Postal Rate Calculator Application Development

Name: Zeyu Chen 260736792

Name: Chen He 260743776

The application is a command line program written in C#.

The application takes no arguments but read inputs from stdin.

$ from to length width height weight type

Where

from is the postcode where you send the package (length of six, with the format of Character|number| Character|number| Character|number)

to is the postcode of the destination (length of six, with the format of Character|number| Character|number| Character|number)

length is the package length in centimeters (cm)

width is the package width in centimeters (cm)

height is the package height in centimeters (cm)

weight is the package weight in kilograms (kg)

type is the type of post (Regular, Xpress, Priority)

The rules for postal rate calculation are as following:

A standard package has a minimum length of 14 cm and a maximum length of 24.5 cm. A standard package has a minimum width of 9 cm and a maximum width of 15.6 cm. A standard package has a minimum height of 15 cm and a maximum height of 35 cm. A standard package has a minimum weight of 0.3 kg. A standard package has a maximum weight of 3 kg.

Non-standard packages cannot exceed a length of 38 cm. Non-standard packages cannot exceed a width of 27 cm. Non-standard packages cannot exceed a width of 50 cm. Non-standard packages cannot exceed a weight of 5 kg.

In regular type the postal rate is as following:
The postal rate for standard packages up to 3 kg is $0.49.
The postal rate for standard packages over 3 kg up to 5 kg is $0.80.
The postal rate for non-standard packages up to 1 kg is $0.98.
The postal rate for non-standard packages over 1 kg up to 5 kg is $2.4.

In Xpress type the postal rate is 1.5 times the postal rate in regular type in each situation.
In Priority type the postal rate is 2 times the postal rate in regular type in each situation.

For calculation between two postcode:

Since the post rate is based on postal area and we do not know the exact formula that is used by CanadaPost, we assume that the postal rate is computed by taking the difference of tha first letter of the postal code (which indicate the region) starting at 9$. The final price would be the product of (1+posttype/2) and the sum of the package cost and the postal rate.

Steps:

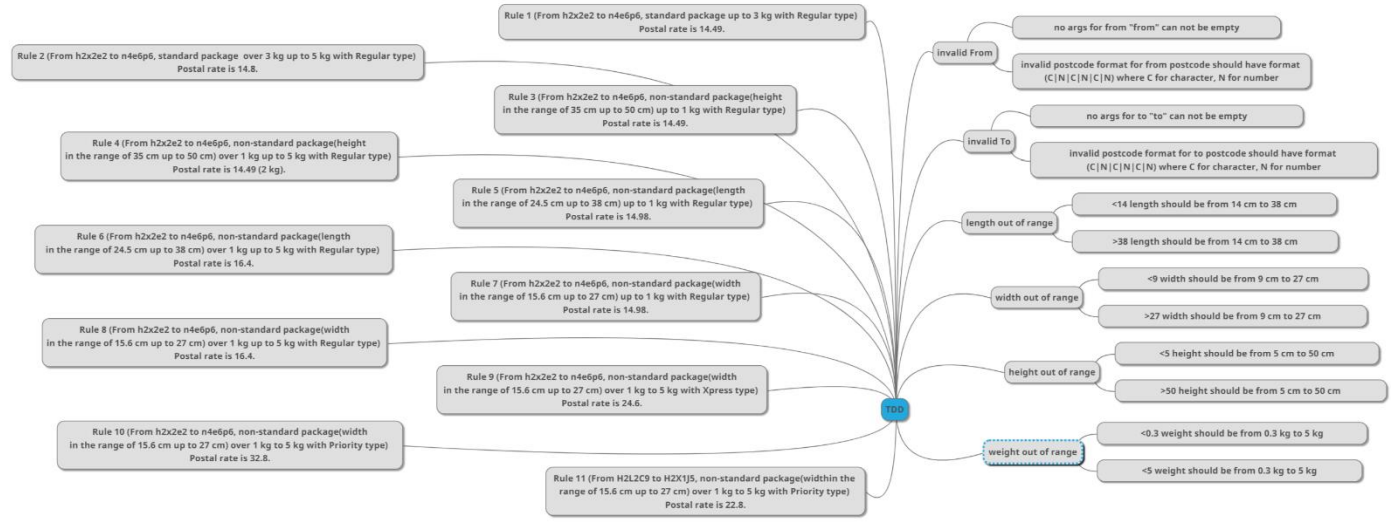
1. Write all unit-tests and declare not implemented methods
2. Write the documentation for unit-tests, then screenshot failed tests
3. Implement the minimum to pass all tests then screenshot.

