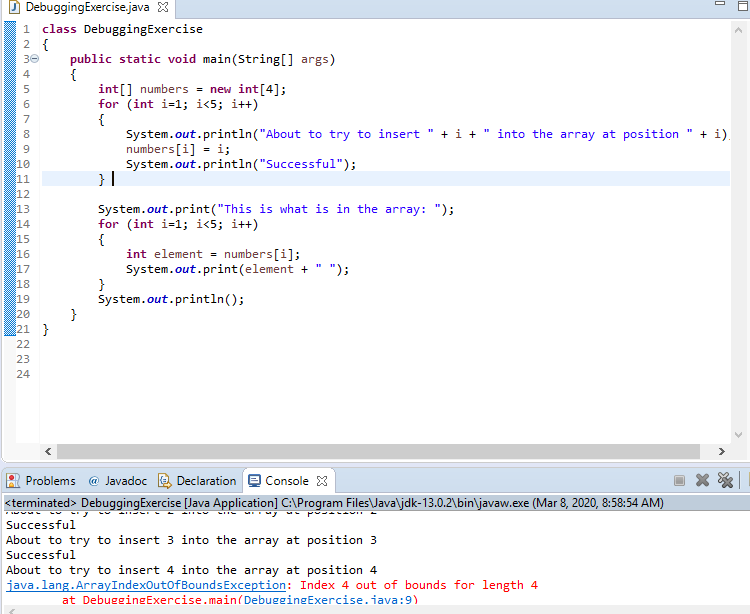
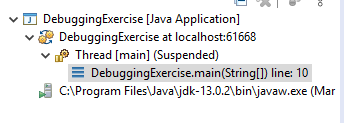
I started debugging in Eclipse IDE for JAVA developers…I just thought I’d document how everything worked and if everything was debugging correctly. There were several programs that I have tested using the Eclipse Debugger.

I ran the DebuggingExercise.java. It runs 1, 2, 3 correctly…but when it gets to length 4 (the array contains 4 items, and it only reads the first 3 in the array.)



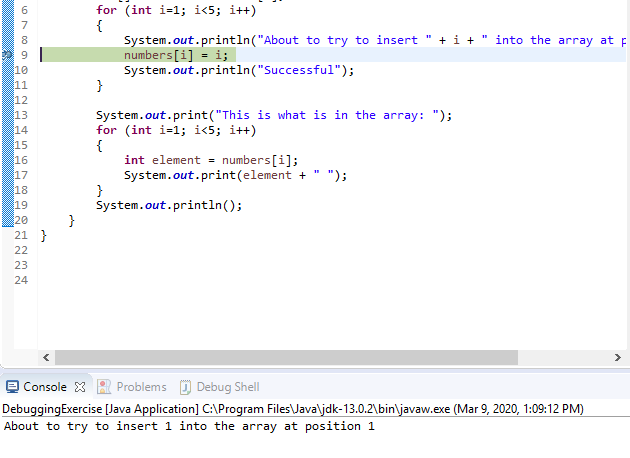
I’m going to set some breakpoints and debug and see what the problem might be.

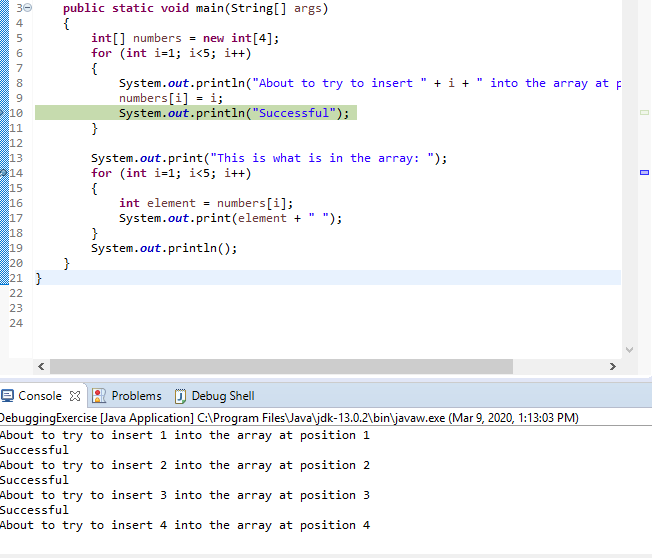
The compiler found an error on line 9 with the index of #4 in the array being “Out Of Bounds.” I put a breakpoint on line 9 and ran again, the program halts at line 10.



So far it isn’t indicating to me there is an error because i = 1, so it says insert “i” into the array at position “i” and the value of i (written in the for loop) is 1. So ‘1’ will be substituted in for i whenever ‘i’ in the println statement for the first for loop ‘about to try to insert’

I am going to step over this logic….





