# OPERATING SYSTEMS

# PROJECT 3 - REPORT

MURAT ŞENOL - 150117039

SÜLEYMAN BARIŞ ESER - 150116055

In our assignment we used "pthread" library in c for threads and semaphores for locking the critical sections.And we also used a simple queue for synchronization of threads.And we used struct for a thread. We assigned a each thread number for thread number and an index line to read.

## <u>Reader Part:</u>

Reader thread reads the data from file and writes the content to an array. To give unique values to the threads create a critical section each thread will have a unique line number.After a thread take a duty, it will seek the file with a unique line and read it. While a read thread reading a line to disable the deadlock we created a new critical section. After those operation, the thread again comes to while again doing same operation and that resume until the last line readed. After the last line readed, the global variable linenumber which keeps the number of lines set equals to reader_tail which keeps last readed line and then, thread again comes to while loop and breaks the while loop.

## <u>Upper & Replace Part:</u>

Same as the reader part we gave unique values to each upper thread. When a reader thread reads and writes to the array an upper or a replace thread will enter the critical sections in their respective methods they do their things. Upper thread turns the letter uppercase and replace thread replaces the white spaces with underscores. While loop in methods is to make a single thread to work on multiple lines.Threads will wait for their assigned lines in the method. After all lines are converted to uppercase then uppercase threads break the while loop and finish their jobs. Same for replace case.

## <u>Write Part:</u>

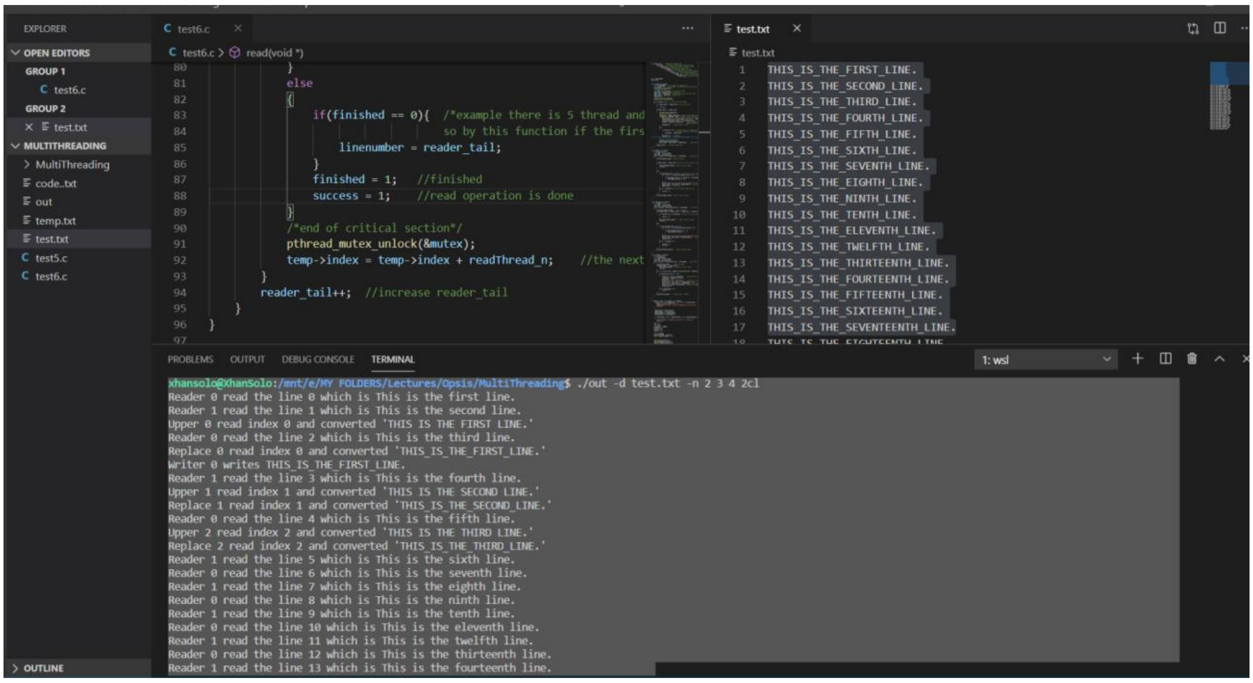
After replace_tail and upper_tail will be grater than write_tail, write threads write the line to into file with specified location and those operations continue until there is no more unwrited index in the array.

