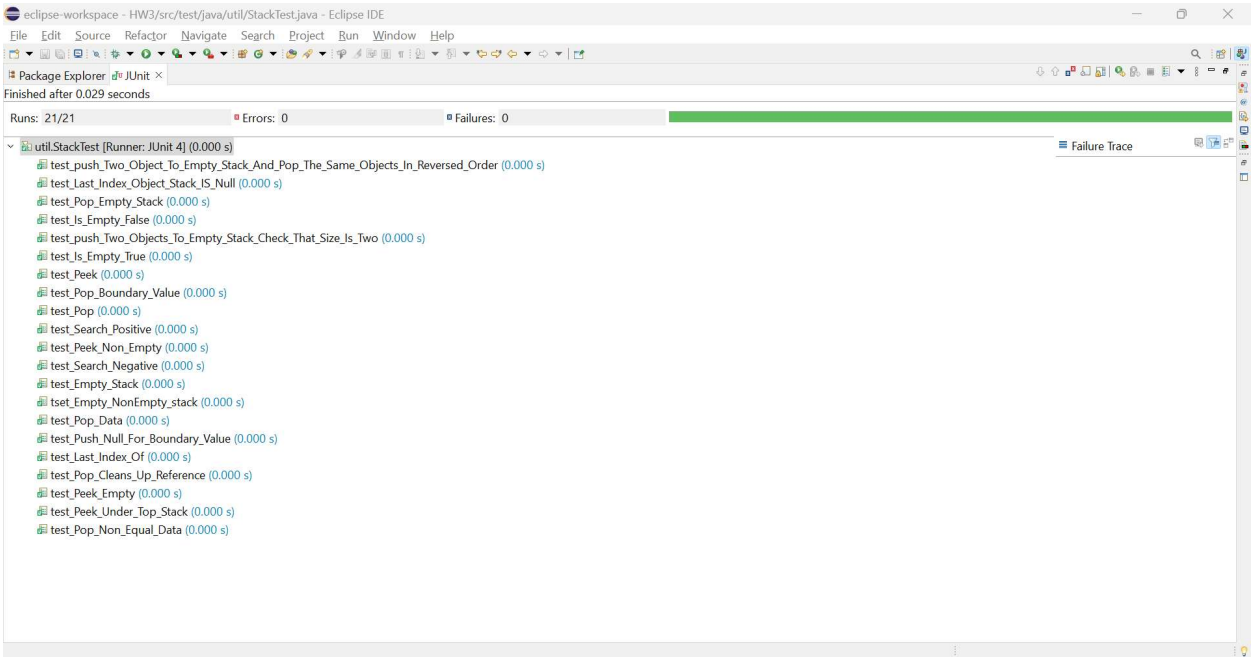
Results of Branch Coverage & Test Cases that are Implemented

1) **test_Is_Empty_True() :** Using Equivalence Partioning, the isEmpty function of the class Stack is tested.
2) **test_Is_Empty_False() :** Using Equivalence Partitioning, the isEmpty function test for the class Stack.
3) **test_Push_Null_For_Boundary_Value()** : Using boundary values, the isEmpty function of the class Stack is tested.
4) **test_push_Two_Objects_To_Empty_Stack_Check_That_Size_Is_Two ():** Test of Pushing Earlier Created Objects Using Boundary Value into Empty Stack.
5) **test_push_Two_Object_To_Empty_Stack_And_Pop_The_Same_Objects_In_Reversed_Order()**: Test of Pushing Earlier Created Objects Using Boundary Value into Empty Stack.
6) **test_Pop() :** Stack class's pop function is tested using boundary values.
7) **test_Pop_Data ()** : As classify the software's incoming data into various equivalence data classes.
8) **test_Pop_Non_Equal_Data()** : Check for pop-class Stack non-equal values.
9) **test_Pop_Boundary_Value()** : boundary value analysis
10) **test_Pop_Empty_Stack () :** Test of the pop function of the class Stack's exception handling.
11) **test_Pop_Cleans_Up_Reference() :** Test of the class Stack's Pop Cleaning of Reference with Null function.
12) **test_Peek() :** Test of the class Stack's Peek function using Boundary Value
13) **test_Peek_Empty()** : separate the software's input data into various equivalence data types.
14) **test_Peek_Non_Empty()** : We categorize the software's incoming data into various equivalence data classes.
15) **test_Peek_Under_Top_Stack()** : Test of the class Stack's peek-under-top-stack function using equivalence.
16) **tset_Empty_NonEmpty_stack() :** We categorize the software's incoming data into various equivalence data classes.
17) **test_Empty_Stack() :** We categorize the software's incoming data into various equivalence data classes.
18) **test_Search_Positive()** : We categorize the software's incoming data into various equivalence data classes.
19) **test_Search_Negative() :** We categorize the software's incoming data into various equivalence data classes.
20) **test_Last_Index_Of() :** Equivalence is used to test the Last Index Object with Null function of the class Stack.
21) **test_Last_Index_Object_Stack_IS_Null()** : Test of the class Stack's Last Index Object with Null function using a boundary value

Junit Result:

Total 21/21 Executed in Junit and result is Green.