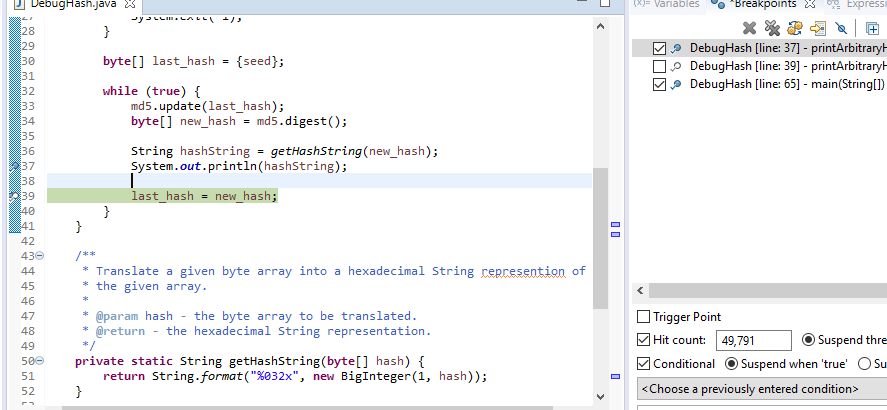
I think I found where the problem may be – I needed to step over the logic, once I stepped over line 9 breakpoint, finally the other parts to the array are run through the program execution.

SOLUTION: It makes sense as there are four items in the array, but the for loop runs to 5….a correct for loop for this would be:

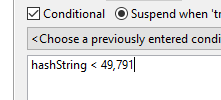
*for (int i = 1; i < 4; i++)*

in BOTH loops.

We open up a new file called DebugHash.java, the file runs an infinite loop of arbitrary hashes. We need to figure out the 49,791st hash value that will be printed. Maybe we need to set a breakpoint hit counter **to stop at 49,791 times the breakpoint on the statement that prints those hashes is reached.**



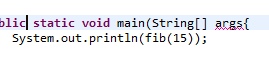
I tried the hit count, but maybe I’ll set a conditional end point to suspend when hashString gets to the 49,791st value.



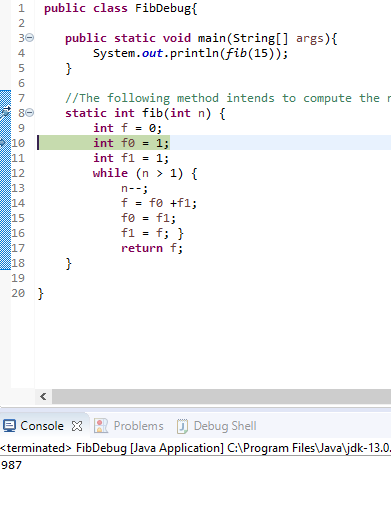
SOLUTION: Adding a for loop to keep printing hash values until “i < 49791.” That way, when the loop prints out the 49,791st hash value, the program will halt running the print loop.

Next file is FibDebug.java – we are looking to compute the n-th Fibonacci number, but the program has a bug that will require the use of the Eclipse debugger to fix the problem.

(First things first I see….I FOUND A MISSING PARENTHESES in the main…)



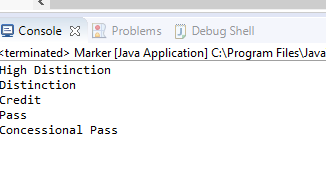
I can add that in and then debug the other errors.

987 is actually the 16th Fibonacci number. Output should read 610.

Maybe the loop is going too many times…

SOLUTION:

Next lab is the Marker.java – ONLY ONE message should be printed for the grade marker, let’s see what the output gives us when we compile and run

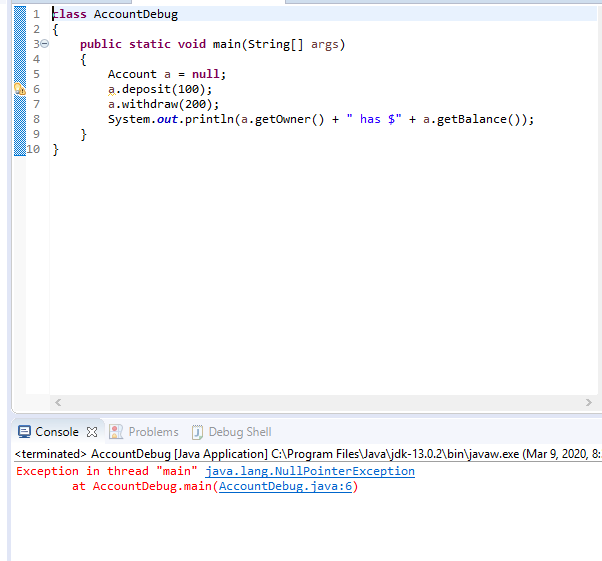


I put a breakpoint on each if statement (stepping over each if), but it would still output multiple lines.

I thought “this must be a logic error, I’ll try adding brackets {} to the if statements that the breakpoints were set at

SOLUTION:

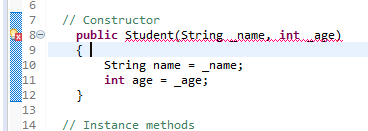
I added the Account and AccountDebug



I might need to set some breakpoints here.

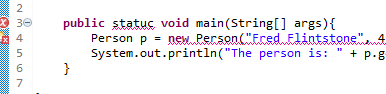
SOLUTION:

We now added person.java and persondebug. I found an error being highlighted right off the bat



I need a return statement here.

Then we can start adding breakpoints.



That’s not how you spell static. I’ll fix them with the debugger.