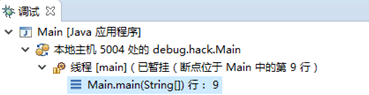
**Objective A**

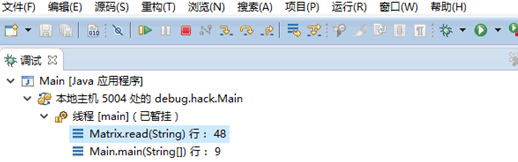
1. Start debugging

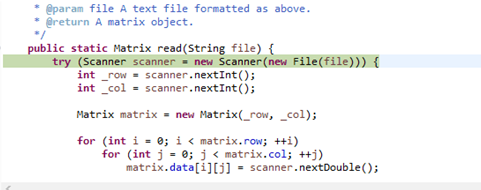


1. Step into

Step to the next line executed

This is the button of it. 

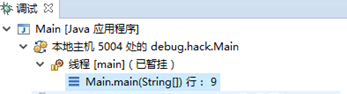


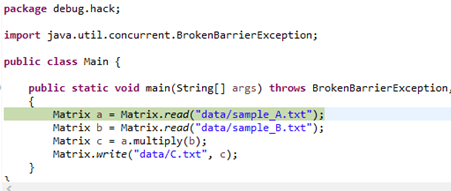


1. Step out

Step to the first line executed after running from this method.

This is the button of it. 

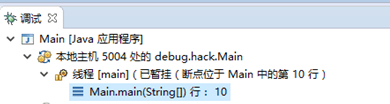


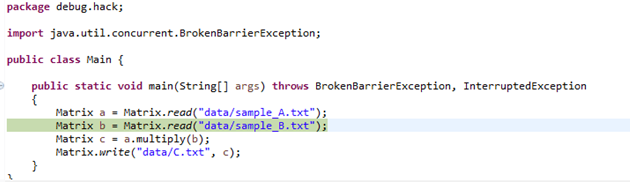


1. Step over

Step to the next line in this file.

This is the button of it. 

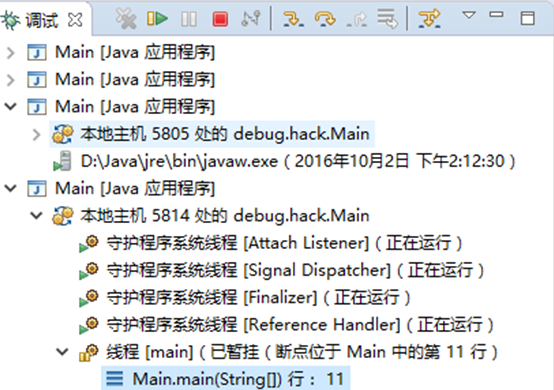
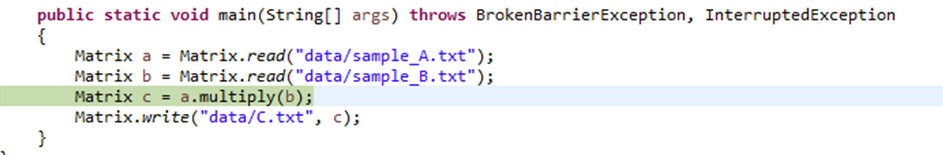




1. Resume Execution

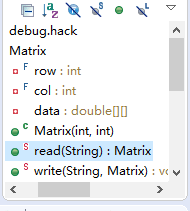
Otherwise,if you want to jump straightly to a breakpoint(if there is no breakpoint is waiting for the end of the program),so we would use resume execution.

This is the button of it. 

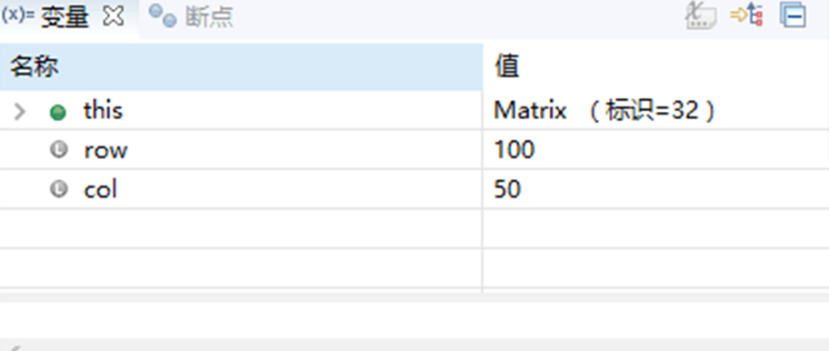
1. Call Stack

View the call stack is also a important work in the process of debugging , it helps to clarify function calls between relations .