Command line program:

Calculates the postal rate of the package with from, to, length, width, height, weight, type as input value
Implemented with Test Driven Development.

Unit Test Mind Map

A mind map was created based on reviewing the requirements to guide development of the unit tests required to implement the postal rate calculator.



Initialization

Initialize the data lookup table.

Valid package: from:h2l2c9, to:h2x1j5, length:15, width:10, height:30, weight:2, PostType:Regular

Test 1

1. Test name: no_args for from

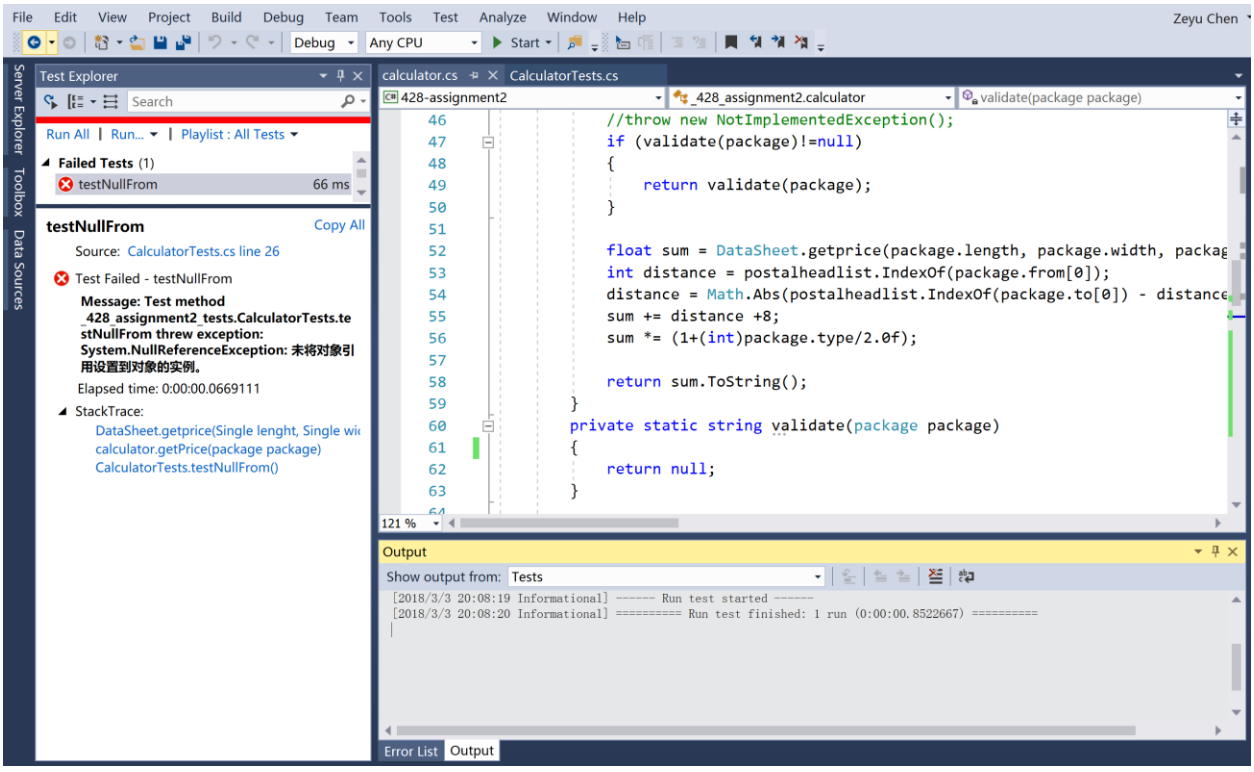
2. Call Setup: create a new package from valid package.

