Assignment #4 - Word Blast

Goal: Read 'War & Peace' to count and tally each word longer than 6 characters or longer.
Only using Linux file functions (open, close, read, lseek, pread) no additional libraries
But: must do this using threads (This assignment will be using pthreads)

- Each thread will take a chunk of the file and process it, returning its results to the main which tallies.
- The main will print the ten, 6 or more character, words with the highest tallies, from highest to lowest.
- Program should take two parameters - COMPLETE
  - FileName is the name of the file to read - WarAndPeace.txt ✓ Read-Only mode
  - TreadCount is the number of threads you should spawn to evenly divide the work ✓
- These threads should be launched together in a loop

* HINTS:
- Do not forget to protect critical sections
- Make sure to use thread safe library calls
- You need to know how long the input file is (use lseek)
- Do not use pipes. You must use mutex locks

- strtok_r is a tokenizer that is thread safe

TODO:
- Accepting parameters - COMPLETE
  - FileName - Read into opening files w/ C ✓
  - TreadCount as int (accepting argv[2] as TreadCount ✓
    - Argv is a pointer to a string - used atoi function to convert to int
    - You need to know how long the input file is (use lseek) Determine file size ✓
    - Partition must be determined (Divide by number of threads) ✓
- Creating a structure to the top 10 six-character word tally.
  - Option: Create a hashmap to count each word occurance
    < key: word, Value: Occurance Increment >
    - Define the structure - Needs string / char * for word, int for count
- Threading
  - Thread Pool - Create a number of threads at start up and use as needed.? ✓
  - Method or code block must be written that each thread will execute (Instructions or algorithm)
  - pthread_create, pthread_join, p_thread_exit?
    - What needs to be used to destroy the threads after use? Mutex-lock & buffers ✓
    - Function must be created to run pthread_create (Code block to execute)
- Close file - Release allocated memory

- Treading (Cont.)
    - Function must be created to run pthread_create (Code block to execute)
        - There should be a buffer to hold chunks allocated to thread (malloc use) ✓
        - Tokenizer used (strtok_r)
        - Pointer used to point to token
    - ~~Each thread will take a chunk of the file~~ and process it, returning its results to the main which tallies.
        - Determine the chunk - Requires file size & number of threads ✓
        - Load chunks into a buffer - malloc buffer to load for each thread - using read() ✓
            - Pointer need to indicate position - NOT NEEDED ✓ CHECK AFTER COMPLETION
        - Are there bytes left over? - Test ✓
            - If so, modify buffer size and amount to pass it. How to allocate for this? ✓
                - Varies per thread count. Allocate memory in buffer to handle leftover for final thread? ✓
        - Check token if it is a six letter word
            - If so, check if it is in data structure - Compare string function - case sensitive
                - If so, increment
                - If not, add and increment
        - Continue until token returns null