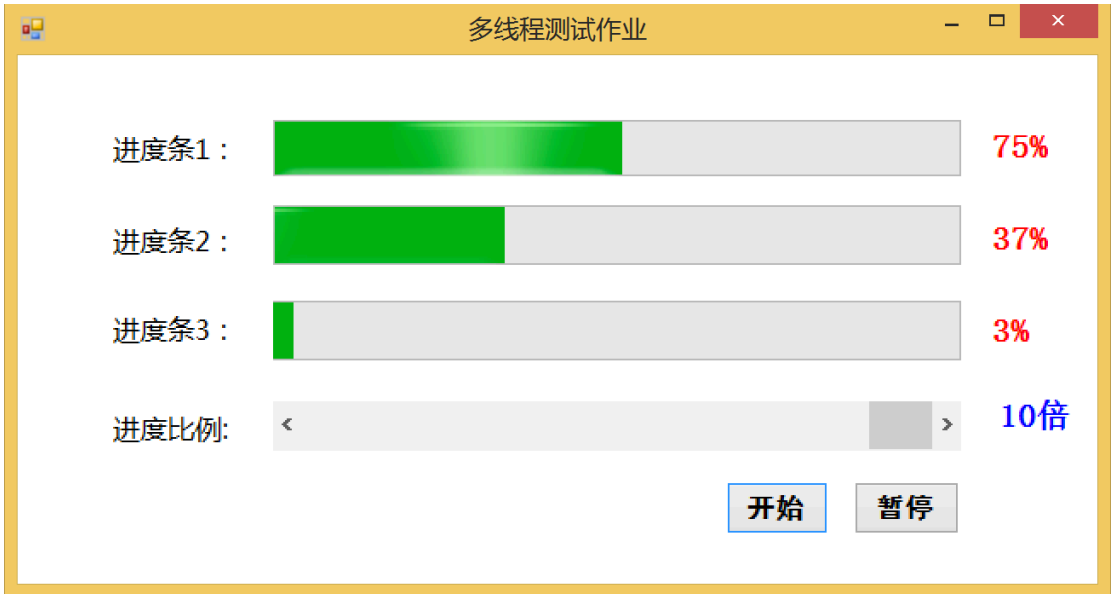
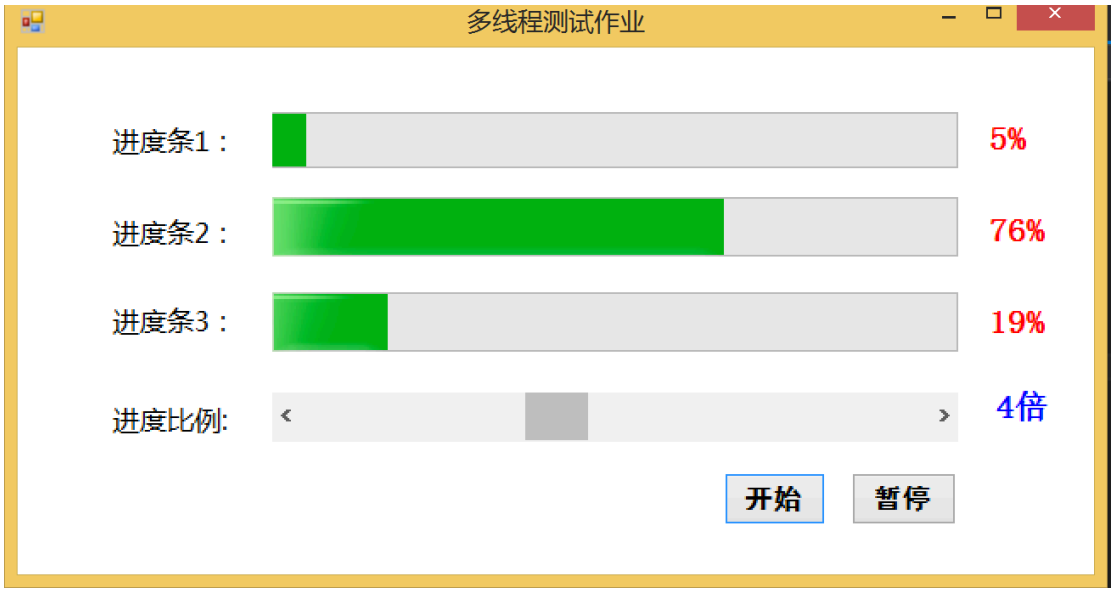
线程和异步编程作业说明

