**新建项目**

**计算机学院（软件学院）实验报告**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **学生姓名** | 张继伟 | **学号** | 171530449 | **实验成绩** |  |
| **专业** | 软件工程 | **班级** | 软工四班 | **实验日期** | 19年11月20日 |
| **课程名称** | C#高级程序设计 | | | **任课教师** | 彭伟国 |
| **实验名称** | 线程的使用 | | | **实验序号** | 五 |
| **实验地点** | S409 | **实验台号** | 36 | **指导教师** | 彭伟国 |
| 1. **实验目的及要求**   1．掌握线程的创建与控制，包括创建线程、暂停和恢复线程、终止线程、设置线程执行的优先级。 2．理解实现线程同步方法。 | | | | | |
| 1. **实验内容（或实验原理、实验拓扑）**   1. 尝试开发一个程序，要求通过使用线程休眠控制程序的执行效果。  2. 写两个线程，交替打印1-100，看谁先执行完；写两个线程，分别打印1-100，看谁先执行完。  3. 写两个线程，运用互斥锁（在一个时刻内只允许一个线程进入执行，而其他线程必须等待）原理解决取钱问题。 | | | | | |
| 1. **实验设备与环境** 2. Pc机 3. VS2012 | | | | | |
| 1. **实验设计方案（包括实验步骤、设计思想、算法描述或开发流程等）** 2. **结合课本，对Thread类的学习完成本章节的实验。** 3. **Thread对象被创建后，调用线程对象的Start（）就可以开始执行对应的线程函数。线程函数可以一直执行下去，直到它的结束。可以通过它的IsAlive属性来检查线程的当前状态，可以判定线程是否被取消。** 4. **Sleep（）方法的作用是在一段时间内，让线程处于休眠的状态。这是一种非常有用的方法，因为处于休眠状态的进程不会占用处理器的时间。处于休眠状态的进程可以被中断。被中断设备，就会抛出ThreadInterruptedException异常。** | | | | | |

|  |
| --- |
| 1. **实验结果（包括设计效果、测试数据、运行结果等）**     **3** |
| 1. **实验小结（包括收获、心得体会、注意事项、存在问题及解决办法、建议等）** 2. 通过本节课的实验，对多线程的使用有了一定的了解，结合课本作出了一些多线程的的优点。 3. 多线程的处理可以随时停止任务。 4. 可以分别设置各个任务的优先级以优化性能。 5. 线程的启动和停止，刚开始还不能很好的完成，结合课本知识，对自己的不解进行处理解决。 6. 线程的学习还应更加一步。 |

|  |
| --- |
| 1. **附录（包括作品、流程图、源程序及命令清单等）**   1．  using System;  using System.Threading;  namespace Test\_5  {  class Program  {  //1.尝试开发一个程序，要求通过使用线程休眠控制程序的执行效果。  static void Main(string[] args)  {  Program obj1 = new Program();  Thread thread1 = new Thread(new ThreadStart(obj1.Count));  thread1.Name = "线程1";  Program obj2 = new Program();  Thread thread2 = new Thread(new ThreadStart(obj2.Count));  thread2.Name = "线程2";  Program obj3 = new Program();  Thread thread3 = new Thread(new ThreadStart(obj3.Count));  thread3.Name = "线程3";  thread1.Start();  thread2.Start();  thread3.Start();  Console.ReadKey();  }  private int cnt = 0;  private void Count()  {  while (cnt < 10)  {  cnt++;  Console.WriteLine(Thread.CurrentThread.Name + "数到" + cnt);  Thread.Sleep(100);  }  }  }  }  2．using System;  using System.Threading;  namespace \_5\_2  {  class Program  {  static void Main(string[] args)  {  /    Program obj1 = new Program();  Thread thread1 = new Thread(new ThreadStart(obj1.Count1));  thread1.Name = "线程1";  Program obj2 = new Program();  Thread thread2 = new Thread(new ThreadStart(obj2.Count1));  thread2.Name = "线程2";  thread1.Start();  thread2.Start();  Console.ReadKey();  }  private static int cnt1 = 0;  private int cnt2 = 1;  private void Count1()  {  while (cnt1 < 100)  {  cnt1++;  Console.WriteLine(Thread.CurrentThread.Name + "数到" + cnt1);  Thread.Sleep(100);  }  }  private void Count2()  {  while (cnt2 < 100)  {  cnt2++;  Console.WriteLine(Thread.CurrentThread.Name + "数到" + cnt2);  Thread.Sleep(100);  }  }  }  }  3．  using System;  using System.Threading;  namespace \_5\_3  {  class Program  {  public static int balance = 3000;  public static object o = new object();  static void Main(string[] args)  {  Program p = new Program();  Thread t1 = new Thread(p.GetMoney);  t1.Name = "t1";  Thread t2 = new Thread(p.GetMoney);  t2.Name = "t2";  t1.Start(1000);  t2.Start(500);  Console.ReadKey();  }  public void GetMoney(object num)  {    lock (o)  {  int a = (int)num;  balance -= a;  Thread.Sleep(100);  Console.WriteLine(Thread.CurrentThread.Name + ":" + balance);  }  }  }  } |