**2020K8009907032**

**唐嘉良**

**作业5**

**5.1 写一个两线程程序，两线程同时向一个数组分别写入1000万以内的奇数和偶数，写入过程中两个线程共用一个偏移量index，代码逻辑如下所示。写完后打印出数组相邻两个数的最大绝对差值。**

**int MAX=10000000;**

**index = 0**

**//thread1**

**for(i=0;i<MAX;i+=2) {**

**data[index] = i; //even ( i+1 for thread 2)**

**index++;**

**}**

**//thread2**

**for(i=0;i<MAX;i+=2) {**

**data[index] = i+1; //odd**

**index++;**

**}**

**请分别按下列方法完成一个不会丢失数据的程序:**

**1) 请用 Peterson 算法实现上述功能;**

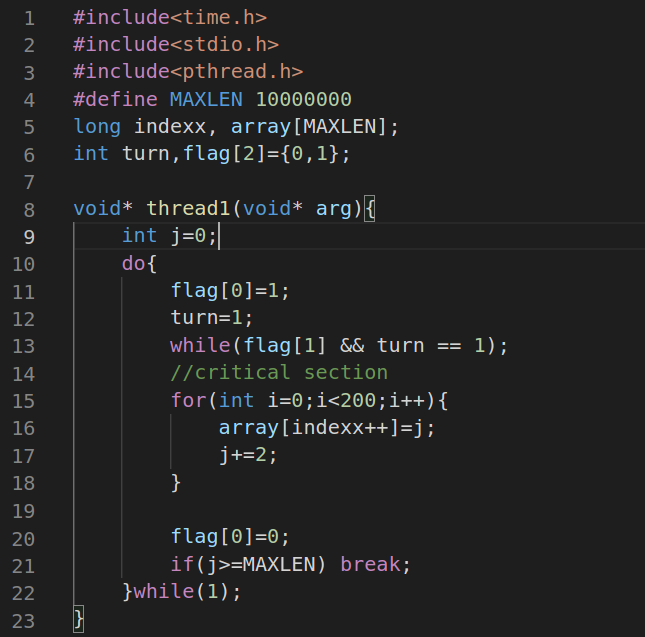
**2) 请学习了解 pthread\_mutex\_lock/unlock()函数, 并实现上述功能;**

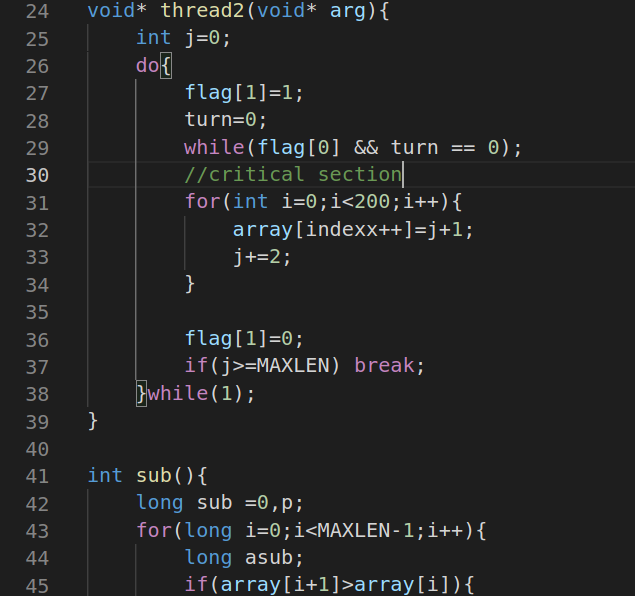
**3) 请学习了解 atomic\_add()( \_sync\_fetch\_and\_add()for gcc 4.1+) 函数, 并实现上述功能。**

