**Homework 1**

Chapter 1: Why do we test?

**Name: Antonio Zea Jr.**

**What to do?**

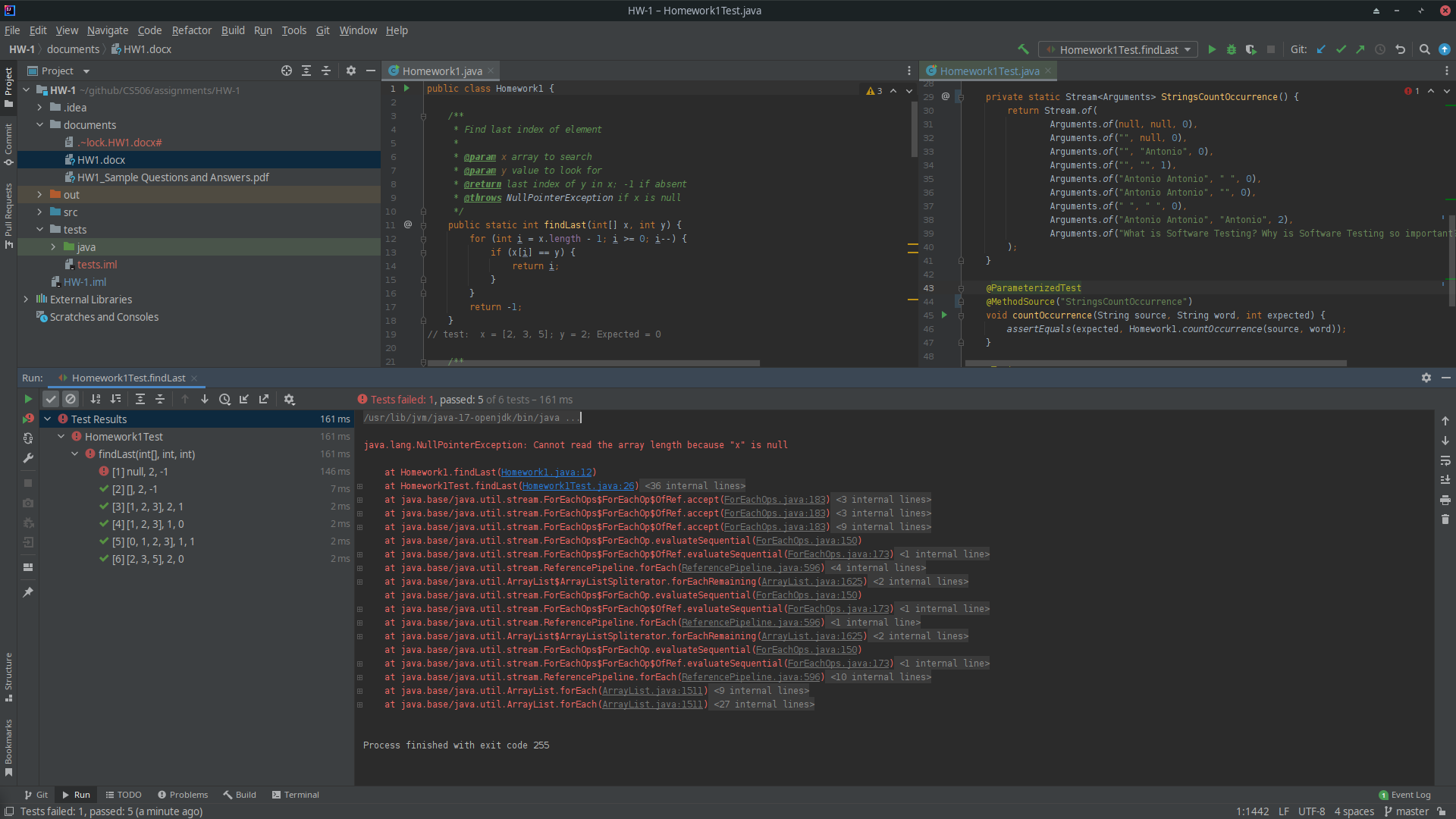
Complete the problems below and submit this word document.

