**实验13 线程控制与线程同步**

**实验目的**

* 1. 理解多线程程序的基本原理
  2. 理解Linux系统的pthread库

**实验内容**

### 一、实验8-1线程控制（完成头歌实验，补充命令截图） 第1关创建线程

补全createThread函数，使用pthread\_create函数创建线程，并将start\_routine作为线程处理函数，arg作为线程处理函数的参数，同时将创建成功的线程ID作为createThread函数的返回值。



**第2关线程挂起**

补全waitThread函数，使用pthread\_join函数挂起当前线程，等待指定线程结束，thread为要等待的线程ID号，waitThread函数等待线程成功返回0，失败返回-1。

电脑萤幕的截图

描述已自动生成

**第3关线程终止**

补全cancelThread函数，使用pthread\_cancel函数终止指定的线程，thread为线程要被取消的线程ID号，调用成功返回0，否则返回-1。

电脑萤幕的截图

描述已自动生成

### 二、实验8-2线程同步1（完成头歌实验，补充命令截图） 第1关：互斥锁

1.补全ThreadHandler函数中代码，使用互斥锁对position和buffer变量加锁，使其同一时刻只能被一个线程访问。

电脑萤幕的截图

描述已自动生成

### 第2关：自旋锁

1. 补全ThreadHandler函数中代码，使用自旋锁对position和buffer变量加锁，使其同一时刻只能被一个线程访问。

电脑萤幕的截图

描述已自动生成

### 第3关：条件变量

1.补全ThreadHandler1和ThreadHandler2函数中代码.

ThreadHandler1函数中对position变量进行加一操作(只有一个线程使用，无需加锁)，当position变量被加一后，则只通知一个线程执行ThreadHandler2函数完成字符串赋值操作。

电脑屏幕的手机截图

描述已自动生成

**三、实验8-3线程同步2**

### 第1关：信号量

1.补全ThreadHandler1和ThreadHandler2函数中代码，使用信号量来同步这两个线程(两个线程相互交替执行)，使其执行顺序为ThreadHandler1 > ThreadHandler2 > ThreadHandler1...；

信号量sem1被初始化为1，信号量sem2被初始化为0；

电脑屏幕的手机截图

描述已自动生成

### 第2关：读写锁

1. 补全ReadHandler和WriteHandler函数中代码，使用读写锁对position和buffer变量加锁；
2. 使同一时刻只能有一个线程执行WriteHandler函数，并且没有线程执行ReadHandler函数；
3. 当没有线程执行WriteHandler函数时，允许有多个线程同时执行ReadHandler函数。

电脑屏幕的手机截图

描述已自动生成

**四、设计程序**（选做，要求有源程序和程序运行截图）

1.设一个int型的一维数组，数组长度是1,000,000，计算数组全部元素的和。

要求：

 1)编制一个串行程序，仅有一个线程完成上述计算。

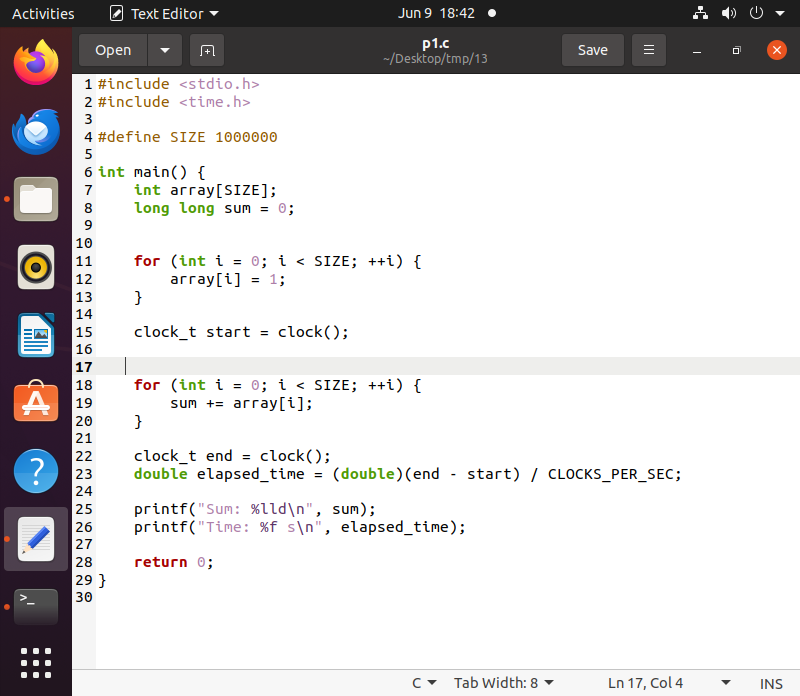
2）编制一个多线程程序，除main线程外，派生两个子线程完成上述计算，main线程负责数据输入和输出计算结果，两个子线程发执行共同计算。

