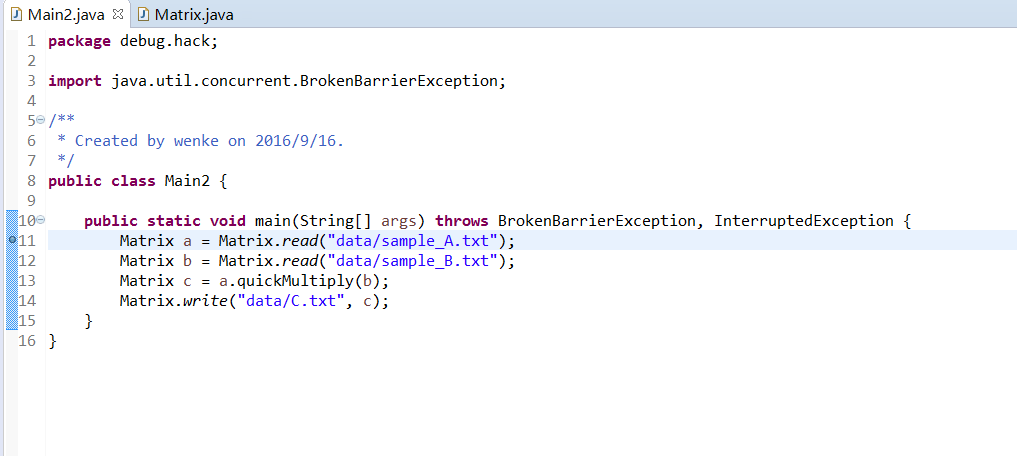
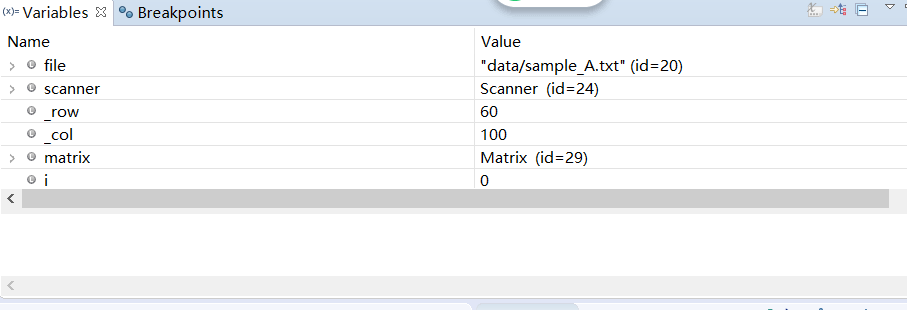
# JAVA Debug Hacks

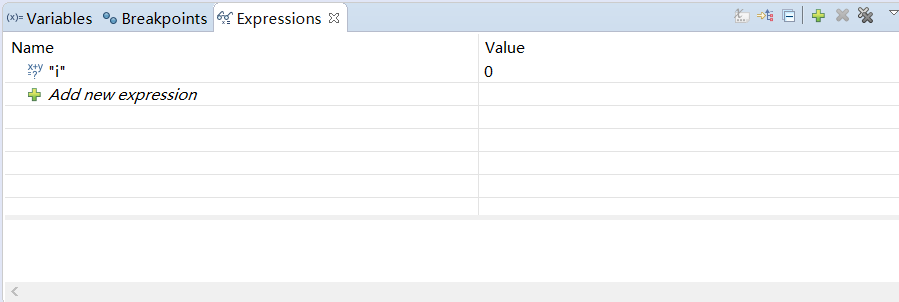
1. Add watching point:



