In this project, I have mimicked the process of a MapReduce task. Specifically, I wrote my own map and reduce functions (without distributing to several machines) to mimic the process of mapper and reducer. The task is to count the number of occurrences of each word in a text file.

## Dataset

The input of this project is a text document (around 13,000 lines) which includes several paragraphs. It is raw data and needs some data cleaning work to prepare it for the next steps.

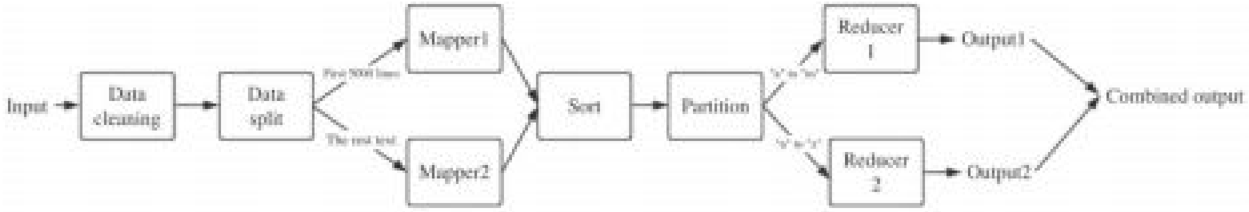
## Task

I built several functions to mimic each step of MapReduce. They are:

| **Function** | **Description** | **Input of this function** | **Output of this function** |
|---|---|---|---|
| **Data cleaning function** | Some data cleaning jobs, such as removing numbers, punctuations and special symbols, uppercase to lower case. | Raw text data | Clean text data |
| **Data split function** | Split the dataset into two parts: Part1 includes the first 5000 lines of the text file, Part2 includes the rest text. | Output of data cleaning function | Two separated subsets: Part1 and Part2. |
| **Mapper function** | Two mapper functions that produce a set of key-value pairs for Part1 and Part2 subsets respectively. | Output of data split function | Key-value pairs of Part1 and Part2. |
| **Sort function** | Sort by key of Part1 and Part2 together, with an ascending sort order | Output of mapper function | Sorted Key-value pairs for the whole dataset |
| **Partition function** | All the tokens (i.e., words) starting with letter “a” to “m” are sent to Reducer1, and the others (“n” to “z”) are sent to Reducer2. | Output of sort function | Two ascending ordered partitions. |

| | | | |
|---|---|---|---|
| **Reducer function** | Collect all values belonging to the key and count the frequency of words for the two ordered partitions. | Output of partition function | Word frequency of the ordered partitions. |
| **Main function** | Wrap all the steps together and combine the output of the two partitions together. | Output of reducer function | Final result of word counting. |

The figure below shows the basic workflow of this word count task.



## Output

**CSV file**: The final word count output will have a format like this:

| Word | Frequency |
|---|---|
| apple | 123 |
| banana | 45 |
| ... | ... |