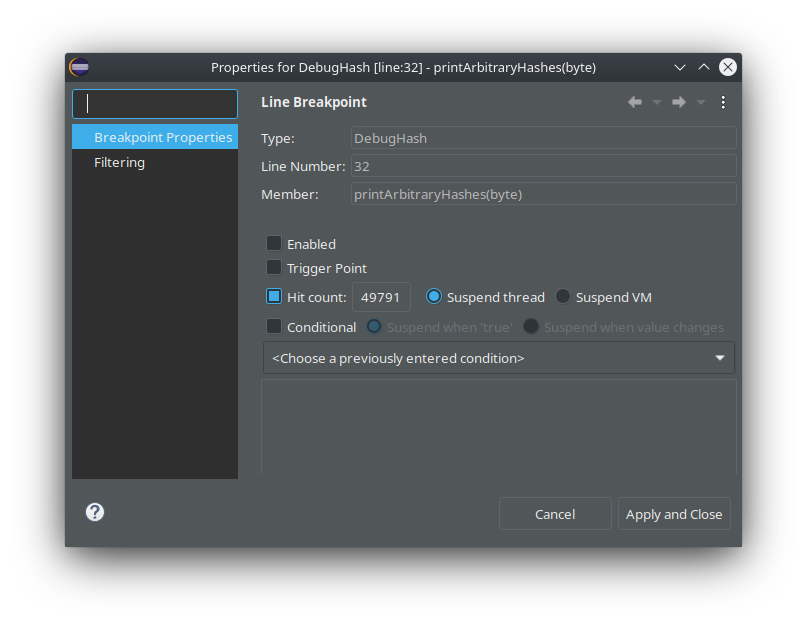
Patrick Temple

Prof. Whitener

CSCI165

21 February 2020

1. Created project.
2. Added file.
3. Array out of bounds exception.
4. Lines 5 – 11: I set a break point at line 6 with a hit count of 4. After I tried to step over line 9, it threw the array out of bounds error. The loop needed to be changed from 1 to 5 into 0 to 4. The array size was also changed to 5. When the numbers were added, I changed line 9 to numbers[i] = i + 1 so it would set from 1 to 5 in arrays 0 to 4.  
    Lines 13 – 19: I stepped through these lines, and it was getting 2 through 5, not 1 to 5. The int element value is also a little clunky, so I figured I would add it to the print statement in place of element. Since it ran only four times, I changed the loop from int i=1; i<5; i++ to int i = 0; i <= 4; i++.
5. The answer (unique every time) is 2a84296c6a45c4734bbe39beebb670ea. I got there by making a break point at line 32, right clicking the break point and selecting Break Point Properties. Then, I added 49,791 to the hit count and obtained the number.
6. I thought that line 3 may need work, so I added a break point. It said this line was throwing Error 18 (problem with compilation – syntax error) as I tried to step through the break point. I added an ending parenthesis to the end of the main function declaration.  
   Moving on, I ran the program as normal. The program produced a result of 987, which does not line up with what Wolfram Alpha gave me: 610. After stepping through, I found out that the Fibonacci function was going an extra time, indicating a problem with the loop. I changed the while statement from (n > 1) to (n > 2) and the program works as it should.
7. I placed a break point at the start of the printGrade function. The function said everything besides the grade being a failure. I tried this after brainwashing Java (ahem, changing the values in the debugger in the variables menu) to make that mark Integer.MAX\_VALUE and got the same result. Numbers lower than 45 will produce “Fail.” I added the word else in front of the ifs on 17, 19, 21, and 23, and I changed 25 to just else.
8. When running the program, it said that there was a Null Pointer Exception at line 6 of the “driver” program (if we are talking in the terms of the style the professor wants us to indicate the main program). The problem showed up after I started to step through the first time. I think I need define an empty constructor like in the reading, but I certainly doubt you define a class variable empty-style with null. Also, I might as well add methods to set the name of the owner (but not balance, deposit works for this). In the empty constructor, I also made sure to force the balance to be initially $0.00 in the blank constructor.  
   Regardless of the balance of the user having less than the withdrawl amount, the withdraw function in Account.java always subtracted the actual balance. I added a break point in the driver program when the amount is depositing and step in, following it, going back to the driver, and then the Account file. The debugger showed me the wrongful negative value as a result. I then added an if-else to prevent the function from withdrawing if there are too low funds. I tested this for proper functioning by withdrawing a passable amount ($50.00 – $2.10) and watching what the debugger said, and my function worked.
9. To start, I put a break point at the public “statuc” line, and the debugger did not run, let alone the entire source code (it was trying to load a class file from the last problem). I changed the spelling and started the rest of the debugging.  
   The next problem also stopped the program in its tracks, because there was a problem in the constructor. The debugger, running from the beginning of the program again, said there was a constructor error that it could not get through, so it crashed. Turns out that, along with some poor choices for variable names, they made the constructor with the name Student, but not the name Person. I changed the names from \_name and \_age to load\_name and load\_age, and changed Student to Person.  
   Moving on, when I started again from the first line of the engine, I saw inside of Person.java that whomever coded this did not remember that they need to use the values defined in the Person file to store an object’s given variables for the existence of the program. I walked through and saw the new variables displaying, but they were not displaying inside of the “this” part. I changed that so it saved to the object’s variables.  
   To top it all off, Fred’s age was not printing. The debugger definitely said the age was saving into the Person object as it was being constructed. The print statement in the main function called for the name. I simply changed the second function call to getAge().