/\*程序功能：

\* 缓冲区只能容纳一个字符。写一个，读一个。

\* 如果没有同步措施，上一次写的还未等到读出，就被刷掉，会导致少打印若干字

\*/

namespace InterLockedDemo1 {

class Program {

private static char buffer;//缓冲区

private static long numberOfUsedSpace = 0;//缓冲区的容量，初始化是0

static void Main(string[] args) {

Thread writer = new Thread(delegate () {

string sentence = "大漠孤烟直，长河落日圆";

for(int i = 0;i < sentence.Length;++i) {

while(Interlocked.Read(ref numberOfUsedSpace) == 1) {

Thread.Sleep(10);

}

buffer = sentence[i];

Interlocked.Increment(ref numberOfUsedSpace);

}

});

Thread reader = new Thread(delegate () {

for(int i = 0; i < 11;++i) {

while(Interlocked.Read(ref numberOfUsedSpace) == 0) {

Thread.Sleep(10);

}

Console.Write(buffer);

Interlocked.Decrement(ref numberOfUsedSpace);

}

});

writer.Start();

reader.Start();

}

}

}

/\*system.Threading命名空间的Interlocked类控制计数器实现同步

\* read（）读取计数器的值

\* increment（） 计数器+1

\* decrement（） 计数器-1

\* add（） 计数器增加指定值

\* exchange（） 设置计数器为指定值

\* Interlocked类的操作几乎都能用一条指令实现，保证了原子性

\*/