               3,148 times (  1.5%)
  10. was              2,806 times (  1.4%)

Word length statistics:
  Shortest word: 1 characters
  Longest word: 23 characters
  Average word length: 4.4 characters
Words appearing only once: 5,621

Generating word analysis visualisation...

Press Enter to continue...

==================================================
           TEXT ANALYSER
==================================================
```

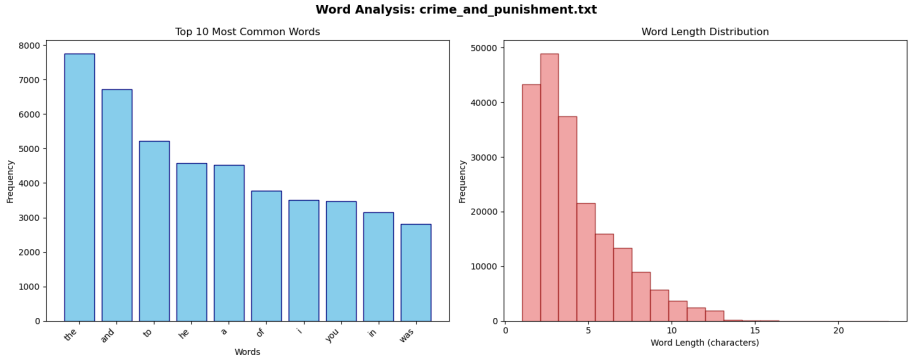
```
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
Current file: crime_and_punishment.txt

Enter your choice (1-7): 4

--- Sentence Analysis for "crime_and_punishment.txt" ---
Total sentences: 14,940
Average words per sentence: 13.6
Shortest sentence: 3 words
Longest sentence: 102 words

Shortest sentence text: CRIME AND PUNISHMENT
Longest sentence text: But that is the beginning of a new
story--the story of the gradual
renewal of a man, the story of hi...

Sentence length distribution (top 5):
  5 words: 1223 sentences
  4 words: 1150 sentences
  6 words: 1126 sentences
  7 words: 1008 sentences
  3 words: 955 sentences

Generating sentence analysis visualisation...

Press Enter to continue...

==================================================
           TEXT ANALYSER
==================================================
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
Current file: crime_and_punishment.txt

Enter your choice (1-7): 5

--- Character Analysis for "crime_and_punishment.txt" ---
Character type distribution:
  Letters: 874,218 (77.9%)
  Digits: 30 (0.0%)
  Spaces: 185,652 (16.5%)
  Punctuation: 40,924 (3.6%)

Most common letters:
   1. "e" - 103,215 times (11.8%)
   2. "t" - 79,319 times (9.1%)
   3. "a" - 73,040 times (8.4%)
```

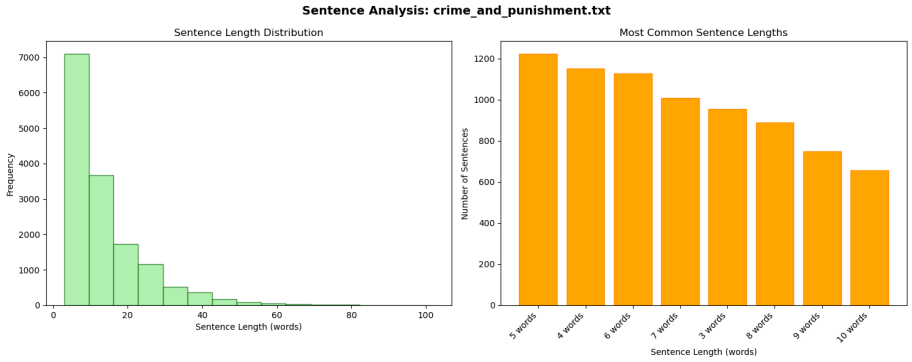
```
   4. "o" - 70,887 times (8.1%)
   5. "n" - 62,120 times (7.1%)
   6. "i" - 61,534 times (7.0%)
   7. "h" - 55,491 times (6.3%)
   8. "s" - 52,754 times (6.0%)
   9. "r" - 47,324 times (5.4%)
  10. "d" - 38,670 times (4.4%)

Generating character analysis visualisation...

Press Enter to continue...

