Compilation and Execution

Simple run once
    $ go run ./strSearch.go

Test different chunk sizes by adding any command line argument(s)
    $ go run ./strSearch.go checks command line length greater than 1

The testing option will check numerous chunk sizes and returns which chunk size (amount of lines to process per goroutine) has the best speed increase from the sequential code.

On my machine, I wasn't seeing a consistent one chunk size that was better than the rest, so I set the default to 20 line chunks.

Results

- In all cases, the word count is the same for sequential and distributed
- Processing 20 lines per goroutine
- Speed up calculated by dividing sequential time by distributed time

Run No.	Sequential Time	Distributed Time	Speed Up
1	57.8491ms	34.6271ms	1.671
2	51.5528ms	25.8198ms	1.997
3	48.4723ms	41.9386ms	1.156
4	42.6816ms	23.7238ms	1.799
5	45.735ms	30.6017ms	1.495
Avg.	49.2582ms	31.3422ms	1.572