CSE 333 - OPERATING SYSTEMS Programming Assignment # 3 Report

Metehan ERTAN – 150117051
Furkan KUSE – 150117041

In this project we were expected to write a multi-threaded program that takes an input file and reads it line by line. After this a thread upper every character of that line and another thread underscore every space character. After updating every line, another thread writes this updated line to an output file. Number of threads and input file is given by user.

In our program we define max length of a line to be 256 characters, maximum lines in an input file as 1000 and maximum number of threads as 1000. User can change them.

Firstly, we count the lines that are in input file. Then, we initialize out global queue that stores lines. Global queue is named readed. We initialized our global variables. We created a mutex for reading threads. At last we created our threads with wanted amount. We create reading threads last in order to make them work simultaneously.

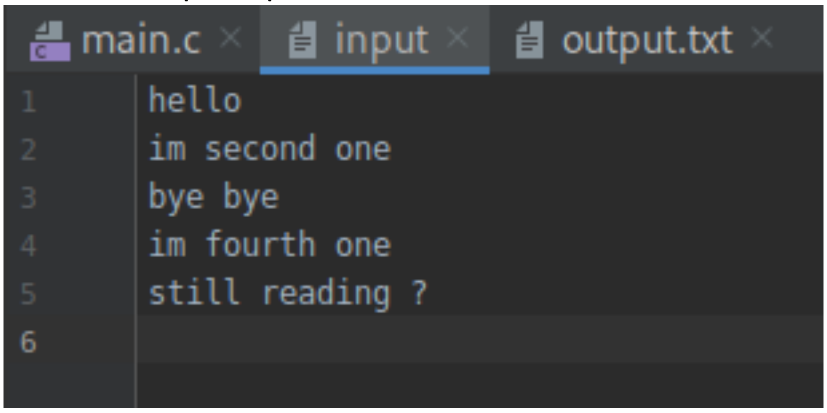
When reading threads enter the critical section, they get protected by mutexread. After they decide which line they will going to read, multiple threads can read from same file at the same time. After deciding which line to read our program increments the variable called readCount which holds the total number of read threads currently reading file by one. Then if readCount is equal to one we lock mutex called mutexRead to prevent writer threats from writing while reading threads reading the file. After finishing read operation, we decrement readCount by one and check if there is there any read thread working, in the protection of mutexRead. If there is no reading thread left, we unlock the mutexRead. Finally they enter a critical section that is protected by mutexchange. They initialize lock value as -1 of the corresponding line.

When we are updating a line with underscore or upper threads, we first lock mutexchange to prevent two different update thread from choosing and updating lock value of the same line at the same time. After choosing the line, we lock writing system to prevent writing thread from writing the line to output file before update threads finish their jobs.

Before we choose the line that is going to be written to the file, we lock the mutexchange to prevent two different write threads to choose same line to write. After we choose which line to write, we update its lock value as 3 to symbolize that line is done. To prevent two different write threads from writing to the output file at the same time, we lock the mutex called mutexwrite.

For writing part firstly, we create a file with the same number of lines as input file; but lines are empty. After this a thread takes the line and the place that line will be written. To add this line to the file thread starts writing output file to a temporary file. When wanted line comes, thread writes the line that is given to the threat for that line and continues copying other line to the temporary file. After this process ends copies the whole temporary file to the output file and deletes the temporary file.

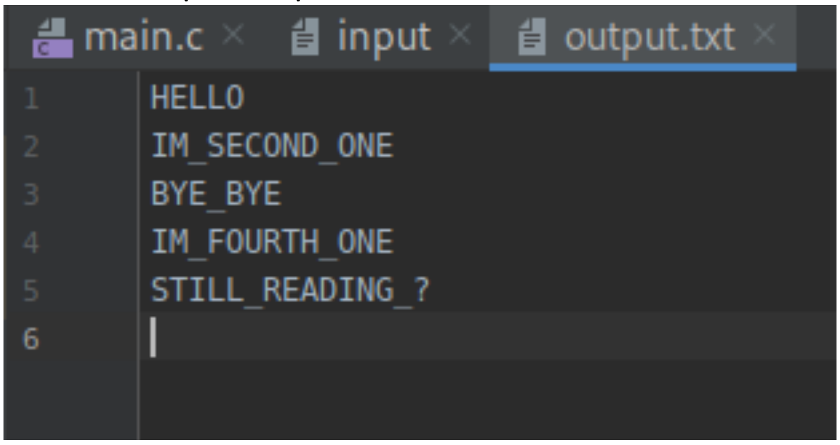
Example input:



```
hello
im second one
bye bye
im fourth one
still reading ?

