Wayne State University

CSC 4420 – Semaphore (Lab 05)

Points Possible: 100

Write a multithreaded program to add two 256x256 matrices. The program should use 8 threads and 1-D data partitioning. Thread i is responsible for adding 1-D blocks composed of rows 32 i to 32 i + 31. Each thread should print the statement

Thread i: Done when it completes the local summation (i is the thread id). The input matrices should have integers from 1 to 256 on each row in increasing order. The main thread generates the initial matrices and prints the result in a file called output.txt.

You must include a semaphore to ensure only one thread will modify the result matrix at a time.

A sample program (lab05.c) is added to this assignment. Modify that one accordingly.

```
#include <pthread.h>
#include <stdio.h>
#include <stdlib.h>
#include <semaphore.h>
//declare globe var, such as the three matrices and mutex lock
int matrix1[256][256], matrix2[256][256], result[256][256];

//to declare semaphore
sem_t mutex;

//this func is to write a matrix into a file called "output.txt"
int matrixOutput(int matrix[256][256]){
    FILE *fp=fopen("output.txt", "w+");
    if(fp==NULL){
        printf("could not open file!\n");
        exit(-1);
    }
```

```c
        for(int i=0; i<256; i++){
            for(int j=0; j<256; j++){
                fprintf(fp,"%d ", matrix[i][j]);
            }
            fprintf(fp,"\n");
        }
        fclose(fp);
}


void* matrixAdd(void *i){
        //first need to convert i datatype
        int num=(int)i;

        //put a semaphore so only one thread can modify the result matrix
at once
        _____

        //this thread will do addition for rows from num*32 to num*32+31
        _____


        printf("Thread %d Done!\n", num);

        //to release the semaphore
        _____

        pthread_exit(NULL);
}


int main(){
        //to initialize semaphore

        _____

        //firstly initialize the two matrices
        //both matrix 1 and 2 will be like:
        // 1,2,...,256
        // 1,2,...,256
        // ...
        // 1,2,...,256
        for(int i=0; i<256; i++){
            for(int j=0; j<256; j++){
                matrix1[i][j]=j+1;
                matrix2[i][j]=j+1;
                result[i][j]=0;
            }
        }

        //secondly to create the threads and assign jobs for them
        pthread_t thread[8];
```

```
        for(int i=7; i>=0; i--) {

                _____
        }

        //thirdly use pthread join to make sure all threads are done
before the main thread
        for(int i=7; i>=0; i--) {

                _____
        }

        //lastly to write the result into output.txt
        matrixOutput(result);

        return 0;
}
```

During the submission, provide the screenshot of the output, the output.txt file, and the modified lab05.c file. Please submit all three documents carefully.