Purpose:让程序执行到这一步的时候停下，通过f5或f6调试后面的语句看清分析清程序内部的代码，找出程序的bug

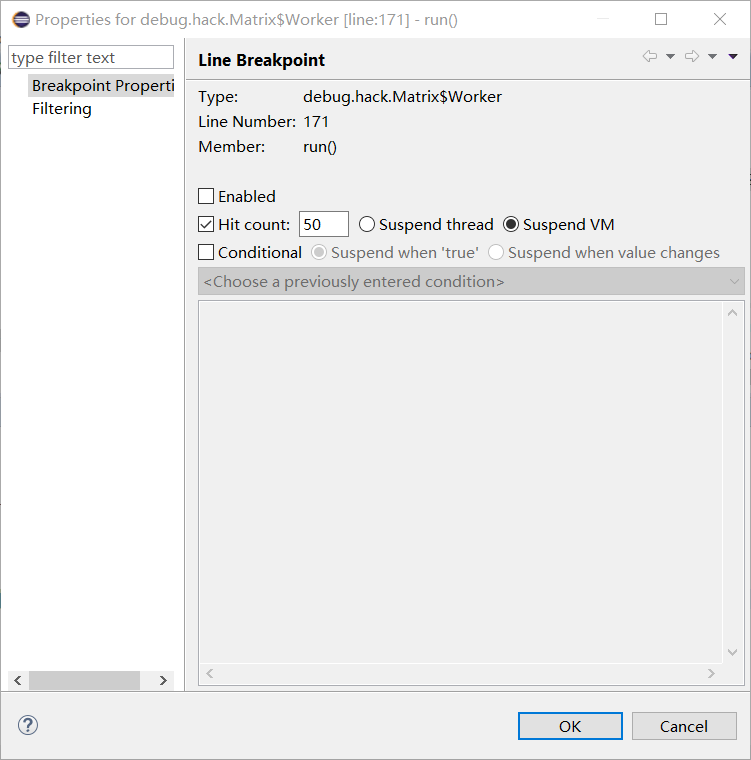
1. modifying variable:





Purpose:监视i变量值的变化，更好地进行程序调试

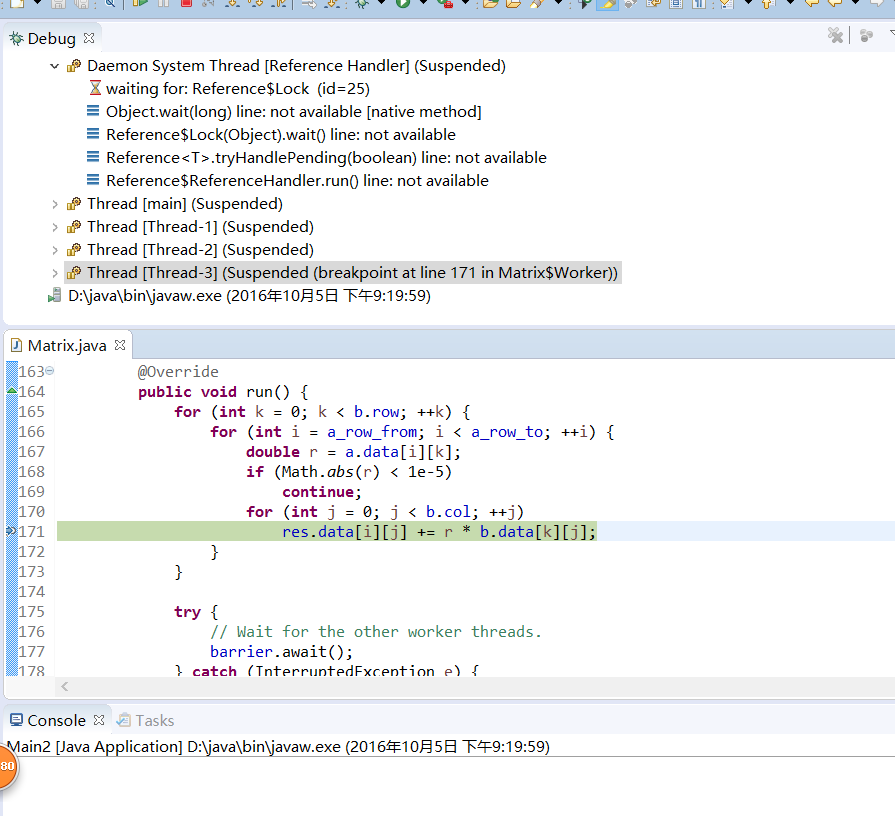
1. Conditional breakpoint和 thread switching