```

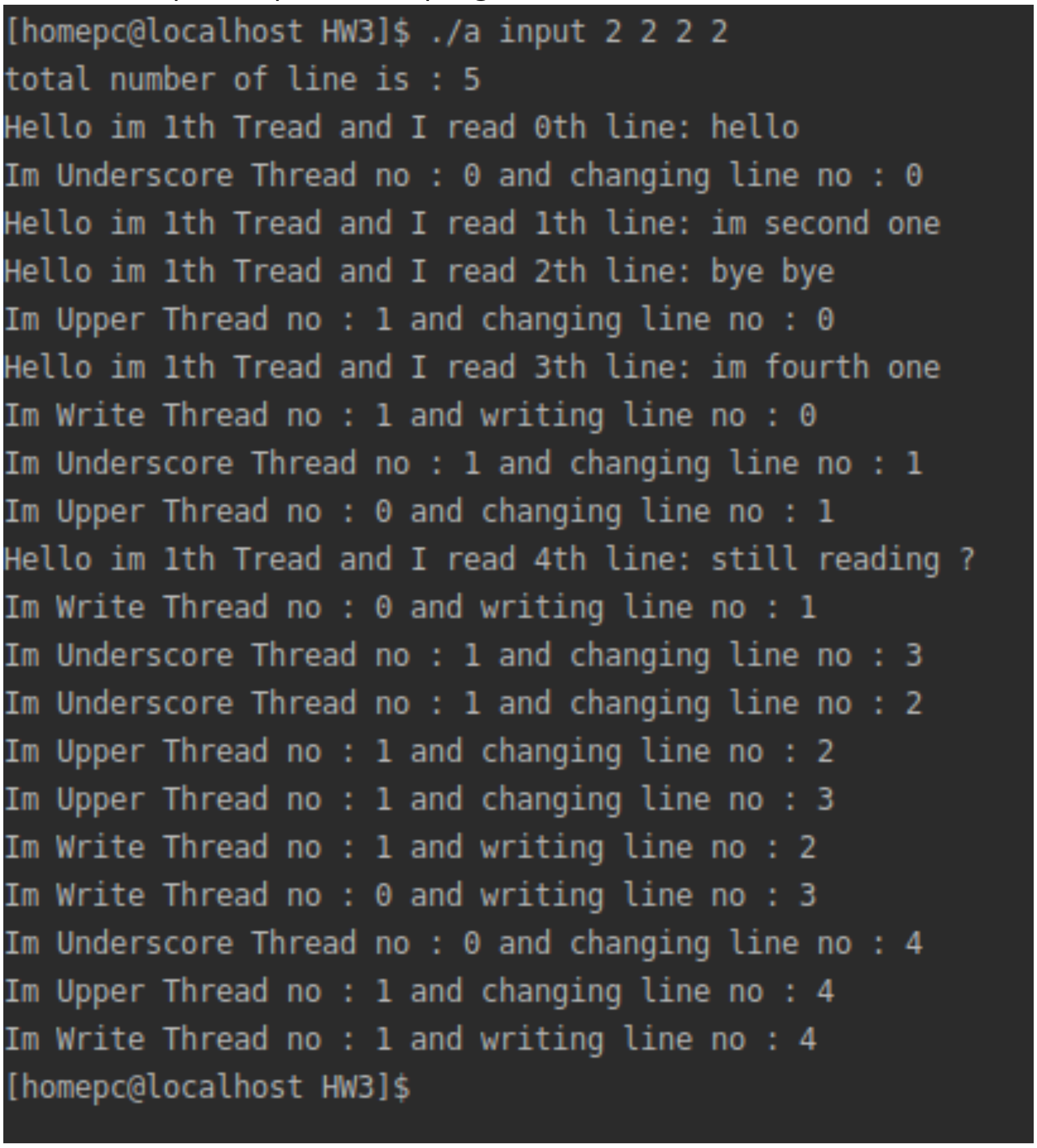
Example Output:



```
HELLO
IM_SECOND_ONE
BYE_BYE
IM_FOURTH_ONE
STILL_READING_?

```

Example output of the program:



```
[homepc@localhost HW3]$ ./a input 2 2 2 2
total number of line is : 5
Hello im 1th Tread and I read 0th line: hello
Im Underscore Thread no : 0 and changing line no : 0
Hello im 1th Tread and I read 1th line: im second one
Hello im 1th Tread and I read 2th line: bye bye
Im Upper Thread no : 1 and changing line no : 0
Hello im 1th Tread and I read 3th line: im fourth one
Im Write Thread no : 1 and writing line no : 0
Im Underscore Thread no : 1 and changing line no : 1
Im Upper Thread no : 0 and changing line no : 1
Hello im 1th Tread and I read 4th line: still reading ?
Im Write Thread no : 0 and writing line no : 1
Im Underscore Thread no : 1 and changing line no : 3
Im Underscore Thread no : 1 and changing line no : 2
Im Upper Thread no : 1 and changing line no : 2
Im Upper Thread no : 1 and changing line no : 3
Im Write Thread no : 1 and writing line no : 2
Im Write Thread no : 0 and writing line no : 3
Im Underscore Thread no : 0 and changing line no : 4
Im Upper Thread no : 1 and changing line no : 4
Im Write Thread no : 1 and writing line no : 4
[homepc@localhost HW3]$
```