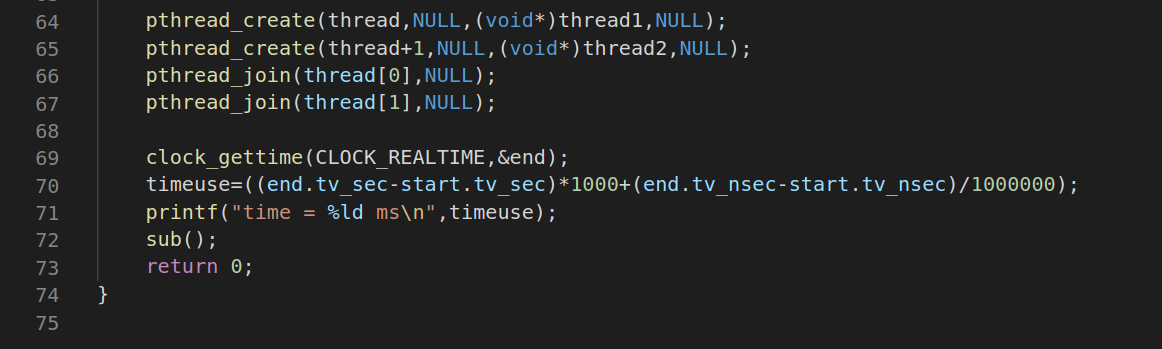
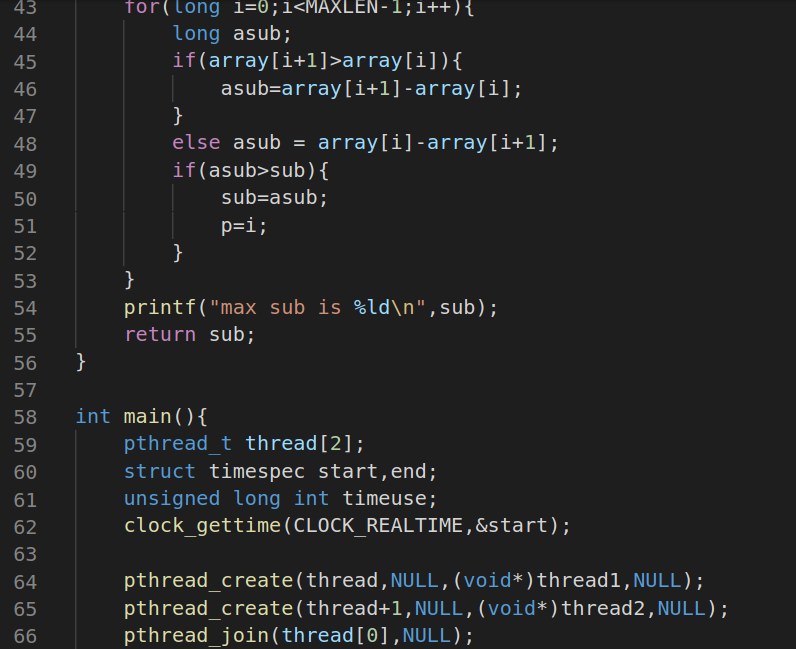
**提交:**

1. **说明你所写程序中的临界区（注意：每次进入临界区之后，执行200次操作后离开临界区。）**
2. **提供上述三种方法的源代码，运行结果截图(即，数组相邻两个数的最大绝对差值)**
3. **请找一个双核系统测试三种方法中完成数组写入时，各自所需的执行时间，不用提供计算绝对差值的时间。**