本次作业要求实现一个可变速的控制条



实现如上图所示。

本次作业实现了调控进度比例的功能，可以通过拖动进度条进行实现。



最终 100 完成界面

同时还实现了暂停功能，按下暂停键后可以让进度条全部停止。再次按下开始后会重新开始。

以下为关于实现的介绍：
本系统通过实现四个线程多线程变法进行实现。

```
    //时间线程
    timekeeper = new TimeKeeper();
    Thread timeT = new Thread(new ThreadStart(timekeeper.runMethod));
    timeT.Start();

    //第一个进度条线程
    myProgressBar1 = new MyProgressBar(1,timekeeper);
    myProgressBar1.setProgressBar += setProgress1Value;
    t1 = new Thread(new ThreadStart(myProgressBar1.runMethod));
    t1.Start();

    //第二个进度条线程
    myProgressBar2 = new MyProgressBar(2,timekeeper);
    myProgressBar2.setProgressBar += setProgress2Value;
    t2 = new Thread(new ThreadStart(myProgressBar2.runMethod));
    t2.Start();

    //第三个进度条线程
    myProgressBar3 = new MyProgressBar(3,timekeeper);
    myProgressBar3.setProgressBar += setProgress3Value;
    t3 = new Thread(new ThreadStart(myProgressBar3.runMethod));
    t3.Start();
```

四个线程分别负责时间计数器和第一二三个控制条。
控制条类是自己实现的，其中因为要控制界面控件，因此进行了委托

```
public class MyProgressBar
{
    //声明一个委托
    public delegate void setProgressBarValue(int value,int number);
    public setProgressBarValue setProgressBar;
    int number;
    public bool isStop = false;
    TimeKeeper t;
```

而在这个进度条进程中，我们规定每隔 50ms 去读一次时间值

```
public void runMethod()
{
    while (!isStop)
    {
        Thread.Sleep(50);
        lock (t)
        {
            setProgressBar(t.getTime(), number);
        }
    }
```

这里用到了线程同步的知识，我们将 timekeeper 这个类锁定起来，保证了线程的同步。

具体的设置进度条的方法如下所示：

```
private void setProgress2Value(int value, int number)
{
    if (this.progressBar2.InvokeRequired)
    {
        MyProgressBar m = new MyProgressBar(number);
        m.setProgressBar += setProgress2Value;
        this.Invoke(m.setProgressBar, new Object[] { value, number });
    }
    else
    {
        if ((value / beishu) > 0 && ((value / beishu) % 100) == 0)
        {
            this.progressBar2.Value = 100;
            this.label5.Text = "100%";
        }
        else
        {
            this.progressBar2.Value = (value / beishu) % 100;
            this.label5.Text = ((value / beishu) % 100).ToString() + "%";
        }
    }
}
```

我们先判断这个请求是不是当前界面线程发出的，如果不是就要用 invoke 方法，就可以在子线程中安全的更新界面的控件。

至于调比率的代码部分，我设置了一个变量

```
int beishu = 10;
```

```
private void hScrollBar1_Scroll(object sender, ScrollEventArgs e)
{
    label8.Text = ((hScrollBar1.Value+10)/10).ToString() + "倍";
     beishu = (hScrollBar1.Value+10) / 10;

}
```

当控制条变化时，我们的变量值也会发生改变，从而界面上的进度条也会发生相应变化。

以上就是有关本次作业的实现。
如果助教有任何和本次作业有关的疑问请联系我

141250136 王家玮
wangjiawei0227@163.com
谢谢助教