作业 3

3.1 pthread 函数库可以用来在 Linux 上创建线程，请调研了解 pthread_create，pthread_join，pthread_exit 等 API 的使用方法，然后完成以下任务：

（1）写一个 C 程序，首先创建一个值为 1 到 100 万的整数数组，然后对这 100 万个数求和。请打印最终结果，统计求和操作的耗时并打印。（注：可以使用作业 1 中用到的 gettimeofday 和 clock_gettime 函数测量耗时）；

（2）在（1）所写程序基础上，在创建完 1 到 100 万的整数数组后，使用 pthread 函数库创建 N 个线程（N 可以自行决定，且 N>1），由这 N 个线程完成 100 万个数的求和，并打印最终结果。请统计 N 个线程完成求和所消耗的总时间并打印。和（1）的耗费时间相比，你能否解释（2）的耗时结果？（注意：可以多运行几次看测量结果）

（3）在（2）所写程序基础上，增加绑核操作，将所创建线程和某个 CPU 核绑定后运行，并打印最终结果，以及统计 N 个线程完成求和所消耗的总时间并打印。和（1）、（2）的耗费时间相比，你能否解释（3）的耗时结果？（注意：可以多运行几次看测量结果）

提示： cpu_set_t 类型，CPU_ZERO、CPU_SET 宏，以及 sched_setaffinity 函数可以用来进行绑核操作，它们的定义在 sched.h 文件中。请调研了解上述绑核操作。以下是一个参考示例。

假设你的电脑有两个核 core 0 和 core1，同时你创建了两个线程 thread1 和 thread2，则可以用以下代码在线程执行的函数中进行绑核操作。

示例代码：

```
//需要引入的头文件和宏定义
#define __USE_GNU
#include <sched.h>
#include <pthread.h>

//线程执行的函数
void *worker(void *arg){
   cpu_set_t cpuset;   //CPU 核的位图
   CPU_ZERO(&cpuset);  //将位图清零
   CPU_SET(N, &cpuset);  //设置位图第 N 位为 1，表示与 core N 绑定。N 从 0 开始计数
   sched_setaffinity(0, sizeof(cpuset), &cpuset);  //将当前线程和 cpuset 位图中指定的核绑定运行

   //其他操作
}
```

提交内容：

（1） 所写 C 程序，打印结果截图等

（2） 所写 C 程序，打印结果截图，分析说明等

（3） 所写 C 程序，打印结果截图，分析说明等

3.2 请调研了解 pthread_create，pthread_join，pthread_exit 等 API 的使用方法后，完成以下任务：

（1）写一个 C 程序，首先创建一个有 100 万个元素的整数型空数组，然后使用 pthread 创建 N 个线程（N 可以自行决定，且 N>1），由这 N 个线程完成前述 100 万个元素数组的赋值(注意:赋值时第 i 个元素的值为 i）。最后由主进程对该数组的 100 万个元素求和,并打印结果,验证线程已写入数据。

提交内容：

（1） 所写 C 程序，打印结果截图,关键代码注释等