

CSE 333 – OPERATING SYSTEMS

PROJECT #3 REPORT

Group Members:

Ömer Faruk Çakı 150117821

Havva Karaçam 150315029

Main

Inside main, we get command line arguments and parse them. After that the first thing we do is getting the line count of the file with our `lineCount()` function and we set our read threads' threshold to that value.

Read Threads:

We first create our thread initializer with: `pthread_t readThreads[readThreadCount];`

Then using a for loop, we set our threads execution function as `_read()`

`_read()` functions works together with `getReadNum()` function. `getReadNum()` function is start checking from the 0th index until find an available line to read(which is not yet read by any other read threads) and return its line index number to `_read()` function. In order to prevent the same line number returned to few different read threads we use a mutex lock inside before executing `getReadNum()` and release it right after we get the line number to read. We don't need to use a mutex to read that line because unique value will be returned for all read threads, so we unlock the mutexRead mutex right after `getReadNum()` method returns an index.

After we get our line number, thread reads that line from the file by executing `getLine()` function.

While loop inside the `_read` ensure that a thread will continue executing by processing the next line until all lines finished, so a thread won't die after processing a single line. After we process every line we set `readCompleted` value to 1.

Upper Threads:

We first create and initialize upper threads inside main. Upper threads run the **upper()** function. Upper function is also attached with `getUppercaseIndex()` function which returns the index value which can be processed, that means that index is already read by a reader thread and currently residing in `records[index]`.

We use `mutexUpper` mutex in order to ensure that not more than a upper thread will execute the same value (converting uppercase).

After a upper thread convert the text to uppercase it changes the string store inside `records[index].text` to new uppercased value and sets `records[index].upper = 1` to mark as uppercased.

Again, using the `while(1)` loop inside the `upper()` we process all values by upper threads until there is no one left. We use `upperCompleted` int value to keep track count of values processed.

Following line exits the thread function when there is no non-uppercased threads left

```
if (readCompleted == 1 && upperCompleted >= (limit + 1))
```

Replace Threads:

Replace threads are very similar to upper threads we explained above. Only difference is replace threads uses the `replace()` function. They also share the same `mutexUpper` mutex in order to get index number to process from `getReplaceIndex()` function. Because replace threads and uppercase threads shouldn't work on the same value at the same time.

Write Threads:

The `getWriteIndex()` method inside `write()` used to get lines which are already processed by upper threads and replace threads and ready to write into file.

`getWriteIndex()` function checks `records[index].upper` and `records[index].replace` values one by one and returns the first record index which has `upper=1` and `replace=1`.

Write thread does its job by executing `writeToLine()` function and tells which value to write into which file's which line. `writeToLine` and `getWriteIndex` functions both executed inside `wireteMutex` mutex.

writeToLine() accomplish the writing task by using a temporary text file instead directly editing the original file. Because we think that this is not the objective of this project and other way was way more complicated to code. So that first we write new text which changed line into a new text file and we rename it to the original text file, after that we delete the temporary text file so it seems like we just edited the original file.

