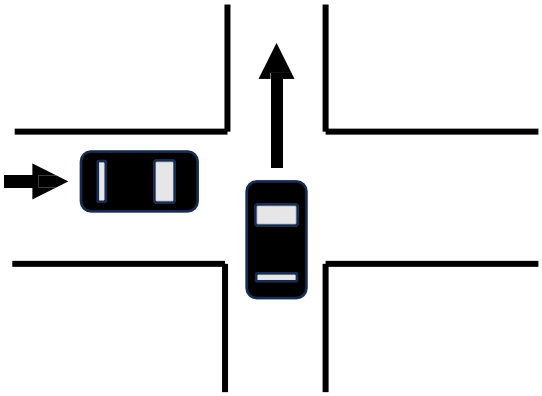
# JC2002 Java 编程 - 实践 8（第 10 天）

本实践的目的是熟悉多线程、定时事件和线程同步。在本实践中，我们的任务是利用定时事件实现一个简单的交通模拟器。假设有两条单行道交叉，一条由南向北，另一条由西向东。当一辆车到达十字路口时，除非该路口已被另一辆车占用，否则该车将继续前行。我们假设一辆车过马路需要两秒钟。当十字路口清空后，另一辆等待的汽车（如果有的话）可以进入十字路口。



1. 实现主类 **TrafficSimulator** 和两个继承自 **Thread** 的类 **WestArrivalThread** 和 **SouthArrivalThread**，模拟汽车从西面和南面到达。为避免重复实现所有功能，可选择实现一个继承自 **Thread** 的联合超类 **ArrivalThread**。

   我们可以假设，来自两个方向的汽车会在 0.5 到 4 秒之间随机到达。因此，两个线程的 **run()** 方法都应包含一个无限循环，在 500 和 4000 毫秒之间随机休眠一段时间，然后分别打印文本 "汽车从西面到达 "或 "汽车从南面到达"，并重复执行。

   *提示：* 通过实例化类 **java.util.Random**，然后调用方法 **nextInt(n)**。

   在 **TrafficSimulator** 的 **main()** 方法中，实例化并启动西行和南行到达的线程。运行程序以确保其按预期运行。

2. 实现继承自 **Thread** 的 **CrossingThread** 类，以模拟汽车穿越十字路口。每次有汽车从西面或南面到达十字路口时，都应实例化并启动该线程。在 **TrafficSimulator** 中实现同步公共静态方法 **cross()**，该方法休眠 2000 毫秒后返回。在 **CrossingThread** 类的 **run()** 方法中，应调用 **TrafficSimulator.cross()**。通过同步，我们可以防止两辆车同时进入交叉口。

3. 最后，实施一种机制来跟踪从西面和南面有多少辆车在排队等候通过路口。为此，您需要在 **TrafficSimulator** 中使用静态整数变量来存储两个队列中的汽车数量。当有新的汽车到达时（从到达线程中），应增加汽车数量、

并在休眠 2000 毫秒后，即返回之前减少方法 **cross()** 中的汽车数量。因此，你需要建立某种机制，告诉方法 **cross()** 到达的汽车是来自西面还是南面。

运行程序，测试其是否正常运行。每次数值发生变化时，都应打印队列中的车辆数。您可以尝试使用不同的最大和最小到达时间间隔，看看它对队列的发展有何影响。