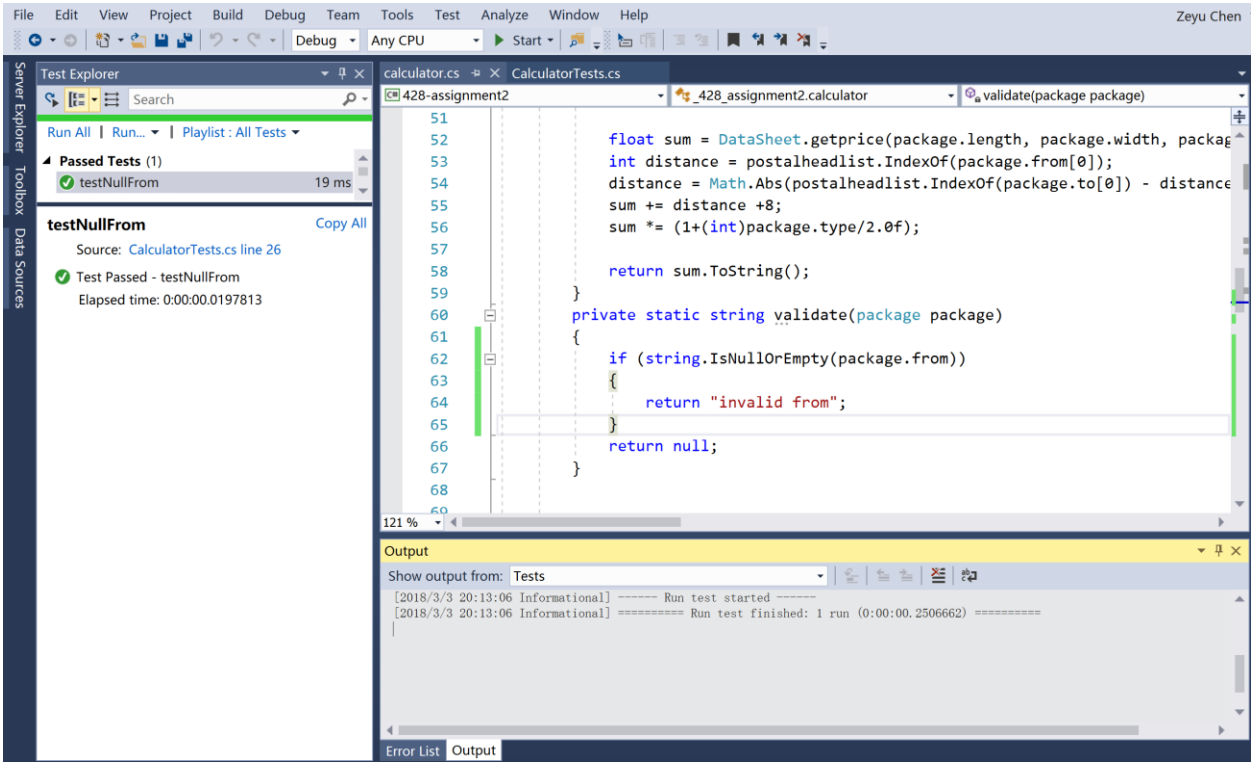
Let package.from equal null.

Calculator.getPrice (package).

3. Expected Result: "invalid from"

4. This test checks if the "to" section is empty and prevent null pointer exceptions


File Edit View Project Build Debug Team Tools Test Analyze Window Help
Zeyu Chen
Debug
Any CPU
Start
Test Explorer
Search
Run All | Run... | Playlist : All Tests
Failed Tests (1)
testNullFrom 66 ms
testNullFrom
Copy All
Source: CalculatorTests.cs line 26
Test Failed - testNullFrom
Message: Test method _428_assignment2_tests.CalculatorTests.testNullFrom threw exception: System.NullReferenceException: 未将对象引用设置到对象的实例。
Elapsed time: 0:00:00.0669111
StackTrace:
DataSheet.getprice(Single lenght, Single wi
calculator.getPrice(package package)
CalculatorTests.testNullFrom()
calculator.cs
CalculatorTests.cs
428-assignment2
_428_assignment2.calculator
validate(package package)
46 //throw new NotImplementedException();
47 if (validate(package)!=null)
48 {
49 return validate(package);
50 }
51
52 float sum = DataSheet.getprice(package.length, package.width, packag
53 int distance = postalheadlist.IndexOf(package.from[0]);
54 distance = Math.Abs(postalheadlist.IndexOf(package.to[0]) - distance
55 sum += distance +8;
56 sum *= (1+(int)package.type/2.0f);
57
58 return sum.ToString();
59 }
60 private static string validate(package package)
61 {
62 return null;
63 }
121 %
Output
Show output from: Tests
[2018/3/3 20:08:19 Informational] ------ Run test started ------
[2018/3/3 20:08:20 Informational] ========== Run test finished: 1 run (0:00:00.8522667) ==========
Error List
Output