Four faulty programs are given below. Each includes test inputs that result in failure. For each of these four programs, answer the following questions **a — f**. For **“f”** copy and paste your corrected code in this document.

* 1. **findLast method (22 points)**

1. Explain what is wrong with the given code. Describe the fault precisely by proposing a modification to the code.
   1. The array is being searched from the back to front but the front is never reached. In the for loop’s condition, I would change it to “x >=0”.
2. If possible, give a test case that does not execute the fault. If not, briefly explain why not.
   1. The fault is not reached if the array is null.
3. If possible, give a test case that executes the fault, but does not result in an error state. If not, briefly explain why not.
   1. An empty array will reach the fault but not result in an error state. An empty array will cause i=-1 which will evaluate as false in i>0 from the for loop. The same exact state is achieved in the “good” version of the program so no error state is achieved.
4. If possible give a test case that results in an error, but not a failure. If not, briefly explain why not. Hint: Don’t forget about the program counter.
   1. x=[7,15,22],y=3, expected=-1,actual=-1
   2. BAD STATE
   3. <x=[7,15,22],y=3,i=0,x[i]=7,x.length=3,PC=[i>0]>
   4. <x=[7,15,22],y=3,i=0,x[i]=7,x.length=3,PC=[return -1]>
   5. GOOD STATE
   6. <x=[7,15,22],y=3,i=0,x[i]=7,x.length=3,PC=[i>=0]>
   7. <x=[7,15,22],y=3,i=0,x[i]=7,x.length=3,PC=[if(x[i]==y)]>
5. For the given test case below, describe the first error state. Be sure to describe the complete state. Hint: Don’t forget about the program counter.
   1. BAD STATE
   2. <x=[2,3,5],y=2,i=0,x[i]=2,PC=[i>0]>
   3. <x=[2,3,5],y=2,i=0,x[i]=2,PC=[return -1]>
   4. GOOD STATE
   5. <x=[2,3,5],y=2,i=0,x[i]=2,PC=[i>=0]>
   6. <x=[2,3,5],y=2,i=0,x[i]=2,PC=[if(x[i]==y)]>
6. Implement your repair and verify that the given test now produces the expected output. Submit a screenshot demonstrating your new program works.

|  |
| --- |
| */\*\*  \* Find last index of element  \*  \** ***@param x*** *array to search  \** ***@param y*** *value to look for  \** ***@return*** *last index of y in x; -1 if absent  \** ***@throws*** *NullPointerException if x is null  \*/* **private static int** findLast (**int**[] x, **int** y) {  **for** (**int** i = x.**length** - 1; i >= 0; i--)  {  **if** (x[i] == y)  {  **return** i;  }  }  **return** -1; }  *// test: x = [2, 3, 5]; y = 2; Expected = 0* |



* 1. **countOccurrence method (22 points)**

1. Explain what is wrong with the given code. Describe the fault precisely by proposing a modification to the code.
   1. String comparison is attempted using == instead of String.equals() method. Object refrences are compared instead of string contents.
   2. I would change the if statement’s condition from word==str to word.equals(str)
2. If possible, give a test case that does not execute the fault. If not, briefly explain why not.
   1. A null as the source causes a null pointer exception to be thrown. We end up not reaching the fault.
3. If possible, give a test case that executes the fault, but does not result in an error state. If not, briefly explain why not.
   1. If the source and word are both empty strings. The java compiler uses the the refrence for both of these literals and it ends up not producing an error state.
4. If possible, give a test case that results in an error, but not a failure. If not, briefly explain why not. Hint: Don’t forget about the program counter.
   1. Not possible, an error state in this case will result in a fault.
5. For the given test case below, describe the first error state. Be sure to describe the complete state. Hint: Don’t forget about the program counter.
   1. source = "What is Software Testing? Why is Software Testing so important?"; word = "Software”; Expected = 2
   2. str=”Software”, word=”Software”, Reference are compared and found not to be equal.
   3. Count=0;

