要避免同步问题，应尽量不要在线程之间共享数据

**lock语句**

int num = 0;  
object numLock = new object();

for (int n = 0; n < 10; n++)  
{  
 // 使用线程池线程（后台线程）执行任务  
 Task.Run(() =>   
 {  
 // 锁住numLock，确保只有一个线程能进入这段代码  
 lock (numLock)  
 {  
 num++;  
 Console.WriteLine($"当前num：{ num }");  
 }  
   
 });  
}

Console.ReadKey();

**读写锁**

允许多个读线程同时执行，但“读，写线程，写，写线程”不能同时执行

ReaderWriterLockSlim \_rwlock = new ReaderWriterLockSlim();

\_rwlock.EnterReadLock();

\_rwlock.ExitReadLock();

\_rwlock.EnterWriteLock();

\_rwlock.ExitWriteLock();

**Semaphore信号量**

// 设定信号量为2  
int semaphoreCount = 2;  
// 初始化信号  
Semaphore semaphore = new Semaphore(semaphoreCount, semaphoreCount);  
for (int n = 0; n < 10; n++)  
{  
 // 使用线程池线程（后台线程）执行任务  
 Task.Run(() =>  
 {  
 // 阻止当前线程，直到收到信号。  
 semaphore.WaitOne();

num++;  
 Console.WriteLine($"当前num：{ num }");

// 退出信号量并返回前一个计数。  
 semaphore.Release();  
 });  
}