File Edit View Project Build Debug Team Tools Test Analyze Window Help
Zeyu Chen
Debug
Any CPU
Start
Test Explorer
Search
Run All | Run... | Playlist : All Tests
Passed Tests (1)
testNullFrom 19 ms
testNullFrom
Copy All
Source: CalculatorTests.cs line 26
Test Passed - testNullFrom
Elapsed time: 0:00:00.0197813
calculator.cs
CalculatorTests.cs
428-assignment2
_428_assignment2.calculator
validate(package package)
51
52 float sum = DataSheet.getprice(package.length, package.width, packag
53 int distance = postalheadlist.IndexOf(package.from[0]);
54 distance = Math.Abs(postalheadlist.IndexOf(package.to[0]) - distance
55 sum += distance +8;
56 sum *= (1+(int)package.type/2.0f);
57
58 return sum.ToString();
59 }
60 private static string validate(package package)
61 {
62 if (string.IsNullOrEmpty(package.from))
63 {
64 return "invalid from";
65 }
66 return null;
67 }
68
121 %
Output
Show output from: Tests
[2018/3/3 20:13:06 Informational] ------ Run test started ------
[2018/3/3 20:13:06 Informational] ========== Run test finished: 1 run (0:00:00.2506662) ==========
Error List
Output

Test 2

1. Test name: no_args for to

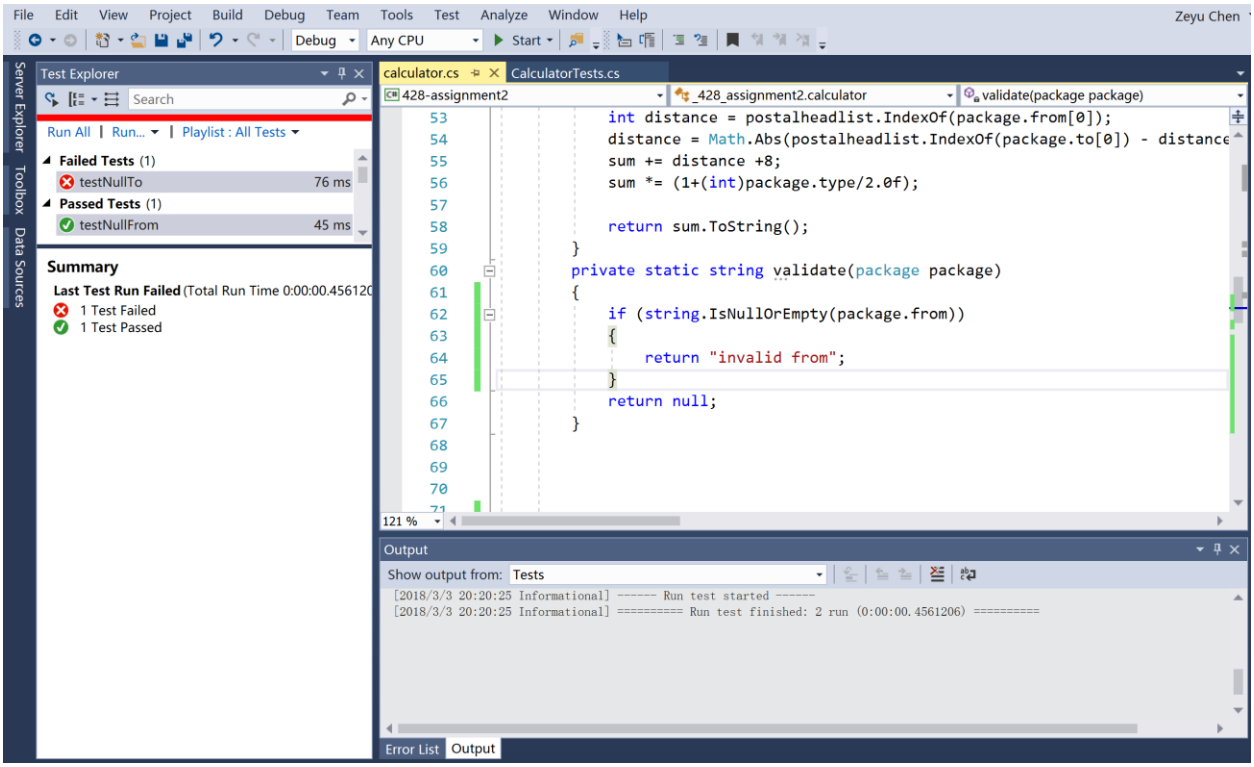
2. Call Setup: create a new package from valid package.

