操作系统实验二

实验目的：

1、 掌握 Linux 下 C/C++程序的编译过程；
2、 掌握 Linux 下多线程编程。

实验内容：

1、通过下图中的近似公式，使用多线程编程实现 pi 的计算；
2、通过控制变量 N 的数值以及线程的数量，观察程序的执行效率。

$$\pi = \int_0^1 \frac{4}{1+x^2}dx \approx \sum_{0 \le i \le N} \frac{4}{1+\left(\frac{i+0.5}{N}\right)^2} \times \frac{1}{N}$$

实验步骤：

1、 首先需要搭建 Ubuntu 下 C/C++程序的编译器，这里使用的是 gcc 编译器。

（1） 打开控制台：可以使用快捷键 Ctrl + Alt + T；

（2） 先输入命令 gcc –v 查看是否安装了 gcc（我拷的 ubuntu 10.04 中已经默认安装了），如果没有安装可以在控制台中输入命令进行安装： sudo apt-get install gcc；

（3） 新建并编写测试程序 hello.c 源代码，并将其放置在桌面上：

```
#include<stdio.h>

int main(void)
{
    printf("Hello, world!\n");
    return 0;
}
```

（4） 在控制台中，首先需要将目录设为桌面，输入命令：cd Desktop；再输入 ls，查看是否存在 hello.c 文件；

（5） 在控制台中输入编译命令：gcc hello.c –o hello；成功编译后在桌面上会生成一个 hello 文件；

（6） 在控制台中输入运行命令：./hello 运行程序，可以发现打印出 Hello world!；

2、 学习 Linux 下多线程编程，详细编程模式可见压缩包中书籍的第十二章。

（1） 首先程序中需要添加头文件 pthread.h，并了解 Linux 下的 pthread 多线程编程关键语句，如 pthread_create()、pthread_join()等；

（2） 假设线程数量为 t，则可将上述公式中的 N 平均分为 t 份，每份交给一个线程运行，最后再将计算得到的结果进行累加，在这里需要注意，累加的过程中涉及到了对同一变量的修改，出现临界区的问题，需要加锁进行解决，可使用 pthread_mutex_lock(&lock)和 pthread_mutex_unlock(&lock)。

（3） 编写的程序要求可以输入两个参数：N（公式中的 N，即计算量大小）、t（线程数量），输出同样为两个：Pi 的值、程序的运行时间。N 可以取值：100000、1000000、10000000 等，t 取值可为 2、4、6、8 等，观察同样计算量下不同线程数程序的计算时间，以及线程数为 2 时，不同的 N 计算量下得到的 Pi 值的精确度。

（4） 假设程序名为 test.c，则多线程程序的编译命令为：gcc test.c –o test –lpthread。

（5） 程序中的一些关键语句，仅供参考。

```
pthread_t *threads;
threads=(pthread_t *)malloc(sizeof(pthread_t)*t);
```

```
for(i=0;i<t;i++)
 {
       pthread_create(&threads[i],NULL,thread,(void*)i); // 创建 t 个线程，并将线程标
                                                          记为 i
}


for(i=0;i<t;i++)
{
   pthread_join(threads[i],NULL);  //等待所有的线程结束
}


void *thread(void *ID)   //每个线程需要执行的工作
{
     int id = (int) ID;

    将 N 分为 t 份，则每份长度为 length = N/t，则第 id 个线程计算的范围为
    id*length~(id+1)*length；

    pthread_mutex_lock(&lock);
      将每个线程的计算结果进行累加；
    pthread_mutex_unlock(&lock);

}
```