答：1）Peterson算法源代码如下：（临界区为注释//critical section后的一段代码，以空行为结束分界）



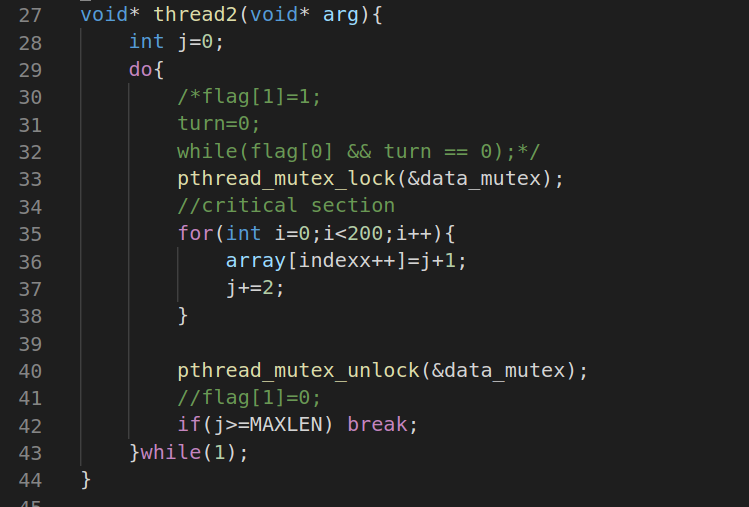




运行结果截图：

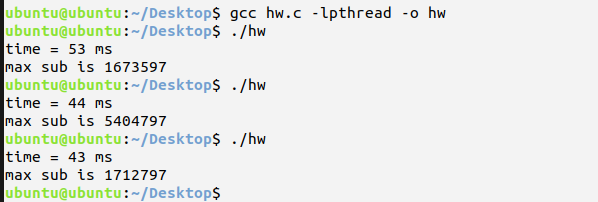


2）方法二代码如下：（临界区为注释//critical section后的一段代码，以空行为结束分界）



其余代码同方法一。

运行结果如下：

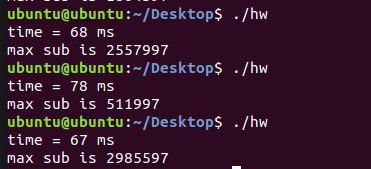


3）方法三代码如下：（临界区为注释//critical section后的一段代码，以空行为结束分界）



其余代码同方法一。

运行结果如下：

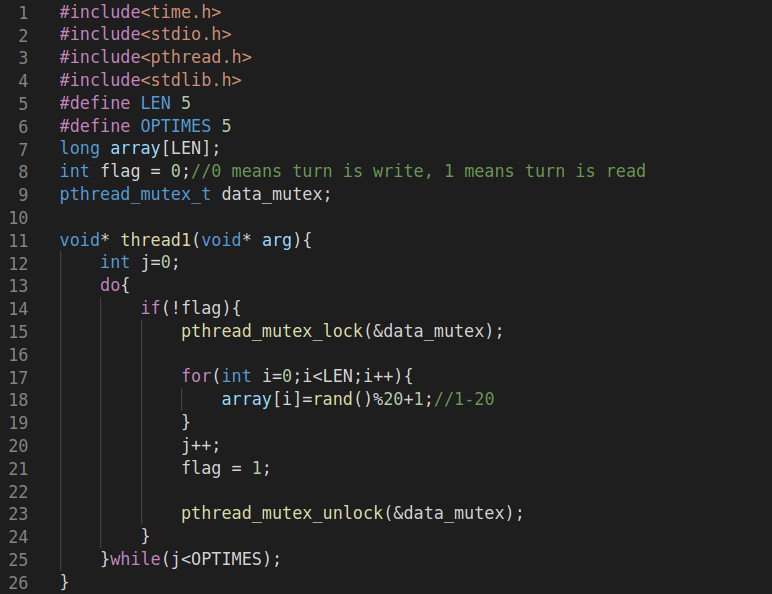


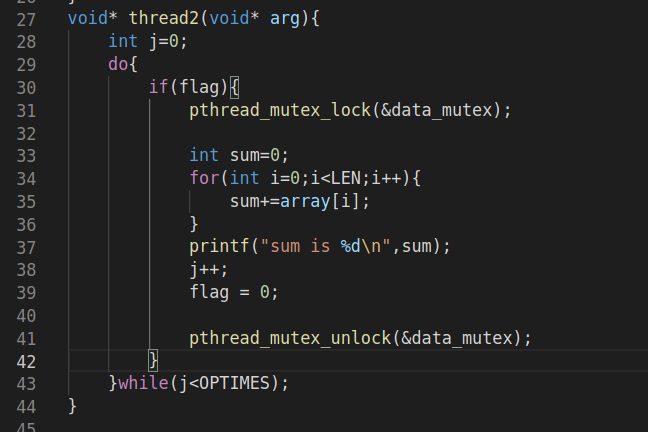
**5.2 现有一个长度为5的整数数组，假设需要写一个两线程程序，其中，线程1负责往数组中写入5个随机数（1到20范围内的随机整数），写完这5个数后，线程2负责从数组中读取这5个数，并求和。该过程循环执行5次。注意：每次循环开始时，线程1都重新写入5个数。请思考：**

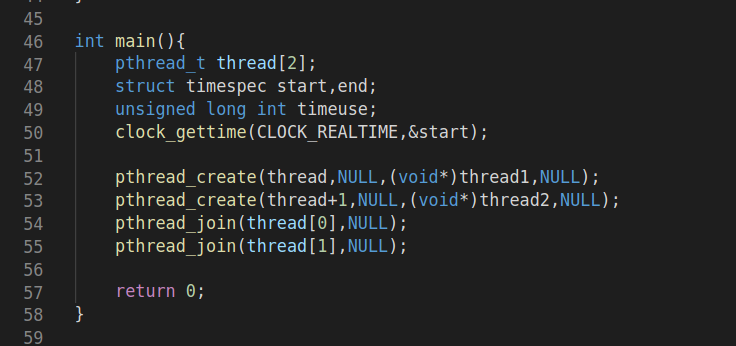
**1）上述过程能否通过pthread\_mutex\_lock/unlock函数实现？如果可以，请写出相应的源代码，并运行程序，打印出每次循环计算的求和值；如果无法实现，请分析并说明原因。**

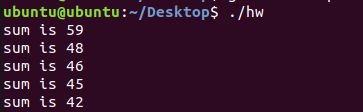
**提交: 实现题述功能的源代码和打印结果，或者无法实现的原因分析说明。**

答：在允许使用flag这类标记变量的时候，能。源代码如下：





运行结果截图如下：



在不允许使用标记变量，仅仅可以使用互斥锁的时候，不能。因为同一个锁只能保证本线程操作的原子性，却没有对线程之间的交互执行有任何要求。