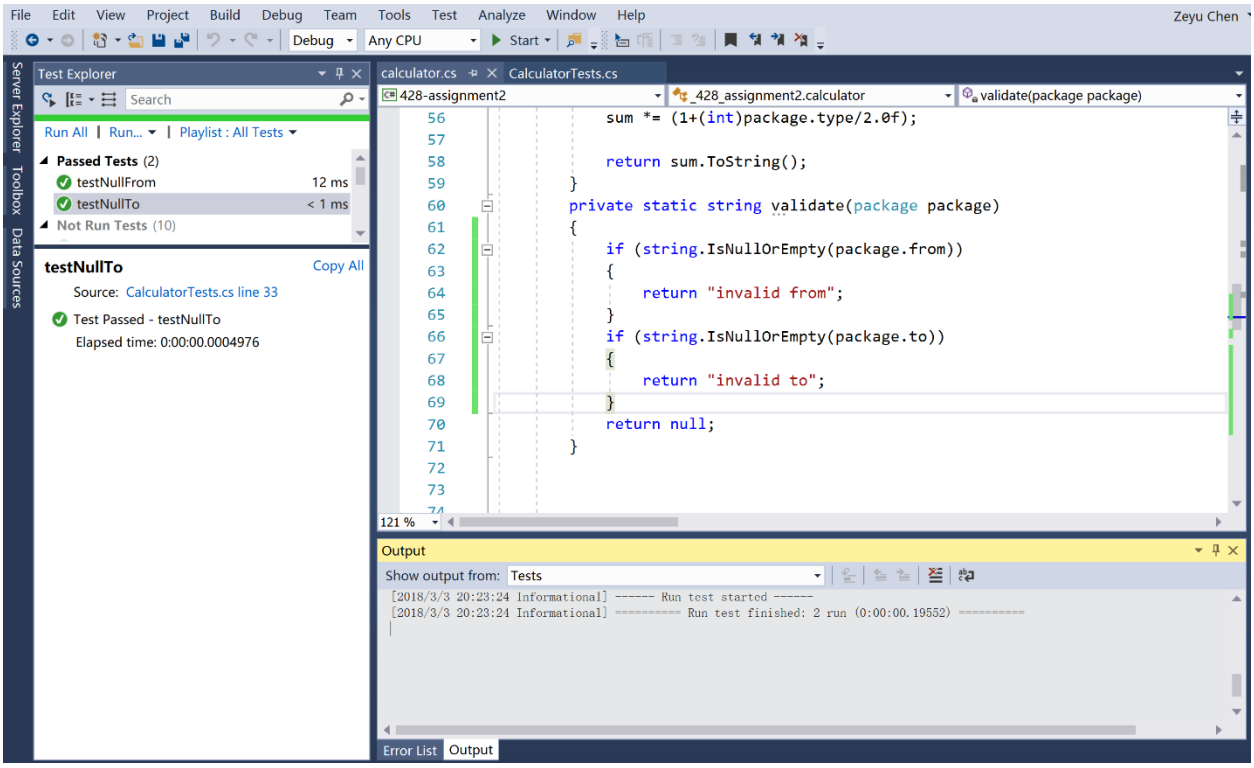
Let package.to equal null.

Calculator.getPrice (package).