All of the read , upper , replace and write operations will write a simple info message to the console when it is done. Comments in the source code is more detailed and on place.Please look at the comments if you did not understand enough.

To sum up, a replace thread and a uppercase thread and read thread or write thread can be occur in unit time. In that unit time, the replace thread operate a unique line, the uppercase thread can convert a unique line and read can read a unique line or write can write into a unique line. To achieve this scenario, we used global semaphores and mutexes.

And more informations are included in the code by the comments.

Sample output.



```c
                }
                else
                {
                    if(finished == 0){  /*example there is 5 thread and
                                         so by this function if the firs
                        linenumber = reader_tail;
                    }
                    finished = 1;   //finished
                    success = 1;    //read operation is done
                }
                /*end of critical section*/
                pthread_mutex_unlock(&mutex);
                temp->index = temp->index + readThread_n;    //the next
            }
            reader_tail++;  //increase reader_tail
        }
    }
```

```
THIS_IS_THE_FIRST_LINE.
THIS_IS_THE_SECOND_LINE.
THIS_IS_THE_THIRD_LINE.
THIS_IS_THE_FOURTH_LINE.
THIS_IS_THE_FIFTH_LINE.
THIS_IS_THE_SIXTH_LINE.
THIS_IS_THE_SEVENTH_LINE.
THIS_IS_THE_EIGHTH_LINE.
THIS_IS_THE_NINTH_LINE.
THIS_IS_THE_TENTH_LINE.
THIS_IS_THE_ELEVENTH_LINE.
THIS_IS_THE_TWELFTH_LINE.
THIS_IS_THE_THIRTEENTH_LINE.
THIS_IS_THE_FOURTEENTH_LINE.
THIS_IS_THE_FIFTEENTH_LINE.
THIS_IS_THE_SIXTEENTH_LINE.
THIS_IS_THE_SEVENTEENTH_LINE.
```

```
xhansolo@XhanSolo:/mnt/e/MY FOLDERS/Lectures/Opsis/MultiThreading$ ./out -d test.txt -n 2 3 4 2cl
Reader 0 read the line 0 which is This is the first line.
Reader 1 read the line 1 which is This is the second line.
Upper 0 read index 0 and converted 'THIS IS THE FIRST LINE.'
Reader 0 read the line 2 which is This is the third line.
Replace 0 read index 0 and converted 'THIS_IS_THE_FIRST_LINE.'
Writer 0 writes THIS_IS_THE_FIRST_LINE.
Reader 1 read the line 3 which is This is the fourth line.
Upper 1 read index 1 and converted 'THIS IS THE SECOND LINE.'
Replace 1 read index 1 and converted 'THIS_IS_THE_SECOND_LINE.'
Reader 0 read the line 4 which is This is the fifth line.
Upper 2 read index 2 and converted 'THIS IS THE THIRD LINE.'
Replace 2 read index 2 and converted 'THIS_IS_THE_THIRD_LINE.'
Reader 1 read the line 5 which is This is the sixth line.
Reader 0 read the line 6 which is This is the seventh line.
Reader 1 read the line 7 which is This is the eighth line.
Reader 0 read the line 8 which is This is the ninth line.
Reader 1 read the line 9 which is This is the tenth line.
Reader 0 read the line 10 which is This is the eleventh line.
Reader 1 read the line 11 which is This is the twelfth line.
Reader 0 read the line 12 which is This is the thirteenth line.
Reader 1 read the line 13 which is This is the fourteenth line.
```

## REFERENCES

- https://www.geeksforgeeks.org/use-posix-semaphores-c/
- https://www.geeksforgeeks.org/multithreading-c-2/