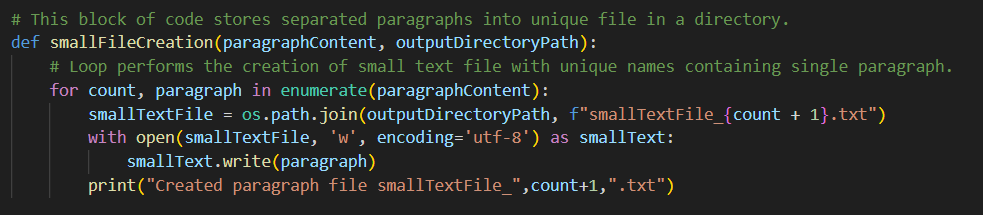
|  |
| --- |
| Assignment-1  Report |
|  |
| October 1  2023 |

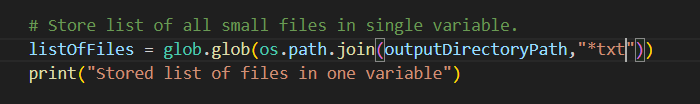
# Pseudo Word Count Distribution

|  |
| --- |
| Abstract The application needs a large text file containing multiple paragraphs as input. This large text file will be split into multiple small text files containing one paragraph each.  Using the multiprocessing and threading modules of python word count is performed on these small text files, such that each process handles multiple text files and has multiple threads to handle one small text file per thread.  Threads in the process execute parallel to each other, at the end of the process the output from each thread is aggregated and consolidated into one single output file containing the word count of all the small text files. |
| Implementation The python code imports 8 libraries in order to run distributed word count application successfully. The libraries are as shown in the below screenshot:    The input file and the directory location (to store the small text file containing one paragraph each) are defined in a variable and passed as inputs to the function.  The application first performs the sanity check to ensure that the input files exist, a function **sanityCheck** is defined and it returns the Boolean value 0 or 1 to main function depending on the input provided. Using the **sys library** the program would exit if the input file is not present at the defined location. If the output directory is not present then using the **os library** the output directory will be created. |
| Once the sanity check is performed the application performs the splitting of large text by paragraph and saves each of the paragraph into a variable. This operation is performed by splittingLargeText function. This function uses the re library to define the regular expression, to split the paragraph when two new line occurs. These split paragraphs are stored in a variable. |

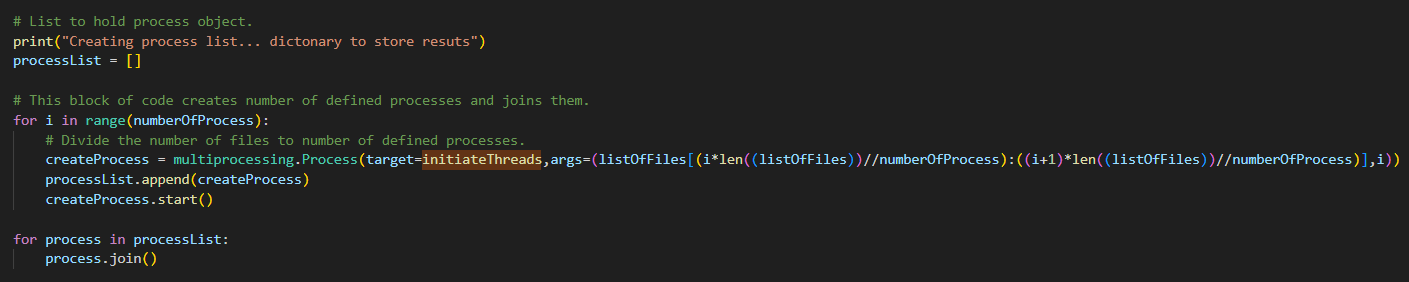
The variable paragraphContent is returned to main, and later on passed to new function smallFileCreation, this function takes the paragraphContent and outputDirectoryPath as input and using the loop creates small text files containing one paragraph each.



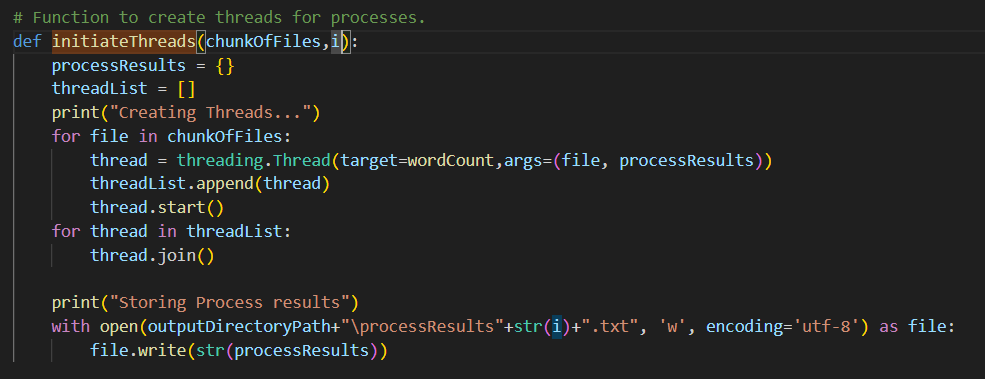
After creation of small files, the list of names of these files are stored in variable listOfFiles, the storage operation is performed by the glob module of python.