3. Expected Result: "invalid to"

4. This test checks if the "to" section is empty and prevent null pointer exceptions





Test3

1. Test name: out_of _range_length_low

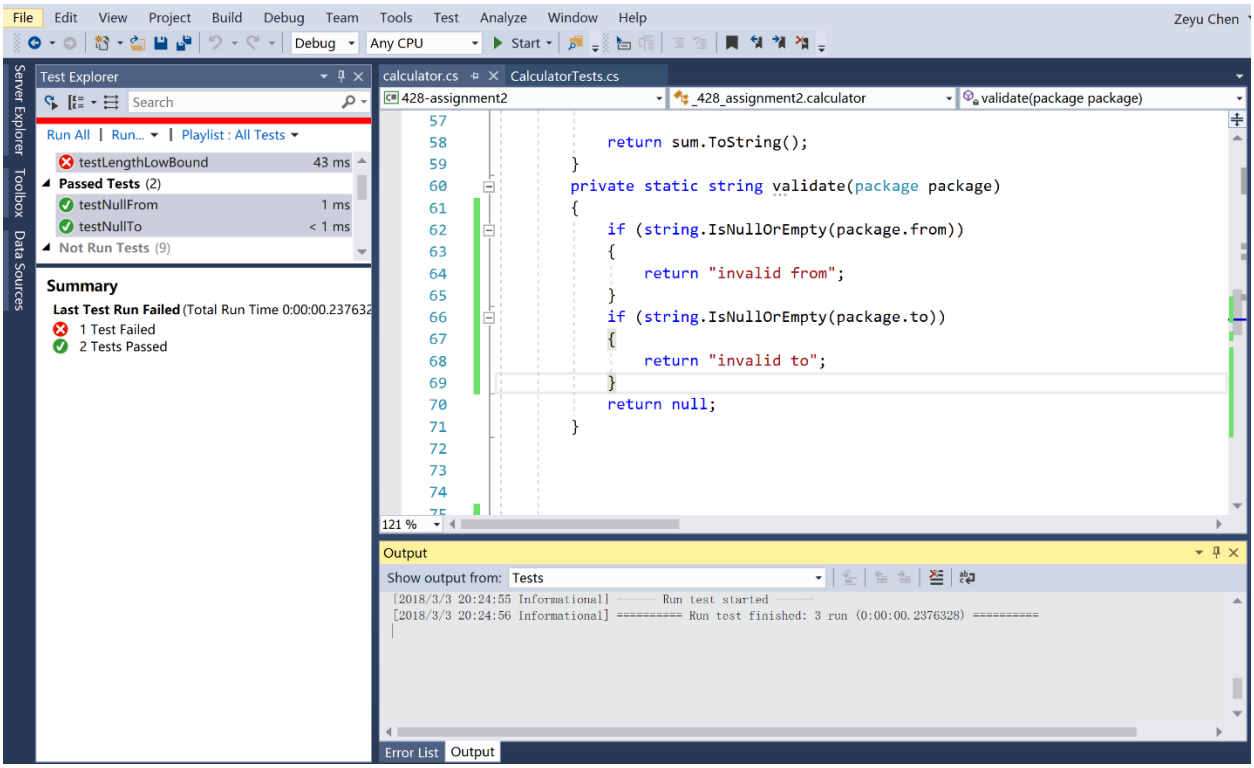
2. Call Setup: create a new package from valid package.