BAD STATE

<source = "What is ...",word = "Software", temp=[“What”, ”is”,..], count=0, str=”Software”, PC=[if(word==str)]>

< PC=[for(String str: temp)]>

<str=”Testing?”, PC=[if(word==str)]>

GOOD STATE

<source = "What is ...",word = "Software", temp=[“What”, ”is”,..], count=0, str=”Software”, PC=[if(word.equals(str))]>

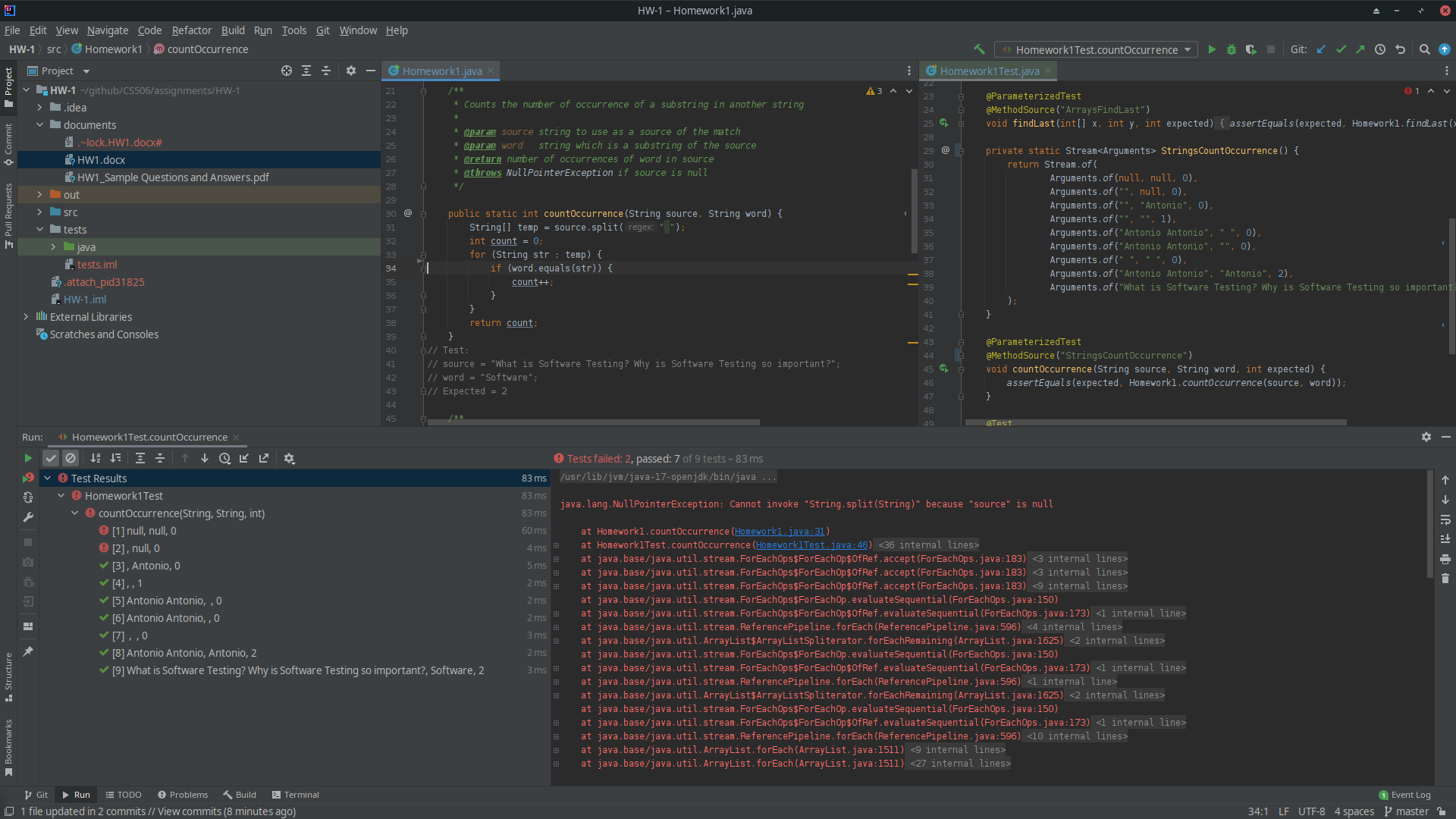
< PC=[count++]>

<count=1, PC=[for(String str: temp)]>

<str=”Testing?”, PC=[if(word.equals(str))]>

1. Implement your repair and verify that the given test now produces the expected output. Submit a screenshot demonstrating your new program works.

|  |
| --- |
| */\*\*  \* Counts the number of occurrence of a substring in another string  \*  \** ***@param source*** *string to use as a source of the match  \** ***@param word*** *string which is a substring of the source  \** ***@return*** *number of occurrences of word in source  \** ***@throws*** *NullPointerException if source is null  \*/* **private static int** countOccurrence(String source, String word) {  String[] temp = source.split(**" "**);  **int** count = 0;  **for** (String str : temp) {  **if** (word.equals(str)){  count++;  }  }  **return** count; }  *// Test:  // source = "**What is Software Testing? Why is Software Testing so important?";  // word = "**Software"; // Expected = 2* |



* 1. **countPositive method (22 points)**

1. Explain what is wrong with the given code. Describe the fault precisely by proposing a modification to the code.
   1. The function is meant to count the number of positive elements in a supplied array. The function counts zero as a positive value.
   2. I would change the if statement’s condition from x[i] >= 0 to x[i] > 0 .
2. If possible, give a test case that does not execute the fault. If not, briefly explain why not.
   1. Null passed as the array and empty array both bypass the fault.
3. If possible, give a test case that executes the fault, but does not result in an error state. If not, briefly explain why not.
   1. x= [-4, -22, 2, -2], count=1, i=2, x[i]=2, x.length=4
   2. This will execute the fault but not result in an error state because there are no zeros in the array.
4. If possible, give a test case that results in an error, but not a failure. If not, briefly explain why not. Hint: Don’t forget about the program counter.
   1. Not possible, if an error state occurs then failure will also occur because zero has been counted as a positive number.
5. For the given test case below, describe the first error state. Be sure to describe the complete state. Hint: Don’t forget about the program counter
   1. x=[-4,2,0,2], count=1, i=2,x[i]=0,x.length=4

