Dorado, Abram C. **Sir Timothy Ranera**
Ruiz, Marvic Gabriel **CMSC 135**

MULTIPROCESSING CHALLENGE

In this solution, the goal is to determine the top 100 word tokens based on their occurrence in a given dataset. The approach used here involves multiprocessing to speed up the counting process.

The solution first defines a function `preprocess_text()` to preprocess the raw text. This function removes unwanted characters, converts the text to lowercase, and splits it into a list of tokens.

Next, the `count_word_occurrences()` function is defined to count the occurrences of each word token in a given list of tokens. It iterates through the tokens and maintains a dictionary to keep track of the counts for each token.

In the `main()` function, the dataset is read from the input file, and the text is preprocessed using the `preprocess_text()` function. Then, multiprocessing is used to count the word occurrences. The `count_word_occurrences()` function is applied to the preprocessed tokens using the `map()` method from the **`multiprocessing.Pool`** class, which distributes the workload among multiple processes.

The word counts obtained from each process are combined into a single dictionary **`combined_word_counts`**. The dictionary is then sorted based on the counts in descending order to obtain the **`sorted_word_counts`** list.

From the sorted word counts, the top 100 word tokens are extracted into the **`top_100_tokens`** list. Finally, the unique tokens are written to a file named 'unique_tokens.txt', and the top 100 tokens are written to 'top_100_tokens.txt' using the `write_list_to_file()` function.

To measure the execution time of the program, the start and end times were recorded using the **`datetime`** module. The duration of the process was calculated by subtracting the start time from the end time.

The solution leverages multiprocessing to parallelize the counting process, improving the performance by utilizing multiple CPU cores. By distributing the workload among processes, the solution can efficiently handle large datasets.

SCREENSHOT OF THE CALCULATED TIME EXECUTION: