





#include <iostream>

//using namespace::std;

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

//    // insert code here...

//    std::cout << "Hello, World!\n";

//    return 0;

//}

#include <stdio.h>

#include <stdlib.h>

#include <pthread.h>

#include <errno.h>

#include <unistd.h>

/\*开启16个线程\*/

#define PTHREAD\_NUM 16

**unsigned** **long** sum = 0;

pthread\_mutex\_t mutex\_t;

**void** \*thread (**void** \*arg) {

**for** (**int** i = 0; i < 10000; i++) {

        // 访问资源的问题，在选择访问thread的时候，加上锁，防止线程冲突

        pthread\_mutex\_lock(&mutex\_t);

        sum += 1;  //不使用原子锁

        pthread\_mutex\_unlock(&mutex\_t);

    }

**return** 0;

}

**int** main(**void**) {

    // 锁的初始化

    pthread\_mutex\_init(&mutex\_t, **NULL**);

    printf("before ...sum = %lu\n", sum);

    pthread\_t pthread[PTHREAD\_NUM]; //被创建线程的标识

**int** ret; //援收返回值

**void** \*retval[PTHREAD\_NUM];

**for** (**int** i = 0; i < PTHREAD\_NUM; i++) {

        ret = pthread\_create(&pthread[i], **NULL**, thread, **NULL**);

        printf("ret:%d\n", ret);

**if** (ret != 0) {

            perror ("cause:");

            printf("creat pthreads %d failed.ln", i+1);

        }

    }

**for** (**int** i = 0; i < PTHREAD\_NUM; i++) {

        pthread\_join(pthread[i], &retval [i]);

        //

        printf("pthread->PP:%p LD:%ld \n", pthread[i], pthread[i]);

    }

    // 释放锁

    pthread\_mutex\_destroy(&mutex\_t);

    printf("after . . . ... sum = %lu\n",sum);

**return** 0;

}