* **Thread类的有关方法**
* **void start()**：启动线程，并执行对象的run()方法。
* **run()**：线程在被调度时执行的操作。
* **String getName()**：返回线程的名称。
* **void setName(String name)**：设置该线程名称。
* **static currentThread()**：返回当前线程。
* **void yield()**：调用此方法的线程释放当前CPU的执行权。
* **void join()**：在A线程中调用B现成的额join()方法，表示，当执行到此方法，A线程停止执行，直至B线程执行操作完毕，A线程再接着join()之后的代码执行。
* **boolean isAlive()**：判断当前线程是否还存活。
* **void sleep(long l)**：显式的让当前线程睡眠l毫秒。
* **线程的调度**
* 调度策略：

1. 时间片
2. **抢占式：高优先级的线程枪战CPU**。

* Java的调度方法
* 同优先级线程组成先进先出队列（先到先服务），使用时间片策略。
* 对高优先级，使用优先调度的抢占式策略。
* **线程的优先级**
* 线程的优先级控制
* **MAX\_PRIORITY（10）**;
* **MIN\_PRIORITY（1）**;
* **NORM\_PRIORITY（5）**;
* 涉及的方法：
* **getPriority()**：返回线程优先值。
* **setPriority(int newPriority)**：改变线程的优先级。
* 线程创建时继承父线程的优先级。

public class TestThread1 {  
  
 public static void main(String[] args) {  
 //3）创建一个Thread的子类对象  
 SubThread1 st1 = new SubThread1();  
 st1.setName("子线程1");  
  
 //4）调用线程的start()方法，启动此线程：调用响应的run()方法  
 //一个线程只能执行一次。  
 //不能通过Thread实现类对象的run()方法去启动一个线程。  
 st1.setPriority(Thread.***MAX\_PRIORITY***);  
 st1.start();  
  
 Thread.currentThread().setName("==========主线程");  
 for (int i = 0; i <= 100; i++) {  
 System.***out***.println(Thread.currentThread().getName() + ":" + Thread.currentThread().getPriority() + ":" + i);  
  
// if (i % 10 == 0) {  
// Thread.currentThread().yield();  
// }  
// if (i == 20) {  
// try {  
// st1.join();  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }  
// }  
 }  
 System.***out***.println(st1.isAlive());  
 }  
  
}  
  
/\*\*  
 \* 1）创建一个继承于Thread的子类  
 \*/  
class SubThread1 extends Thread {  
  
 /\*\*  
 \* 2）重写Thread类的run()方法，方法内实现此子线程要完成的功能。  
 \*/  
 @Override  
 public void run() {  
 for (int i = 0; i <= 100; i++) {  
// try {  
// Thread.currentThread().sleep(1000);  
// } catch (InterruptedException e) {  
// e.printStackTrace();  
// }  
 System.***out***.println(Thread.currentThread().getName() + ":" + Thread.currentThread().getPriority() + ":" + i);  
 }  
 }  
}