分别获得以上两个程序的计算时间，注意两种情况计算结果应该相同，对比分析。

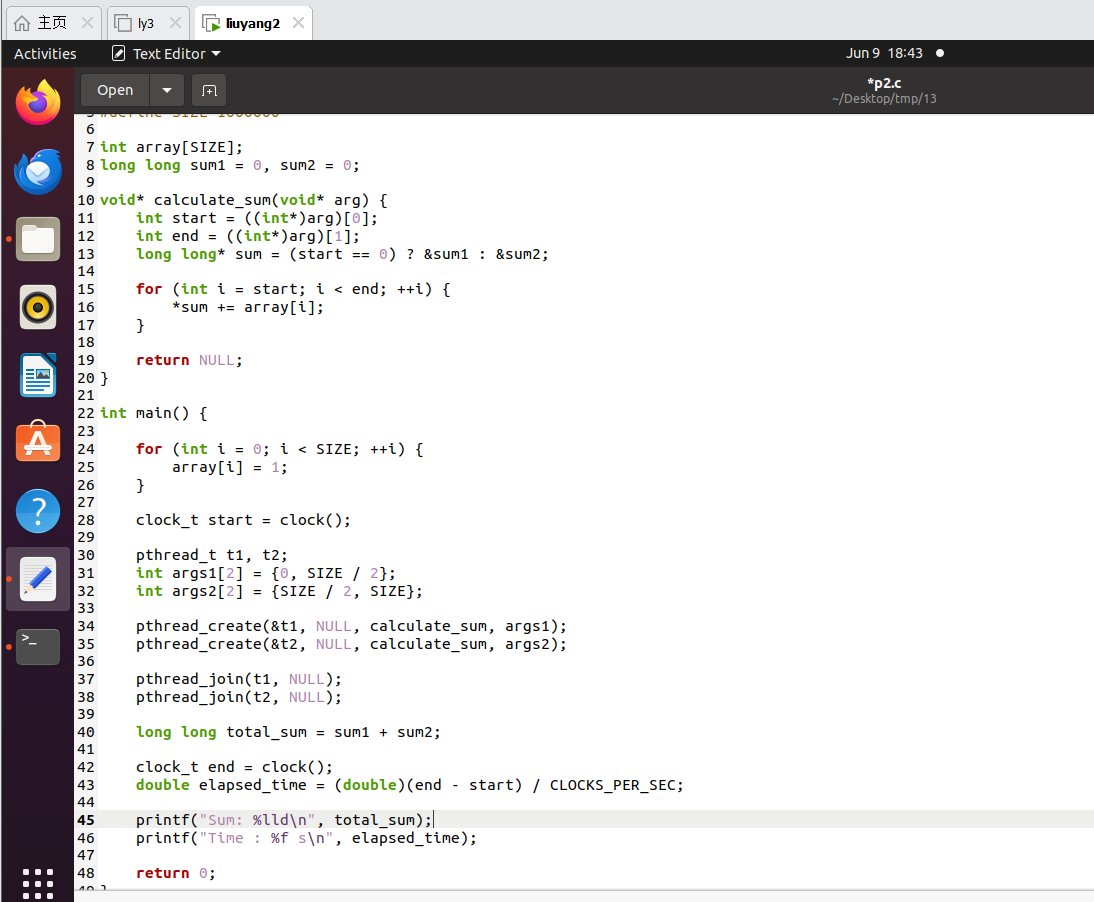
提示：clock()是C/C++中的计时函数（头文件time.h），与其相关的数据类型是clock\_t，通过clock()可以获得当前的系统时间。

