//202135574 Jeon Sihyeon(전시현)

/\*\*

 \* File : hw5\_multi\_202135574.c

 \*

 \* Author : Jeon Sihyeon (wjstlgus13@gachon.ac.kr)

 \* Date : 05/25/2022

 \* Partner : I worked alone

 \* Course : Operating System(14463\_003)

 \*

 \* Summary of File :

 \*      This file Get the number of prime numbers from 2 to 2000000

 \*      and display it with using 20 threads

 \*

\*\*/

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <pthread.h>

#define true 1

#define false 0

#define MAX 20

void \*func(void \*val);  //function declaration

/\*\*

 \* int cntPrime(int num)

 \*

 \* Summary of the cntPrime function:

 \*

 \*      cntPrime function checks if the give num is prime number or not

 \*

 \* Parameters : integer number

 \*

 \* Return value : integer number

 \*

 \*

 \*/

int cntPrime(int num)

{

    int cnt = 0;

    int i;

    if(num<2){

        return false;

    }

    for (i=1; i\*i<num; i++) {

        if (num%i == 0) cnt++;

    }

    if (i\*i == num) cnt++;

    if (cnt == 1) {

        return true;

    }

    else return false;

}

int count = 0;

int range = 0;

int eachRange = 0;

struct timespec start, finish;

double elapsed;

pthread\_mutex\_t mutex;

int main(int argc, char\* argv[])

{

    clock\_gettime(CLOCK\_MONOTONIC, &start);

    range = atoi(argv[1]);

    pthread\_mutex\_init(&mutex, NULL);   //initialized

    eachRange = range/MAX;

    pthread\_t tid[MAX+5];

    for(int i = 0; i < MAX; i++){

        pthread\_create(&tid[i], NULL, func, (void \*)i); //thread is being created

    }

    for (int i = 0; i<MAX; i++){

        pthread\_join(tid[i], NULL);     //wait for a thread to end

    }

    clock\_gettime(CLOCK\_MONOTONIC, &finish);

    elapsed = (finish.tv\_sec - start.tv\_sec);

    elapsed += (finish.tv\_nsec - start.tv\_nsec) / 1000000000.0;

    printf("elapsed time: %f sec \n", elapsed);

    printf("The number of prime numbers between 1~%d is %d\n", range,count);

    pthread\_exit(NULL); //terminated main thread

    return 0;

}

/\*\*

 \* void \*func(void \*val)

 \*

 \* Summary of the func function:

 \*

 \*      func counts the number of prime number and increase count in global

 \*

 \* Parameters : void pointer

 \*

 \* Return value : void pointer

 \*

 \*/

void \*func(void \*val){

    int id = ((int)val);

    int start = eachRange \* id;

    int end = eachRange \* (id+1);       //decides the range(start to end) by id

    printf("thread for range ( %d ~ %d)\n", start, end-1);

    for (int n=start; n<end; n++) {

        if (cntPrime(n) == true) {

            /\*applying mutex lock for multi-thread for synchronization\*/

            pthread\_mutex\_lock(&mutex);     //acquire the mutex lock

            //======== critical section =============

            count++;

            //=======================================

            pthread\_mutex\_unlock(&mutex);   //release the mutex lock

        }

    }

    pthread\_exit(NULL); //terminated id th thread

}