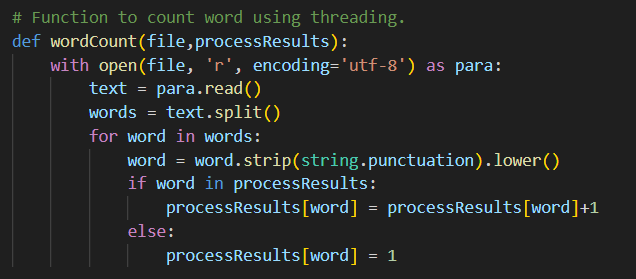
The next step is to define number of processes and divide the number of small files amongst the process.



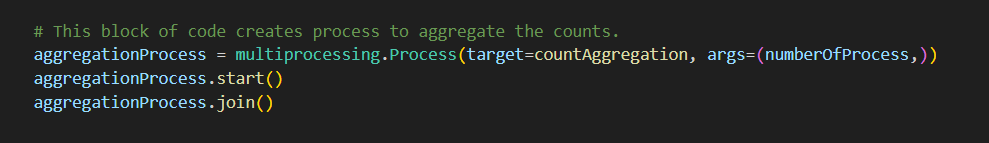
This is done using the looping and performing mathematical operations on the variable listOfFiles the process targets to the function initiateThreads, this function takes a counter and group of files as input and create threads for each file to perform the word count operation. After the word count is performed the function creates the word file and saves the word count for each process that initiates the threads.



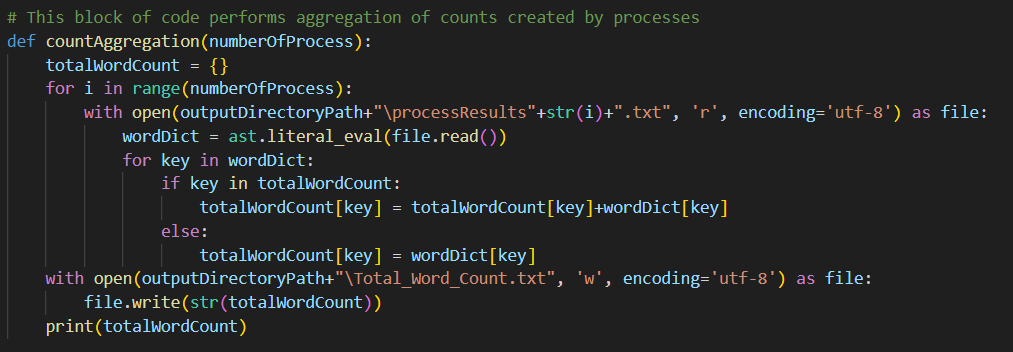
The function wordCount is nothing but the threads that are initiated, each thread performs word count operation on a single small text file and returns the output to respective process.



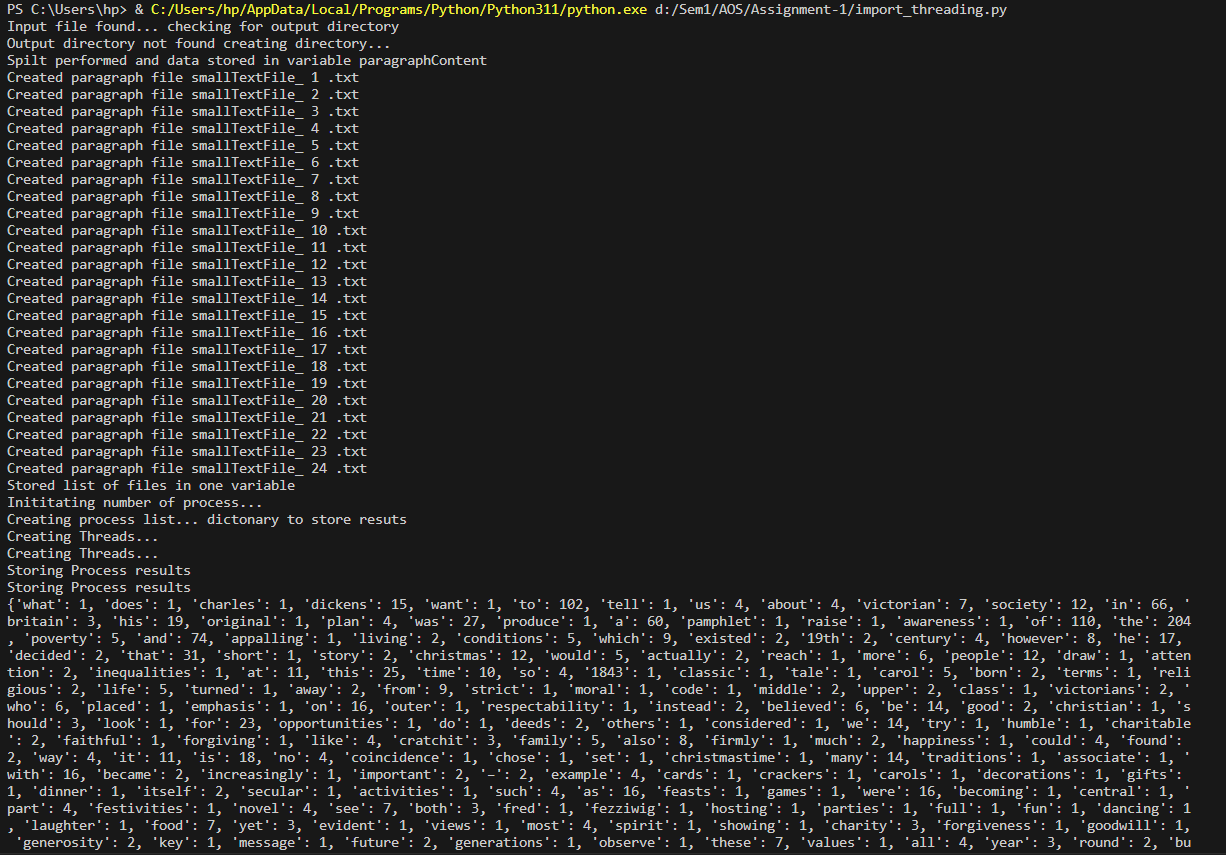
Lastly the aggregation should be performed for the word counts done by all the processes, for this a new process called aggregationProcess is defined that targets to the countAggregation function



