**CIS163 Lab 2**

**Debugging and Reference lab:**

**Preparation:**

* Have completed lab 1 from last week
* Attended class and stayed up to date with class material

**Objectives** (after completing the lab you will be able to do:)

* Use the Eclipse IntelliJ Debugger to help with your projects
* Use references in debugging

**Why/Purpose: Every developer must be able to effectively use a debugger and perhaps it will shorten your development time for project 1. 😊**

**Activities: (Do All Steps And Associated *Task’* )**

**Watch instructor doing part of this lab. (*Task’* Take notes) – Recorded**

1. ***Task’*** Using IntelliJ create a project named: Labs with a package named lab1 and do the following:
   1. Copy over, and open the SimpleClass and Example files using IntelliJ
   2. Copy over, and open the open the SimpleClassTest using IntelliJ and remove all errors
      1. DO you remember how to do that?
2. ***Task’*** Using your art talent draw pictures of the different objects created by the example.java class and determine what the output is (up to line 26). Now, run the program and verify your output.

(use some drawing programs, such as paint, MS whiteboard, etc)

C1🡪 3 🡨 (c1 local)

C2🡪 5 🡨 (c2 local)

OUPUT:

Line 1: 6 4

Line 2: 3 8

Line 3: 6 8

Line 4: 6 11

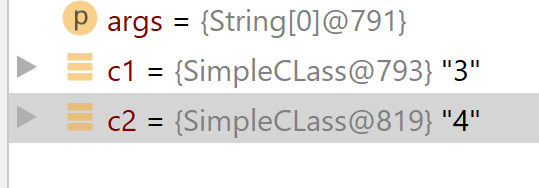
1. Using the debugger (place a breakpoint at line 15 of the Example class and execute with the debugger) and the “step into” function, count the number of clicks it takes to execute the program. ***Task’*** Number of clicks: \_\_121\_\_ (if greater than 50 write >50)
2. Using the debugger and the step over function count the number of clicks it takes to execute the program. ***Task’*** Number of clicks: \_13\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Does it make sense to you, what the differences are between step 4 and step 5? ***Task’* (YES,NO)**

Yes, it does make sense to me what the differences are between steps 3 and 4.

1. Using the debugger (break at line 15 of Example.java) and the picture above, ***Task’*** write down the actual reference number to each of your objects above by stopping the program at the appropriate places (see below), and then next to each arrow above, ***Task’*** write a reference number next to that arrow.

Example:



C1(793) 🡪 3 🡨 (c1 local (793))

C2 (819) 🡪 5 🡨 (c2 local (819))

C1 = 810

C2 = 827, at line 15, it becomes 845,

(***Task’*** Attempt to do this with the local variables too)

1. Using the debugger, stop the program at line 28 (Example.java) and change the values of c1 and c2 to 2 and 3 respectively. Does your code make this output? ***Task’*** Yes or NO

Output:

Line 1: 6 4

Line 2: 3 8

Line 3: 6 8

Line 4: 2 6

Yes it does

1. Using the generate tool within IntelliJ, create set and get methods for the static variable named numberOfInst in the SimpleClass. Now, from the SimpleClassTest class, create a test case that (1) sets that static variable to zero, (2) creates four new SimpleClass objects, (3) calls the get method to see if numberofOfInst is now equal to four (for this step use an assertEqual method)

***Task’*** What is your value \_0\_ and what does the value mean? It means that the assert equals is fine.

***Task’* Run in Debug mode and step through the code.**

1. Using the debugger, create a conditional break point at line 38 in SimpleClassTest file so that it stops when the value of s.value == 20. Execute SimpleClassTest. ***Task’*** Did your code stop there? Yes or NO

It stopped, but s was not equal to 20, using step over or 5 next steps, it was then equal to 20

1. ***Task’*** Run all the test cases found in SimpleClassTest class, did they all execute correctly. If not, then place a break point at the assertion that failed. Using the step into method, correct the problem.

If time is remaining (optional) Now open your project 1 with your partner and see what works and does not work. Using the debugger, start fixing project 1 issues.