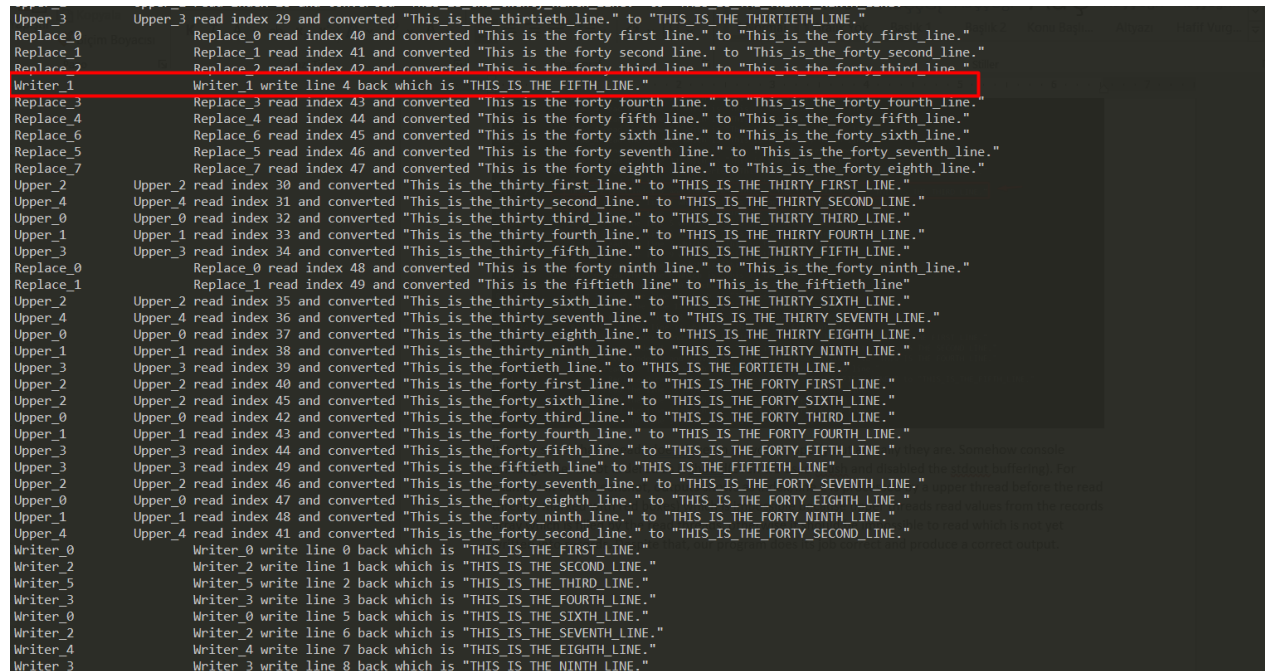
Example Execution

```
ofaruk@DESKTOP-AR6CVQ2:/mnt/c/Users/omerf/Desktop/DERS/OS/Project3$ gcc main.c -o main.out -lpthread
ofaruk@DESKTOP-AR6CVQ2:/mnt/c/Users/omerf/Desktop/DERS/OS/Project3$ ./main.out -d deneme.txt -n 15 5 8 6
Line count: 49
Read_0 Read_0 read the line 0 which is "This is the first line."
Read_0 Read_0 read the line 15 which is "This is the sixteenth line."
Read_4 Read_4 read the line 4 which is "This is the fifth line."
Read_4 Read_4 read the line 17 which is "This is the eighteenth line."
Read_4 Read_4 read the line 18 which is "This is the nineteenth line."
Read_4 Read_4 read the line 19 which is "This is the twentieth line."
Read_4 Read_4 read the line 20 which is "This is the twenty first line."
Read_4 Read_4 read the line 21 which is "This is the twenty second line."
Read_4 Read_4 read the line 22 which is "This is the twenty third line."
Read_4 Read_4 read the line 23 which is "This is the twenty fourth line."
Read_4 Read_4 read the line 24 which is "This is the twenty fifth line."
Read_4 Read_4 read the line 25 which is "This is the twenty sixth line."
Read_4 Read_4 read the line 26 which is "This is the twenty seventh line."
Read_4 Read_4 read the line 27 which is "This is the twenty eighth line."
Read_4 Read_4 read the line 28 which is "This is the twenty ninth line."
Read_4 Read_4 read the line 29 which is "This is the thirtieth line."
Read_4 Read_4 read the line 30 which is "This is the thirty first line."
Upper_4 Upper_4 read index 4 and converted "This is the fifth line." to "THIS IS THE FIFTH LINE."
Read_1 Read_1 read the line 1 which is "This is the second line."
Upper_2 Upper_2 read index 2 and converted "This is the third line." to "THIS IS THE THIRD LINE."
Read_2 Read_2 read the line 2 which is "This is the third line."
Read_0 Read_0 read the line 16 which is "This is the seventeenth line."
Read_5 Read_5 read the line 5 which is "This is the sixth line."
Read_3 Read_3 read the line 3 which is "This is the fourth line."
Read_7 Read_7 read the line 7 which is "This is the eighth line."
Read_9 Read_9 read the line 9 which is "This is the tenth line."
Read_8 Read_8 read the line 8 which is "This is the ninth line."
Read_6 Read_6 read the line 6 which is "This is the seventh line."
Read_10 Read_10 read the line 10 which is "This is the eleventh line."
Read_12 Read_12 read the line 12 which is "This is the thirteenth line."
Read_11 Read_11 read the line 11 which is "This is the twelfth line."
Read_13 Read_13 read the line 13 which is "This is the fourteenth line."
Read_14 Read_14 read the line 14 which is "This is the fifteenth line."
Upper_0 Upper_0 read index 0 and converted "This is the first line." to "THIS IS THE FIRST LINE."
Upper_1 Upper_1 read index 1 and converted "This is the second line." to "THIS IS THE SECOND LINE."
Upper_3 Upper_3 read index 3 and converted "This is the fourth line." to "THIS IS THE FOURTH LINE."
Read_4 Read_4 read the line 31 which is "This is the thirty second line."
Replace_0 Replace_0 read index 4 and converted "THIS IS THE FIFTH LINE." to "THIS IS THE FIFTH LINE."
Read_1 Read_1 read the line 32 which is "This is the thirty third line."
Read_2 Read_2 read the line 33 which is "This is the thirty fourth line."
Read_0 Read_0 read the line 34 which is "This is the thirty fifth line."
Read_5 Read_5 read the line 35 which is "This is the thirty sixth line."
```

It might seem that threads doesn't seem to work properly but actually they are. Somehow console printing order is not ordered as it should(even we used fflush and disabled the stdout buffering). For example in this screenshot, outputs shows as index/line 2 is processed by a upper thread before the read thread(I marked with red boxes) which is impossible because upper threads read values from the records array which is filled by the read threads, otherwise it should be impossible to read which is not yet written into records. Despite that, our program does its job correct and produce a correct output.

