**实验八 多线程机制**

**一、实验目的**

1. 掌握线程的4种状态
2. 掌握用thread类创建线程的方法
3. 了解挂起、恢复和终止线程的方法

**二、 实验内容和过程**

1、验证书中例子6，注意学习其中使用Thread类创建线程的过程，并回答下列问题：

（1）复习书中使用Thread类创建线程的过程，并回答此例子中创建**线程的目标对象**是哪个对象？

（2）阅读程序，找到其中student和teacher线程新建、运行、中断所对应的语句，并将答案写入实验报告。

     2、将下列程序代码填写完整，理解线程的四种状态：ThreadExample.java

class Tortoise extends Thread{

   int sleepTime=0,liveLength=0;

   Tortoise(int sleepTime,String name,int liveLength)   {

      this.sleepTime=sleepTime;

      this.liveLength=liveLength;

      setName(name);

   }

   public void run()   {

      while(true)      {

          liveLength--;

          System.out.print("@");

          try            {

              【代码1】//让线程调用 sleep方法进入中断状态，sleepTime毫秒后线程重新排队

                                                                 // 等待CUP资源。

            }

          catch(InterruptedException e) {

            }

           if(liveLength<=0)   {

                System.out.print(getName()+"进入死亡状态\n");

                【代码2】//结束run方法的语句。

            }

      }

   }

}

class Rabbit extends Thread{

   int sleepTime=0,liveLength;

   Rabbit(int sleepTime,String name,int liveLength)   {

      this.sleepTime=sleepTime;

      this.liveLength=liveLength;

      setName(name);

   }

   public void run()   {

      while(true)  {

          liveLength--;

          System.out.print("\*");

          try   {

              【代码3】//让线程调用 sleep方法进入中断状态，sleepTime毫秒后线程重新排队

                                                                 // 等待CUP资源。

            }

          catch(InterruptedException e) {

            }

           if(liveLength<=0)  {

                System.out.print(getName()+"进入死亡状态\n");

                【代码4】//结束run方法的语句。

            }

      }

   }

}

public class ThreadExample{

   public static void main(String args[ ]) {

       Rabbit  rabit;

       rabit=【代码5】    //新建线程rabit。

       Tortoise tortoise;

       tortoise=【代码6】  //新建线程tortoise

       【代码7】         //启动线程tortoise。

       【代码8】        //启动线程rabit。

   }

}

3、编写一个应用程序，在主线程中用thread类的子类创建2个线程，2个线程共享一个int型的数据，并各自有自己独占的数据。模板如下，将程序填写完整。

BankExample.java

class Bank implements Runnable{

    【代码1】 //声明一个int型变量money，初值为100。

    Thread zhang,keven;

    Bank() {

      【代码2】   //创建zhang，Bank对象为zhang的目标对象。

       zhang.setName("会计");

      【代码3】   //创建keven，Bank对象为keven的目标对象。

       keven.setName("出纳");

    }

    public void run()  {

       【代码4】 //声明一个int型变量i，初值为0。

        while(true)   {

           if(【代码5】){  //判断线程zhang是否正在占有CUP资源。

              i=i+1;

               money=money+1;

               System.out.println(zhang.getName()+"将money的值改为"+money);

               System.out.print(zhang.getName()+"的局部变量i="+i);

             if(i>=6) {

                     System.out.println(zhang.getName()+"线程进入死亡状态");

                      【代码6】 //使得线程zhang进入死亡状态。

                  }

               try{

                     Thread.sleep(1000);

                  }

               catch(InterruptedException e) {

                  }

            }

           else if(【代码7】) { //判断线程keven是否正在占有CUP资源。

               i=i-1;

               money=money-1;

               System.out.println(keven.getName()+"将money的值改为"+money);

               System.out.print(keven.getName()+"的局部变量i="+i);

               if(i<=-6)   {

                     System.out.println(keven.getName()+"线程进入死亡状态");

                     【代码8】 //使得线程keven进入死亡状态。

                  }

               try{

                     Thread.sleep(1000);

                  }

               catch(InterruptedException e)   {

                  }

             }

         }

    }

}

class BankExample{

    public static void main(String args[ ])  {

        Bank bank=new Bank();

        bank.zhang.start();

        bank.keven.start();

    }

}

4、创建两个线程A和B，A线程打印输出你的学号5次，每次输出之间睡眠1秒，B线程打印输出你的姓名5次，每次输出之间睡眠1秒。

5、验证书上例子7，掌握线程同步的基本原理。回答：

（1）线程同步的作用是什么？

（2）JAVA中线程同步的机制是什么？

6、设计两个线程， 一个充当电子表， 每隔 1 秒显示下一系统时间； 另一个充当闹钟， 每到整点就开始报时， 即显示 5 次整点提示， 同时将第一个线程挂起，报时完毕再将第一个线程恢复运行。

**三、提交各实验程序源代码及实验报告**