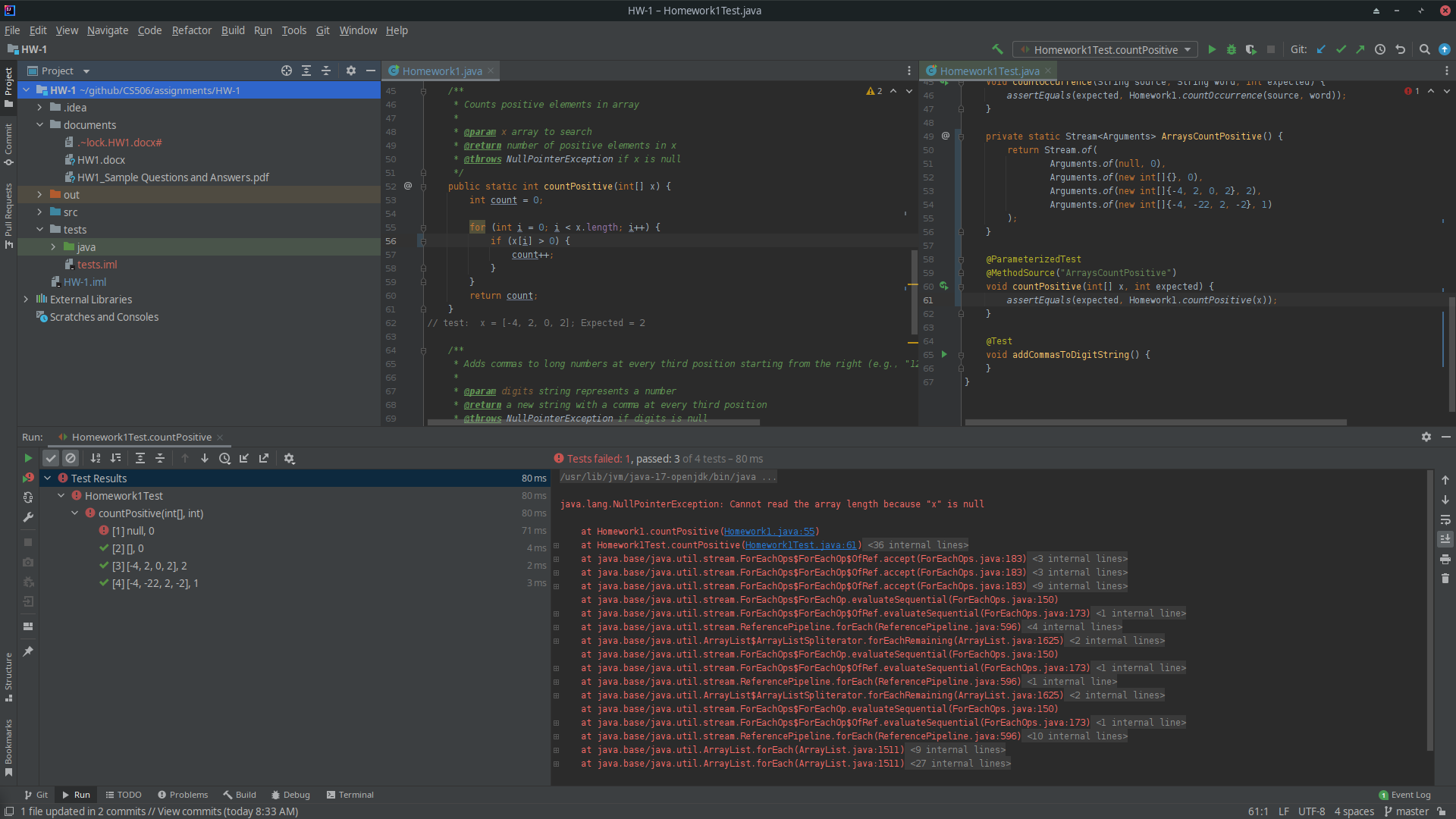
<i=2, x[i]=0, PC=[if(x[i]>=0)]>

<PC=[count++]>

<count=2, PC= [for (int i = 0; i < x.length; i++)]>

1. Implement your repair and verify that the given test now produces the expected output. Submit a screenshot demonstrating your new program works.

|  |
| --- |
| */\*\*  \* Counts positive elements in array  \*  \** ***@param x*** *array to search  \** ***@return*** *number of positive elements in x  \** ***@throws*** *NullPointerException if x is null  \*/* **private static int** countPositive (**int**[] x) {  **int** count = 0;   **for** (**int** i = 0; i < x.**length**; i++)  {  **if** (x[i] > 0)  {  count++;  }  }  **return** count; } *// test: x = [**-4, 2, 0, 2]; Expected = 2* |



