CSE 333 - OPERATING SYSTEMS

PROJECT #3

“MULTI-THREADED TEXT MANIPULATION PROGRAM”

ADVISOR:

ASST. PROF. DR. ALİ HAYDAR ÖZER

TA:

ZUHAL ALTUNTAŞ

SANEM ARSLAN YILMAZ

ADVISEE:

150114022 – OĞUZHAN BÖLÜKBAŞ

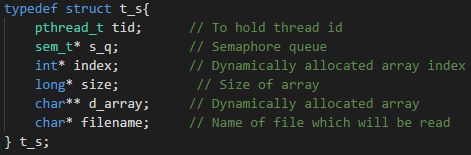
This program gets four arguments like as:

$ ./<object\_name> “-d” <example\_directory> “-n” <thread\_number>

It opens obtained directory and processes every “.txt” type files. Each files are assigned to only one thread. This thread opens specified file and reads it line by line. Each line is divided into words in order to store them. Before storing the word, the thread checks whether the word is stored before. If the word is not stored, the thread puts the word into dynamically allocated array.

Aim of the project is managing multi-threads which work at the same time. Managing multi-threads are difficult and may be harmful if it is not managed carefully and correctly because each thread tries to store a word into shared dynamic allocated array with using shared index. In order to prevent faults, semaphores are used in this project. The semaphore manages that how many threads can reach shared variables at the same time. It provides that with locking a part of the code. In this part, the thread reaches the shared array and obtains shared array index and stores its word into this index part of the array. After storage process, semaphore opens the lock to allow another thread for same processing.

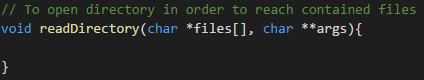
Now, we can examine detail of the program:



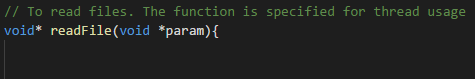
This structure stores shared variables’ address in order to change this variable threads

together.





This function opens the obtained directory in order to reach contained files.



This funciton reads each file in that directory.



This code line starts lock code part which is used to change shared variables.



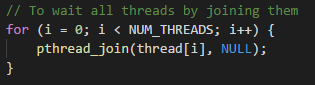
This code line unlocks the code part. After processing this line, any other thread can enter lock state.



This code line used to generate threads and structured argument arrays.



First line of the code is used to generate a semaphore queue in order to manage threads. The second line of the code defines how many thread can enter the lock state at the same time.



This code line provides that any thread which have finished execution waits other thread until the last thread is finished.



After all threads are executed, the program exits successfully