多线程

- 经验
  - 这里面sleep函数放在循环里面 一般好像run方法里面都是循环
- join效果图
  - 运行效果
  - join位置
  - 补充
- 分支主题
- 分支主题
- 分支主题
- 分支主题
- 分支主题
- Thread.join
  - 放在t1和t2上面 中间 下面 结果都不同 以及t1和t2只有一个运行和两个都运行结果也都不一样 （！！！join必须要紧接着放在当前执行的线程，以起到隔断的作用。！！！）才能达到图片的效果
- lock死锁例子
  - 方法
  - 相互关联
  - 打印 标注：例如ThreadA.join表示进程让给进程A去执行只有执行完进程A其他进程再来执行
- lock死锁解锁---重要易错点
- thread睡眠
  - 推荐使用TimeUnit.MILLISECONDS.sleep(10);代替thread.sleep
- finally知识点+lock锁
  - 如果产生死锁的话finally就不会执行到 卡死进程
- callable和runable的区别
  - callable(要有返回值 callable后要有泛型 比如String 进行线程实现需要调用futuretask函数 实现的是call方法 以及如果不多次调用futuretask函数就会单条线程执行到底 例如三条线程需要new三次futuretask 可在futuretask后加上泛型 例如String 但多次实验发现即使多次new该futuretask 交替执行并没有那么明显 现在推荐使用runable') runable没有返回值 实现的是run方法 常用
- 建议使用runable创建多线程
  - 扩展 并发与并行的区别
    - 1. 什么是并发?
      - 指同一时刻，有多个指令在单个CPU上交替执行
    - 2. 什么是并行?
      - 指同一时刻，有多个指令在多个CPU上同时执行