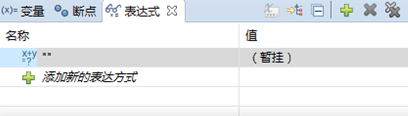


1. Monitor

monitoring window

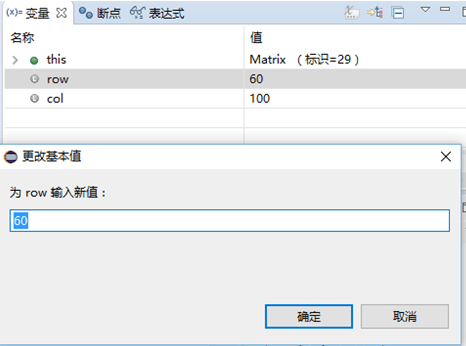


1. Computational expressions

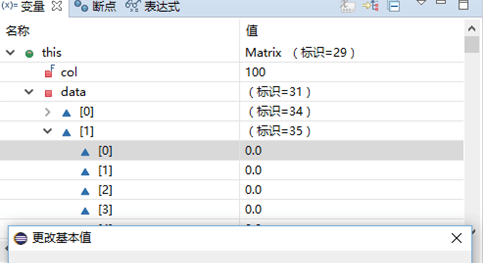


1. Temporary change variables

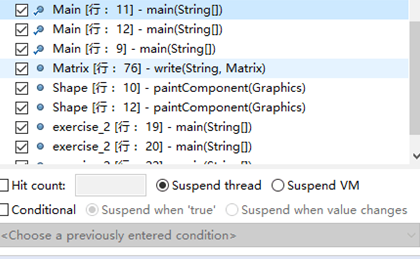
Set Value



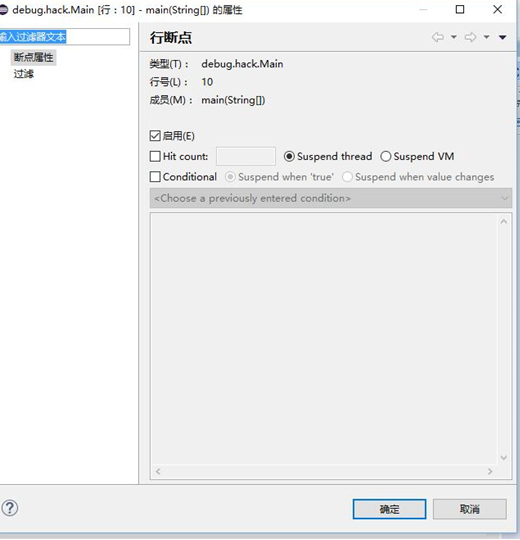
For the array or object , you can open it step by step the small triangle , until we find at the bottom of the base type .



1. Set a breakpoint properties

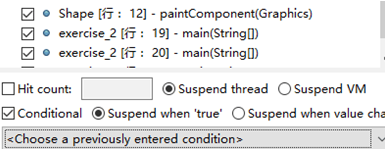


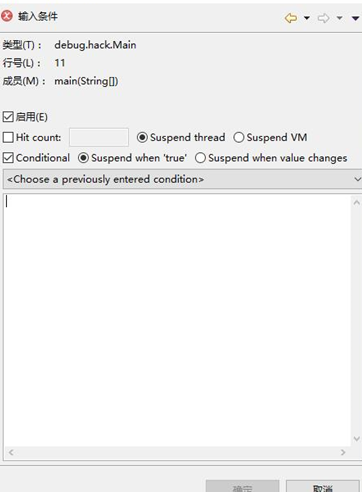
In idea



Through this interface , we can give a breakpoint set thread hung Suspend XXX,Condition, Evaluate and log or Remove once hit and so on.

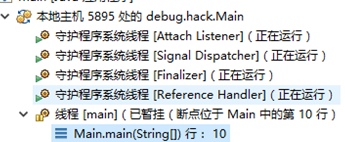
1. Conditional breakpoint

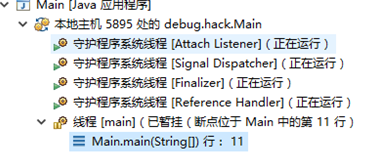
Conditional breakpoint is to give the breakpoint additional a Boolean expression.

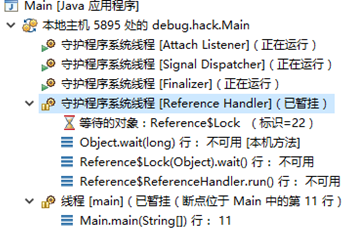


1. Thread testing

If you opened a multithreaded program , you can in the drop-down menu of the call stack switch do you want to see threads , at the same time automatic switch stack , watch window , etc.







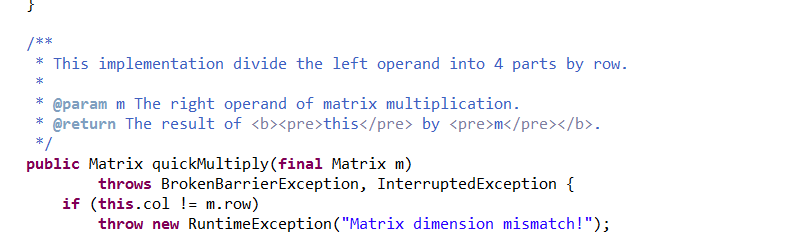
**Objective B**

**In function quickMultiply, a****CyclicBarrier serve for a meeting of multiply-Thread.**

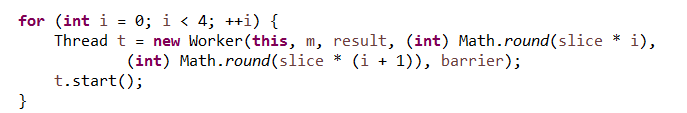


**When many Thread run at a same time, there is a Barrierto be needed,it’s funtion is to wait all the thread reach the same point,then run together.**

*Function quickMultiply() divide the matrix to 4 part.*



**There are 4 threads.**



**And before it there is one thread.**

**So there are 5 threads.**

**If the barrier value is 4 instead of 5 ,the process will return before end.**