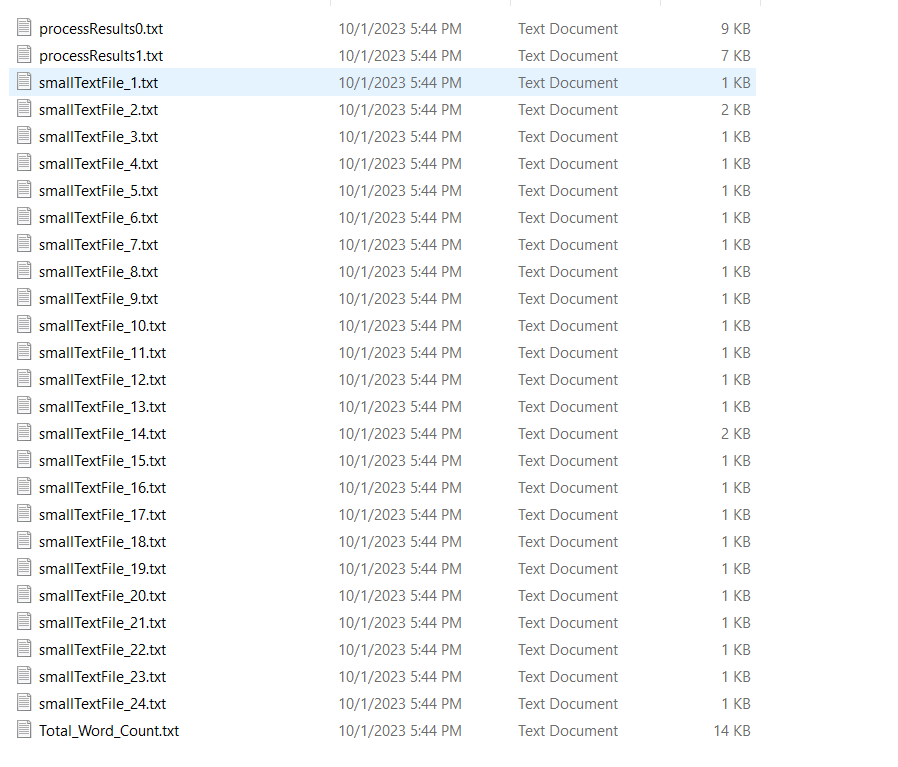
The countAggregation function takes the number of process and input and performs the aggregation of counts for words that are done by all the processes, the python ast library is used to convert the string to key value pair (map). Using the loops, the aggregation is performed and the final file named totalWordCount is generated and stored in the output directory.



After passing the input file and running the program the output should look like below image :



The small para and the output file should also be generated in the directory like shown below:



## Citations:

1. <https://www.analyticsvidhya.com/blog/2022/05/an-introduction-to-mapreduce-with-a-word-count-example/>
2. <https://docs.python.org/3/library/re.html>
3. <https://docs.python.org/3/library/multiprocessing.html>
4. <https://www.geeksforgeeks.org/python-sys-module/>
5. <https://www.educative.io/answers/what-is-astliteralevalnodeorstring-in-python>
6. <https://www.geeksforgeeks.org/python-program-to-count-words-in-a-sentence/>
7. <https://www.geeksforgeeks.org/create-an-empty-file-using-python/>