* 1. **addCommasToDigitString method (22 points)**

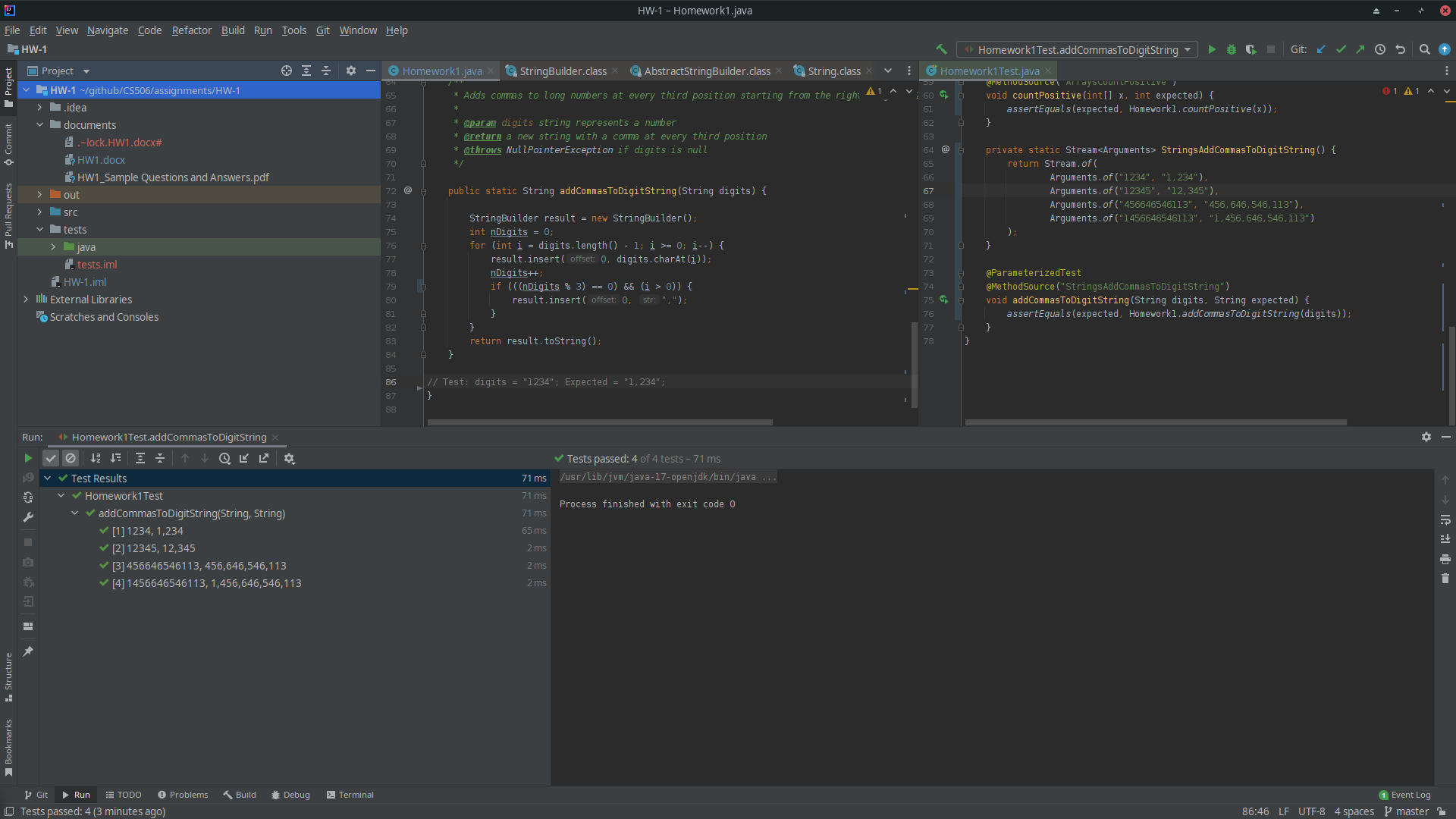
1. Explain what is wrong with the given code. Describe the fault precisely by proposing a modification to the code.
   1. The if statements tries to prevent a comma being added at end of processing, ie “,123”. The issue is that the end of the number occurs at 0 not 1.
   2. I would change if (((nDigits % 3) == 0) && (i > 1)) into if (((nDigits % 3) == 0) && (i > 0)).
2. If possible, give a test case that does not execute the fault. If not, briefly explain why not.
   1. An empty string will not execute the fault.
3. If possible, give a test case that executes the fault, but does not result in an error state. If not, briefly explain why not.
   1. A three digit or less string
4. If possible give a test case that results in an error, but not a failure. If not, briefly explain why not. Hint: Don’t forget about the program counter.
   1. If the length of the string of digits module 3 is 2 or 0 then failure is not reached.
   2. Digits=”12345”, result=”12,345”, expected=”12,345”
5. For the given test case below, describe the first error state. Be sure to describe the complete state. Hint: Don’t forget about the program counter.
   1. Digits=”1234”, result=”234”,nDigits=3,i=1 , Expected=”1,234”
   2. Digits=”1234”, result=”1234”,nDigits=4,i=0 , Expected=”1,234”