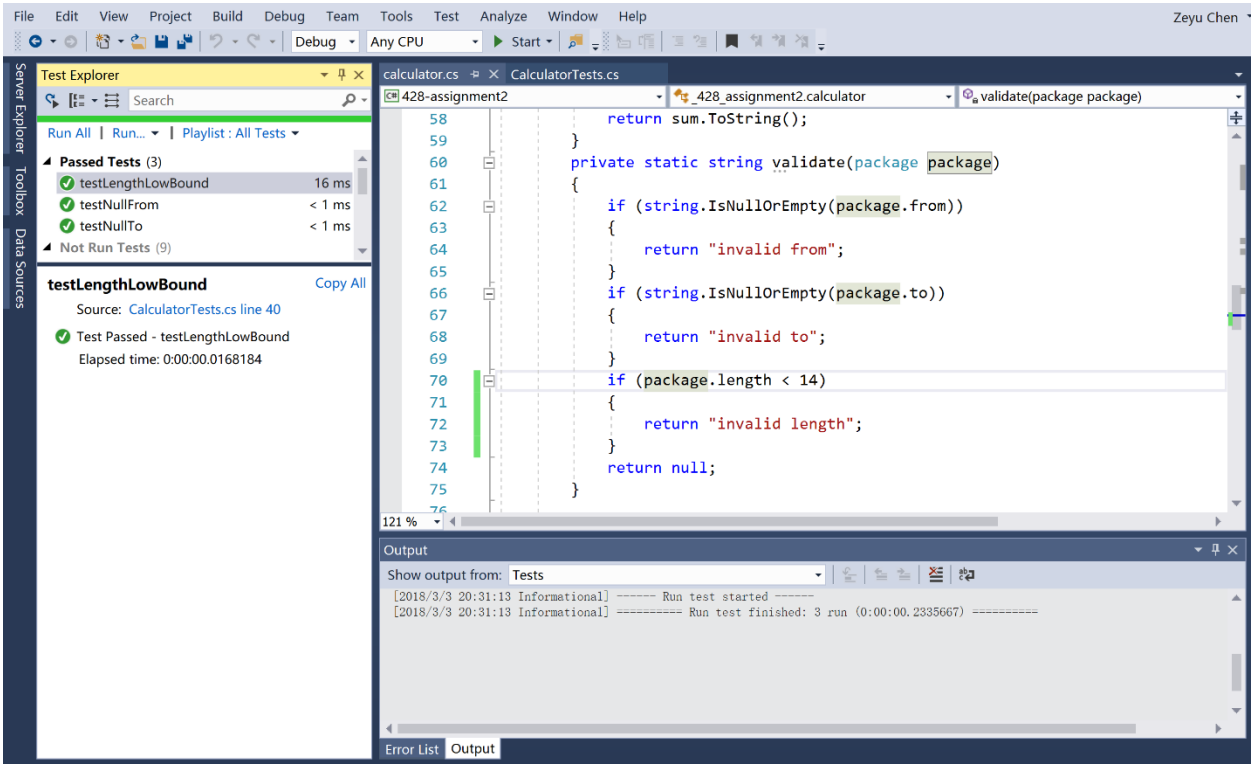
Let package.length=13.9.

Calculator.getPrice (package).

3. Expected Result: "invalid length"

4. It was decided that the range of valid length would be extended down to 14cm.





Test4

1. Test name: out_of _range_length_high

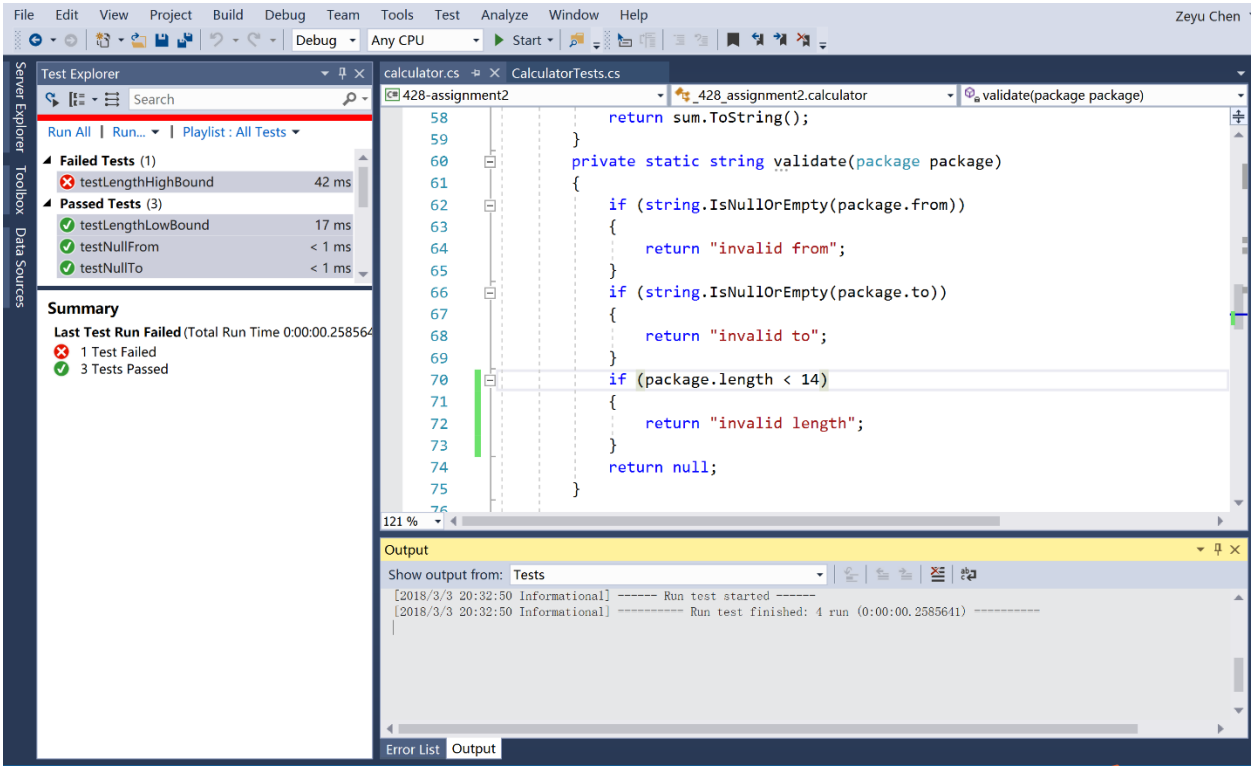
2. Call Setup: create a new package from valid package.