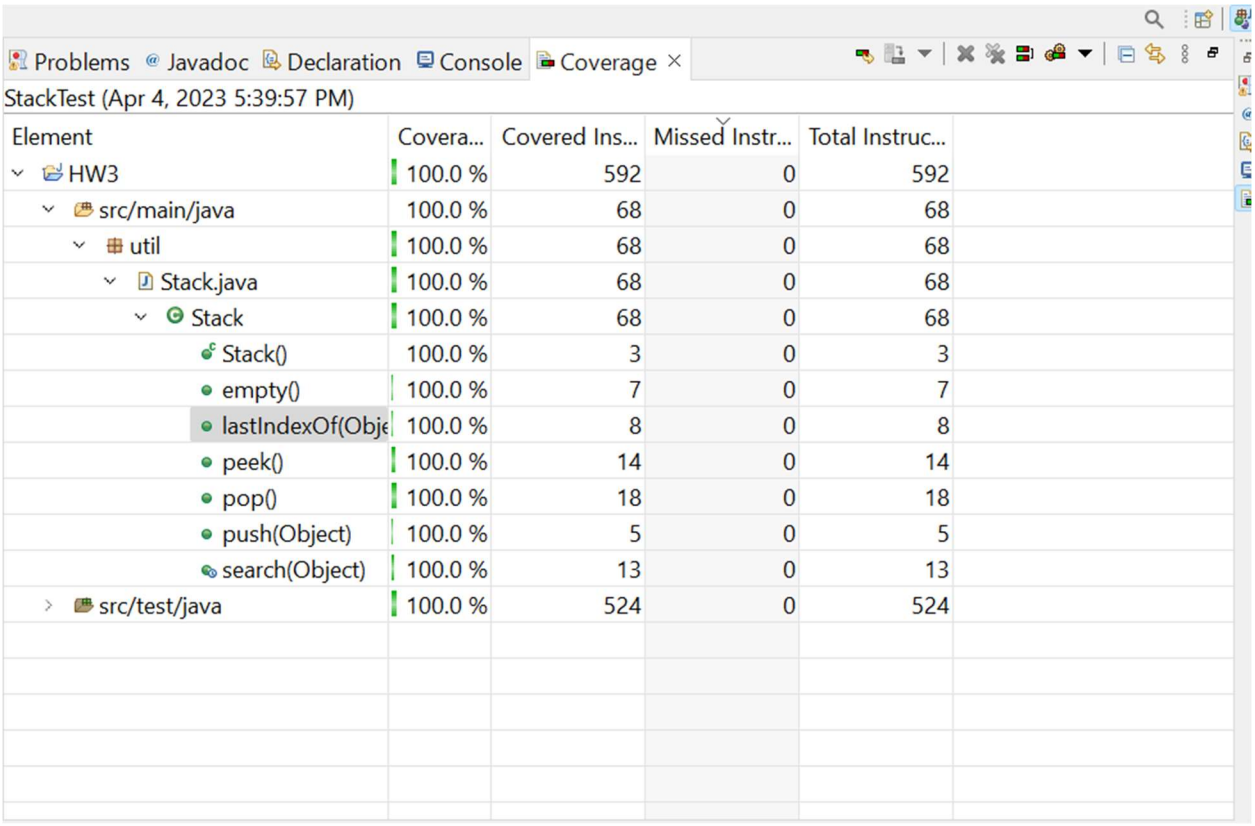
Below is the proof of it.



Branch Coverage:

The Solution accomplished 100% branch coverage, Below are screenshots for **Stack.java** & **StackTest.java**

1) **Stack.java** achieved 100% coverage:



Element	Covera...	Covered Ins...	Missed Instr...	Total Instruc...
HW3	100.0 %	592	0	592
src/main/java	100.0 %	68	0	68
util	100.0 %	68	0	68
Stack.java	100.0 %	68	0	68
Stack	100.0 %	68	0	68
Stack()	100.0 %	3	0	3
empty()	100.0 %	7	0	7
lastIndexOf(Obje	100.0 %	8	0	8
peek()	100.0 %	14	0	14
pop()	100.0 %	18	0	18
push(Object)	100.0 %	5	0	5
search(Object)	100.0 %	13	0	13
src/test/java	100.0 %	524	0	524

2) <u>StackTest.java</u> <u>Achieved 100% Branch Coverage:</u>

StackTest (Apr 4, 2023 5:39:57 PM)

Element	Covera...	Covered Ins...	Missed Instr...	Total Instruc...
src/test/java	100.0 %	524	0	524
util	100.0 %	524	0	524
StackTest.java	100.0 %	524	0	524
StackTest	100.0 %	524	0	524
empty()	100.0 %	25	0	25
lastIndexOf()	100.0 %	42	0	42
peek()	100.0 %	18	0	18
pushObjectsInO	100.0 %	22	0	22
pushTwoObjects	100.0 %	19	0	19
pushTwoObjectT	100.0 %	23	0	23
setUp()	100.0 %	6	0	6
testEmpty()	100.0 %	8	0	8
testIsEmptyFalse	100.0 %	14	0	14
testIsEmptyTrue(	100.0 %	8	0	8
testLastIndexOb	100.0 %	38	0	38
testPeekEmpty()	100.0 %	8	0	8
testPeekNonEmp	100.0 %	25	0	25
testPeekUnderTo	100.0 %	34	0	34
testPop()	100.0 %	16	0	16
testPopBoundar	100.0 %	44	0	44
testPopCleansUp	100.0 %	28	0	28
testPopData()	100.0 %	20	0	20
testPopEmptySt	100.0 %	8	0	8
testPopNonEqua	100.0 %	24	0	24
testPushNonEmp	100.0 %	13	0	13
testPushNullForE	100.0 %	19	0	19
testSearchNegat	100.0 %	21	0	21
testSearchPositiv	100.0 %	21	0	21