<digits=”1234”, result=”234”, nDigits=3, i=1,PC=[if (((nDigits % 3) == 0) && (i > 1))]>

< digits=”1234”, result=”234”, nDigits=3, i=1,PC=[for (int i = digits.length() - 1; i >= 0; i--)]>

1. Implement your repair and verify that the given test now produces the expected output. Submit a screenshot demonstrating your new program works.

|  |
| --- |
| */\*\*  \* Adds commas to long numbers at every third position starting from the right (e.g., "12345678" returns "12,345,678")  \*  \** ***@param digits*** *string represents a number  \** ***@return*** *a new string with a comma at every third position  \** ***@throws*** *NullPointerException if digits is null  \*/* **private static** String addCommasToDigitString(String digits) {   StringBuilder result = **new** StringBuilder();  **int** nDigits = 0;  **for** (**int** i = digits.length()-1; i >= 0; i--) {  result.insert(0, digits.charAt(i));  nDigits++;  **if** (((nDigits % 3) == 0) && (i > 0)) {  result.insert(0, **","**);  }  }  **return** result.toString(); }  *// Test: digits = "1234"; Expected = "1,234";* |



1. I'm looking for a paragraph to a page here. This is an introspective exercise without a "correct" answer, but with justification. The idea is for you to place yourself (and possibly your company) into the context described by Beizer's classification of levels of testing. You'll receive credit if you turn in something reasonable. If you have worked at a company that did software development, what level of test maturity do you think the company worked at? (0: testing=debugging, 1: testing shows correctness, 2: testing shows the program doesn’t work, 3: testing reduces risk, 4: testing is a mental discipline about quality) If you have not worked at in this scenario, think about a large project you worked on as a team and answer what level of test maturity do you think that project worked at? **(12 points)**

There are two projects that come to mind as I have yet to work as developer in the professional sense. In one project I incorporated Junit testing into the classes that we had constructed so that we could always tell if modifications to those classes broke the functionality that we had already achieved. I think that would have put that project somewhere between level 2 and level 3. The other developers did not put the same effort into testing as they never used those tests in their development process. They hardly debugged their code at all which directly explained how much time we have to spend recoding the new features that were added but broke the program in some way. On average, I would say our team mostly saw the goal of testing as debugging.

In a second project, although we did not use any testing suite to manage our tests, we each thoroughly tested our code independently. The view we took on our testing varied from debugging, showing correctness to testing whether there were ways to break program. I do not believe we had the mindset of reducing risk through testing. But I do believe that our goal with testing was to show that our programs produced correct results.

**Due Date**

This homework is due by **Sunday, February 6, 2022, 11:59 pm.** A penalty of 10% per day will be deducted from your grade, starting at 12:00:01 am.

**What to submit?**

Submit the following file to Blackboard:

* A word document describing your answers to the questions above.