And it also seems like a single read thread blocking other read threads and reads many line at once, I think it because we already opened the same file to count its lines and Operating System is cached it. We believe that's why readers threads are running that much fast then they should be in normal.

Also, our write threads outputs are all together, that doesn't mean our writer threads are not asynchronous, they are executing concurrently as given in this screenshot:



```
Upper_3      Upper_3 read index 29 and converted "This is the thirtieth line." to "THIS IS THE THIRTIETH LINE."
Replace_0    Replace_0 read index 40 and converted "This is the forty first line." to "This is the forty first line."
Replace_1    Replace_1 read index 41 and converted "This is the forty second line." to "This is the forty second line."
Replace_2    Replace_2 read index 42 and converted "This is the forty third line." to "This is the forty third line."
Writer_1     Writer_1 write line 4 back which is "THIS IS THE FIFTH LINE."
Replace_3    Replace_3 read index 43 and converted "This is the forty fourth line." to "This is the forty fourth line."
Replace_4    Replace_4 read index 44 and converted "This is the forty fifth line." to "This is the forty fifth line."
Replace_6    Replace_6 read index 45 and converted "This is the forty sixth line." to "This is the forty sixth line."
Replace_5    Replace_5 read index 46 and converted "This is the forty seventh line." to "This is the forty seventh line."
Replace_7    Replace_7 read index 47 and converted "This is the forty eighth line." to "This is the forty eighth line."
Upper_2      Upper_2 read index 30 and converted "This is the thirty first line." to "THIS IS THE THIRTY FIRST LINE."
Upper_4      Upper_4 read index 31 and converted "This is the thirty second line." to "THIS IS THE THIRTY SECOND LINE."
Upper_0      Upper_0 read index 32 and converted "This is the thirty third line." to "THIS IS THE THIRTY THIRD LINE."
Upper_1      Upper_1 read index 33 and converted "This is the thirty fourth line." to "THIS IS THE THIRTY FOURTH LINE."
Upper_3      Upper_3 read index 34 and converted "This is the thirty fifth line." to "THIS IS THE THIRTY FIFTH LINE."
Replace_0    Replace_0 read index 48 and converted "This is the forty ninth line." to "This is the forty ninth line."
Replace_1    Replace_1 read index 49 and converted "This is the fiftieth line" to "This is the fiftieth line"
Upper_2      Upper_2 read index 35 and converted "This is the thirty sixth line." to "THIS IS THE THIRTY SIXTH LINE."
Upper_4      Upper_4 read index 36 and converted "This is the thirty seventh line." to "THIS IS THE THIRTY SEVENTH LINE."
Upper_0      Upper_0 read index 37 and converted "This is the thirty eighth line." to "THIS IS THE THIRTY EIGHTH LINE."
Upper_1      Upper_1 read index 38 and converted "This is the thirty ninth line." to "THIS IS THE THIRTY NINTH LINE."
Upper_3      Upper_3 read index 39 and converted "This is the fortieth line." to "THIS IS THE FORTIETH LINE."
Upper_2      Upper_2 read index 40 and converted "This is the forty first line." to "THIS IS THE FORTY FIRST LINE."
Upper_2      Upper_2 read index 45 and converted "This is the forty sixth line." to "THIS IS THE FORTY SIXTH LINE."
Upper_0      Upper_0 read index 42 and converted "This is the forty third line." to "THIS IS THE FORTY THIRO LINE."
Upper_1      Upper_1 read index 43 and converted "This is the forty fourth line." to "THIS IS THE FORTY FOURTH LINE."
Upper_3      Upper_3 read index 44 and converted "This is the forty fifth line." to "THIS IS THE FORTY FIFTH LINE."
Upper_2      Upper_2 read index 46 and converted "This is the forty seventh line." to "THIS IS THE FORTY SEVENTH LINE."
Upper_0      Upper_0 read index 47 and converted "This is the forty eighth line." to "THIS IS THE FORTY EIGHTH LINE."
Upper_1      Upper_1 read index 48 and converted "This is the forty ninth line." to "THIS IS THE FORTY NINTH LINE."
Upper_4      Upper_4 read index 41 and converted "This is the forty second line." to "THIS IS THE FORTY SECOND LINE."
Writer_0     Writer_0 write line 0 back which is "THIS IS THE FIRST LINE."
Writer_2     Writer_2 write line 1 back which is "THIS IS THE SECOND LINE."
Writer_5     Writer_5 write line 2 back which is "THIS IS THE THIRD LINE."
Writer_3     Writer_3 write line 3 back which is "THIS IS THE FOURTH LINE."
Writer_0     Writer_0 write line 5 back which is "THIS IS THE SIXTH LINE."
Writer_2     Writer_2 write line 6 back which is "THIS IS THE SEVENTH LINE."
Writer_4     Writer_4 write line 7 back which is "THIS IS THE EIGHTH LINE."
Writer_3     Writer_3 write line 8 back which is "THIS IS THE NINTH LINE."
```

In this run time, only a writer thread manage to run before other replace and upper threads. Because write threads are way more slower compared to upper and replace threads because the need disk I/O operations.

Thanks