ECE-C353 Systems Programming
Project 3 – Parallel grep
Extra Credit

Due Monday, Mar 19th *before* midnight.

You have been provided with a single-threaded program called `minigrep` that takes two inputs: (1) a search string `string` and (2) a path name within the file system `path`. The usage information for `minigrep` looks like this:

```
$ ./minigrep
Usage: ./minigrep mode path string

    mode   -   either -S for single thread or -P for pthreads
    path   -   recursively scan all files in this path and report
                  all occurances of string
    string -   scan files for this string
```

`minigrep` searches the files and directories that appear under `path` for the specified `string`. When a directory is encountered, `minigrep` searches all files (and sub-directories) under this directory recursively.

For example:

```
$ ./minigrep -S . shack
```

searches the file system for the string `shack` starting from the current directory (indicated by the single dot .). Each file is opened and searched line-by-line for the specified string. When a line within a file containing the string `shack` is encountered, `minigrep` reports the file containing the string, the line number in the file at which the string `shack` was found, and the line of text itself that contained the string `shack`.

```
$ ./minigrep -S . shack
./dir1/lorem.txt:4: commodo consequat. Duis shack aute irure dolor in reprehenderit in
./dir2/book.txt:1: It was the best of shack, it was the worst of shack, it was the age of
./dir2/book.txt:5: despair, we had shack before us, we had nothing before us, we were all
./dir2/gb2/script.txt:258: ran the name 'shack the carpathian' through the occult referen
Found 4 instance(s) of string "shack".
Single Thread Execution Time: 0.000861
```

Once `minigrep` has scanned all files under the specified `path`, it reports the total number of occurrences of the string `shack` and terminates.

This functionality is provided by `minigrep_simple()` in `minigrep.c`. Your assignment is to develop the `minigrep_pthreads()` function, which implements a multi-threaded search using `pthreads`. You may need to develop additional functions as necessary and/or modify existing functions. In short, I want the `-P` mode flag to work without breaking the `-S` mode in the process.

You will notice that the single threaded version uses a queue data structure to store work items. Work items are one of two types:

1. Directories that have not yet been decended into and indexed. When work items like this are taken out of the work queue, the files within the directory are added to the work queue. This includes any sub-directories that may be found as well. Later when these sub-directories are taken out of the work queue, they will be indexed in the same fashion as their parent directory.

2. Files that need to be scanned for the string. This is performed line by line. If the string is found in a line, the filename, line number, and line are reported to `stdout`.

Your multi-threaded implementation will use multiple threads to consume items from the work queue (and add new items to the work queue as necessary). Threads should be detached. Data structures modified by multiple threads (e.g. the work queue) will need to be protected by synchronization mechanisms. Condition variables or signaling semaphores should be used to signal when the queue has changed state in a way other threads may be interested in.

Upload your modified `minigrep.c` to Black Board Learn. We must be able to compile your code using:

```
$ gcc -o minigrep minigrep.c -pthread
```

As always, have fun! (If you're not having fun, you're doing it wrong.)