Project 5:
Designing a Thread Pool
& Producer-Consumer Problem

Project 5-1: Designing a Thread Pool

课本中提供了两种可供选择的语言——C 或者 java。我选择使用 C 语言来完成该项目。

（一）问题分析
我需要补充 threadpool.c 源文件中的几个函数，以及一些其他必要的功能。

（二）实现细节
一、全局变量
1、task_queue[]：任务列表，用数组实现
2、worktodo：当前任务
3、threadpool[]：线程池
4、mutex：任务进出任务列表时使用的互斥锁
5、semaphore：任务执行时选择线程使用的信号量
6、curLEN：当前任务列表长度
7、thread_working[]：线程池中线程的占用情况

二、函数
1、pool_init()
初始化全局变量：当前任务列表长度为 0，线程池中所有线程未被占用，初始化互斥锁和信号量。

2、enqueue()
首先如果任务列表已满，则无法再有新任务进入；如果任务列表未满，需要用互斥锁保证同一时间只有一个任务进入列表并修改 curLEN，保证任务列表中的实际任务个数与 curLEN 一致。

3、dequeue()
同理，如果任务列表为空，则没有任务能被执行；如果有任务列表中有任务还未执行，需要用互斥锁保证同一时间只有一个任务被挪出任务列表并 curLEN，保证任务列表中的实际任务个数与 curLEN 一致。

4、pool_submit()
（1）向线程池提交任务，如果任务列表未满，将该任务加入任务列表。
（2）为任务列表的第一个任务，即下一个要执行的任务分配线程，使用信号量保证同一时间只有一个线程被分配给一个任务，保证一个线程同一时间只执行一个任务，并修改该线程的占用情况。

5、*work()
每一个 thread 在 join 后执行的函数，函数参数已传入执行任务的线程编号，执行完该任务后，修改该线程的占用情况，释放信号量，退出线程。

6、pool_shutdown()
执行完线程池中的所有线程，最后删除互斥锁和信号量。

三、主函数
多添加几个 work 进行实验

（三）运行结果

```
thousanrance@thousanrance-VirtualBox:~/Desktop/Code/OS/project/ch7/project-1/pos
ix$ ./example
Executing in thread 2.
I add two values 50 and 100 result = 150
Executing in thread 2.
I add two values 50 and 100 result = 150
Executing in thread 2.
I add two values 500 and 1000 result = 1500
Executing in thread 2.
I add two values 5000 and 10000 result = 15000
```

Project 5-2: Producer-Consumer Problem

课本中提供了两种可供选择的 API——Pthreads 或者 Windows API。我选择使用 Pthreads 来完成该项目。

（一）问题分析
根据课本给出的 Outline of buffer operation，补充完成必要的功能，需要完善 buffer 的功能、producer 和 consumer 线程的执行函数和主函数。
为了简化 Makefile，我将除了 buffer.h 外的代码写在一个源文件 producer_consumer.c 里。

（二）实现细节
一、buffer
1、全局变量
buffer[BUFFER_SIZE]：缓冲区，用数组实现列表
curSize：记录当前 buffer 中有多少 item

2、insert_item()
如果缓冲区未满，将其添加到列表最后。

3、remove_item()
如果缓冲区不为空，将第一个 item 移除。

二、main()

1. Get command line arguments argv[1],argv[2],argv[3]
首先检查输入是否合法，合法的话，存下这些参数。
sleep_time <- argv[1]: How long to sleep before terminating
num_of_producer <- argv[2]: The number of producer threads
num_of_consumer <- argv[3]: The number of consumer threads

2. Initialize buffer
curSize 置零。初始化互斥锁和信号量。
Producer-Consumer 问题需要三个信号量，mutex、empty、full。
mutex 提供缓冲区访问的互斥要求，初始化为 1；empty 和 full 分别用于表示空的和满的缓冲区空间的数量，初始化为 n 和 0。

3. Create producer thread(s)
使用 pthread_create(),按照输入的参数 num_of_producer 创建线程。

4. Create consumer thread(s)
使用 pthread_create(),按照输入的参数 num_of_consumer 创建线程。

5. Sleep
使用 sleep(),参数为输入的 sleep_time。

6. Exit
使用 pthread_cancel()停止所有线程。
删除互斥锁和信号量。

三、producer and consumer threads
1、*producer()
首先等待随机的时间。
生产者生产 item，然后将其加入到 buffer 中。对于生产者，empty 信号量功能为在 buffer 不满时允许生产者将 item 加入到 buffer 中；full 信号量的功能为告诉消费者有 item 可以消耗；使用 mutex 锁住 insert 的过程，保证生产者和消费者在同一时间只有一个能够访问缓冲区。

2、*consumer()
首先等待随机的时间。
消费者 item，将其加入从 buffer 中移除。对于消费者，full 信号量功能为在 buffer 不空时允许消费者消耗 buffer 中的 item；empty 信号量的功能为告诉生产者 buffe 中有空位可以填入 item；使用 mutex 锁住 remove 的过程，保证生产者和消费者在同一时间只有一个能够访问缓冲区。

（三）运行结果

注：由于运行时间是有限的，所以在运行结束时，consumer 有可能没有消耗完 producer 生产的 item。