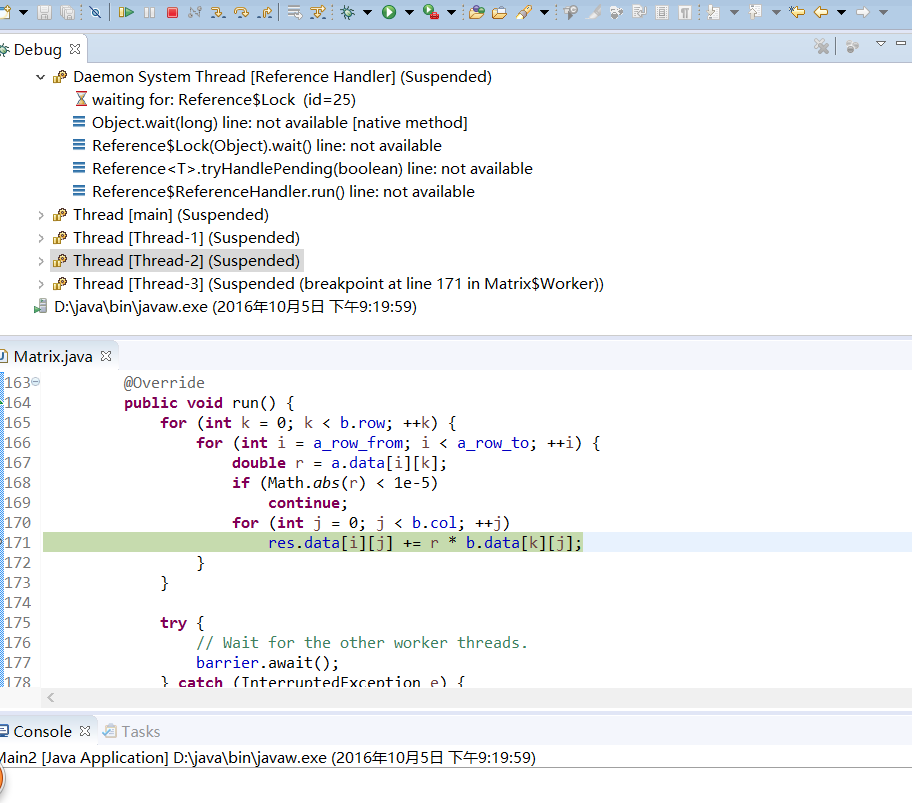
设置条件断点：

Purpose:以防需要重复按f6调试循环体，简化并精确化对循环体或者其他类似结构的调试过程

多线程调试：

原线程：



选中thread-2，按F8进行线程切换：

Purpose:进行线程的调试

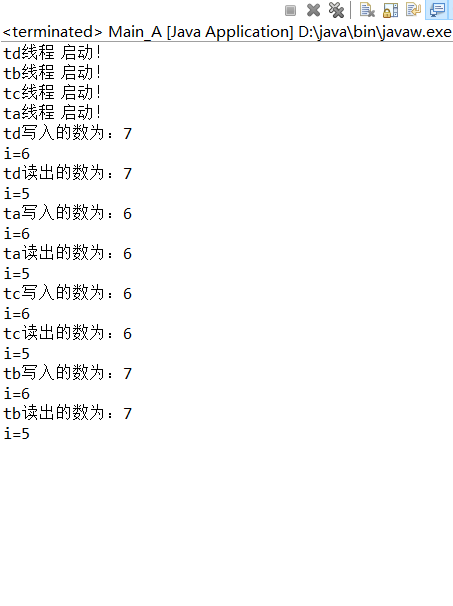
1. CyclicBarrier的作用：等到所有线程都达到某个公共屏障后，才允许他们共同执行某个任务。换句话说，CyclicBarrier就像它的名字一样，可以看成一个障碍，所有参与的给定数量的线程都到齐后才能一起通过这个障碍。

Barrier值是5而不是4的原因：需要等待所有线程都执行完后，再return result.

把它变为4的结果：quickMultiply中的barrier.await()失效，for循环一结束，就返回了result.但最终C.txt中输出的文档不变

# Doubly Linked List

Requirement A 运行结果：

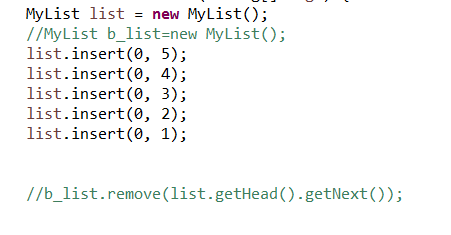


Requirement B运行结果：



Requirement C 说明：

源代码的问题在于：



b\_list可以删list的节点，所以在MyList.java里面这样操作（把remove和insert方法中参数的Node类型int类型，再通过for循环去找那个相应的节点，这样实例化后的对象就只能对自己的节点进行操作），可以达到自己的链表只能删自己的节点的功能，代码如下：