==================================================
           TEXT ANALYSER
==================================================
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
Current file: crime_and_punishment.txt

Enter your choice (1-7): 6
Results exported to "results/
crime_and_punishment.txt_results.txt"
Results saved in simple text format - easy to read and
understand!

==================================================
           TEXT ANALYSER
==================================================
1. Load text file
2. Display basic statistics
3. Word frequency analysis
4. Sentence analysis
5. Character analysis
6. Export results
7. Exit
==================================================
Current file: crime_and_punishment.txt

Enter your choice (1-7): 7
Thank you for using Text Analyser!
```

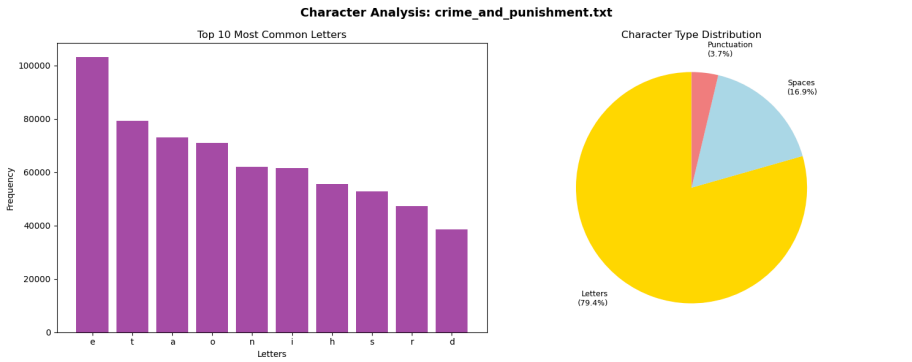
Output from Matplotlib

Below you will find the graphs produced by Matplotlib for the corresponding menu option in the example program. Your output does not need to match this exactly, but will give you an idea of what is expected.









Example Export Format

Lastly, here is an example compilation of the statistics for the book analysed in the example implementation. Again, your implementation does not need to match this exactly.

```
============================================================
TEXT ANALYSIS RESULTS
============================================================
File analysed: crime_and_punishment.txt
Analysis date: 2025-10-03 15:42:02

BASIC STATISTICS
--------------------
Lines: 22,060
Paragraphs: 3,968
Sentences: 14,940
Words: 203,505
Unique words: 12,212
Characters (with spaces): 1,135,110
Characters (without spaces): 949,458
Average words per line: 9.2
Average word length: 4.4 characters
Average words per sentence: 13.6

TOP 10 MOST COMMON WORDS
------------------------------
 1. the              7,754 times (  3.8%)
 2. and              6,715 times (  3.3%)
 3. to               5,219 times (  2.6%)
 4. he               4,578 times (  2.2%)
 5. a                4,523 times (  2.2%)
 6. of               3,773 times (  1.9%)
 7. i                3,502 times (  1.7%)
 8. you              3,476 times (  1.7%)
 9. in               3,148 times (  1.5%)
10. was              2,806 times (  1.4%)

WORD STATISTICS
--------------------
Shortest word: 1 characters
Longest word: 23 characters
Words appearing only once: 5,621

SENTENCE STATISTICS
-------------------------
Total sentences: 14,940
Shortest sentence: 3 words
Longest sentence: 102 words
Shortest sentence text: CRIME AND PUNISHMENT
Longest sentence text: But that is the beginning of a new
story--the story of the gradual
renewal of a man, the story of hi...

CHARACTER STATISTICS
-------------------------
Letters: 874,218 (77.9%)
Digits: 30 (0.0%)
Spaces: 185,652 (16.5%)
Punctuation: 40,924 (3.6%)
```

```
TOP 10 MOST COMMON LETTERS
------------------------------
 1. 'e' - 103,215 times (11.8%)
 2. 't' - 79,319 times (9.1%)
 3. 'a' - 73,040 times (8.4%)
 4. 'o' - 70,887 times (8.1%)
 5. 'n' - 62,120 times (7.1%)
 6. 'i' - 61,534 times (7.0%)
 7. 'h' - 55,491 times (6.3%)
 8. 's' - 52,754 times (6.0%)
 9. 'r' - 47,324 times (5.4%)
10. 'd' - 38,670 times (4.4%)

============================================================
End of Analysis Report
============================================================
```