**Computer Science 271 – Fall 2017**

**Lab #7**

**Lab: Thursday, 10/19/17 in HS 101C**

**Due: Thursday, 10/19/17 at 3:00 PM in D2L**

**Objective:** Use of a built-in debugger in some Java IDE with a graphical user interface, e.g., NetBeans.

You are asked to debug a simple Java class that manipulates a 2D array. You should debug the class and provide well-written answers to a series of questions related to the debugging process. If problems are uncovered during the debugging process, you will need to fix them and describe the fix. **Submit written answers to the questions on this MS Word document**; you don’t have to submit the code that you have fixed. Answers to questions are deemed to be “correct” or “incorrect” by the instructor.

The completed debugging exercise should show that you have attained the learning outcomes listed below:

1. Set breakpoints on and trace a Java program using the debugger in some Java IDE.
2. Identify software bugs using the debugger in some Java IDE.
3. Understand, at a high-level, the general functions of a debugger.

**This lab uses the BuggyQuilt.java sample code on D2L. Download it, and a copy of this lab document (in MS Word) to begin the lab.**

This lab uses the BuggyQuilt.java sample code. Create a new project in NetBeans and add BuggyQuilt.java to it. Fill in the blanks with the answers to each question. Be specific!

Give descriptions in terms of program execution and stopping. Your answer should indicate why somebody would use such a thing (not “it turned green and red”). In general, your answers should be concise, well-written, in complete sentences, with correct spelling and grammar and in a **different font color**. Your answers do not need to fit entirely in the space given (you can add extra space to this document if you would like). **DO NOT put your name on this document anywhere.**

1. Run BuggyQuilt.java and note the runtime error. Now, put a breakpoint on the line:   
   char[][] myQuilt = new char[3 \* myBlock.length][4 \* myBlock[0].length];

in the main method of BuggyQuilt.

1. Debug BuggyQuilt.java. Describe what happens:  
     
   Debugger stopped at line 57. We can see the variables that are available at this time. This is useful as we can see what is available and what should be created.
2. Do a “Step Over (F8)”.

Describe what happens:

Program stopped running due to an uncompilable source code. Created a myQuilt array that can be seen under Variables.

1. Do a “Continue (F5)” and describe what happens:

Throws an ArrayIndexOutOfBoundsException. This means that we are trying to access outside the bounds of an array from the previous line

1. Debug BuggyQuilt.java again. Now do two “Step Over (F8)”   
     
   commands. Describe what happens:

When you step over in line 61 the step over goes to the next java method that is called in this line. In this case it goes to java.util.Random(). If I continue to step over it will execute this method.

1. How do you determine, without printing, to which element in myQuilt the ‘!’ was assigned?

You can continue to step over and look at what values are returned under variables. In my case the first bracket returned a 6 and the second returned a 5. Then you can check under myQuilt, and look at [6], then look at [5] inside of that and you should see the !.

1. Press the “Terminate” button. Then debug BuggyQuilt.java again. Then do “Step Over (F8)” 2 times. Then do “Step Into (F7)” once. Then do “Step (OVER/INTO)” until the program crashes.

Where does the program first crash (give a specific line number or line of code)?

Line 124 inside theBuggyQuilt

1. What is the reason for this first crash?

ArrayIndexOutOfBoundsException

We are trying to access outside the bounds of an array.

Fix the bug you just identified and proceed.

1. Use the debugger to help identify the remaining bugs in the BuggyQuilt program. List the remaining bugs, including before/after code, perhaps with line numbers, and how you fixed them (be as specific yet concise as possible):

Line 122 Added a flippedRow--. It was trying to access outside the array because it was using array length instead of the index.

Line 106 There was an =< which I changed to an < so that it does not access beyond the array

Line 108 the for loop is going outside of the array unable to fix in the allotted time.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. In general, if a breakpoint is In general, during the debugging process, what does the green line (and/or green arrow) in NetBeans indicate?

The current line or method that we are stopped at.

1. In general, if a breakpoint is on a line without a method call and the program is run to that point, what is the difference between doing “step over” and “step into” at that line:

Nothing. If there is no method to step into then it will just step over it.

1. Consider the following runtime error message:  
   Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 17

at MyClass.foo(MyClass.java:92)

at MyClass.main(MyClass.java:37)

Java Result: 1

Where is the best place to set a breakpoint?

Line 92 as the debugger will automatically stop at that line where the error first occurs

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. True or false: In general, use the debugger window to watch variable values while the program is executing
2. True or false: In general, step forward from a breakpoint on a method call, using Step if you think the method call is fine and using Step Into if you think there may be a problem with a method call

True

1. True or false: In general, while stopped (e.g., breakpoint or stepping), check if variables have expected values.

True

**Submission**

To complete this lab, submit your answers, **typed up in this document in red font** (in .DOCX format), to the Lab 7 drop box on D2l by 3:00 PM on Thursday, 10/19. No late submissions will be accepted.