Ryan Myers
CPE 469
Dr. Maria Pantoja
Assignment 2

My code can be compiled by running “go build” on the file “matrix_mult.go” for the 1000x1000 matrix multiplication implementation and the file “word_search.go” for the word search implementation. The matrix multiplication code can be run by calling “./matrix_mult.exe” and it will show the times taken for both the sequential and concurrent implementation of the algorithm. The word search code can be run by calling “./word_search.exe [text file] [word]” where the “text file” is the path to the file to be searched and “word” is the word you are searching for. Running the word search code will return the number of words found as well as the time taken for the sequential and concurrent approaches.

After running the code for multiple trials these were my results:

	Trial 1	Trial 2	Trial 3	Trial 4
Sequential	3.3944395s	3.3968708s	3.4052535s	3.452009s
Concurrent	884.6849ms	886.6284ms	883.6693ms	869.7048ms

Table 1: Results for matrix multiplication using sequential and concurrent approaches

	Trial 1	Trial 2	Trial 3	Trial 4
Sequential	11.9672ms	11.958ms	11.0038ms	11.9682ms
Concurrent	22.9376ms	21.9166ms	21.9099ms	21.9411ms

Table 2: Results for word search using sequential and concurrent approaches

After attempting a few different solutions for the concurrent word search algorithm I was unable to make it faster than the sequential implementation and still get the correct answer. I have also attached a text file with my report called “word_search_implementation2.txt” which has a portion of code to implement the concurrent word search that runs significantly faster but the word frequencies are slightly off and I was unable to figure out the reason why. Overall I believe my solutions work relatively well and I was able to gain a much better understanding of how Goroutines run and communicate.