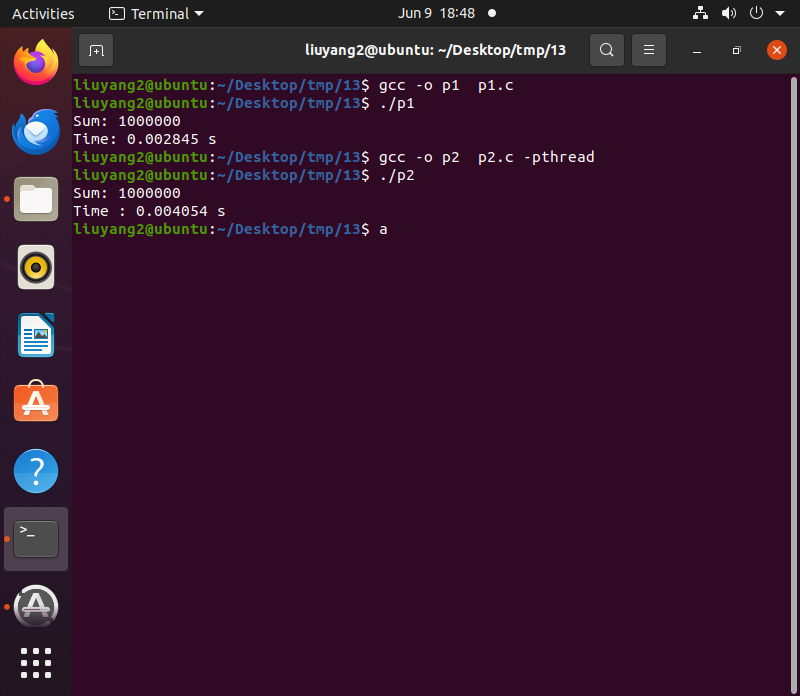
注意：将数组作为共享变量，两个线程并发访问，通过线程的同步方法进行控制。

串行程序：

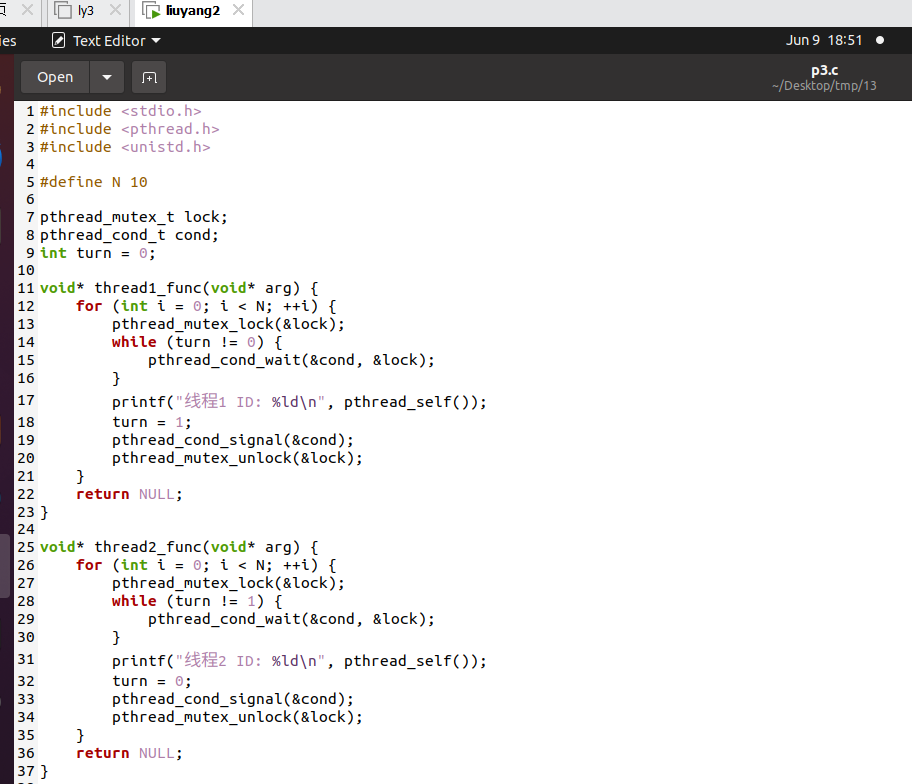


多线程程序：





2.（选做）实现一个有两个线程的程序。每个将自己的ID输出10次，要求输出顺序必须是线程1，线程2，线程1，线程2，线程1，线程2......。



文本

描述已自动生成