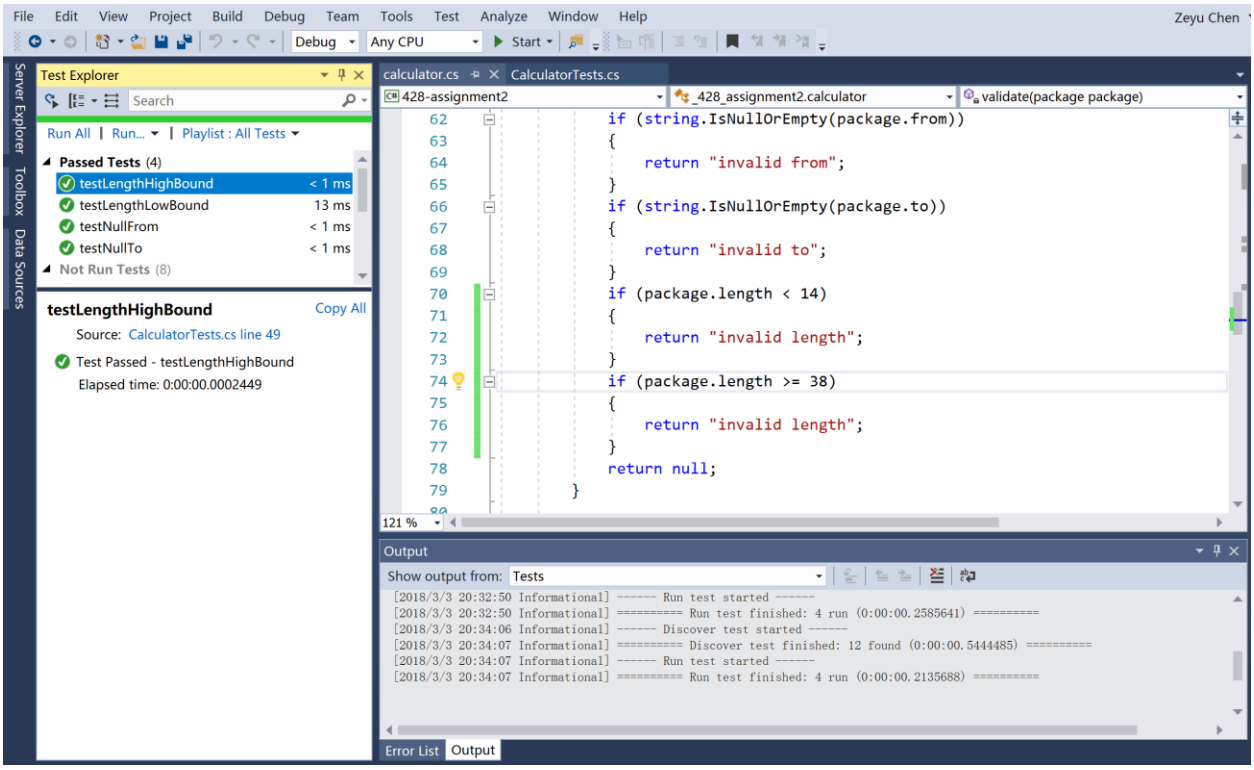
Let package.length=38.1.

Calculator.getPrice (package).

3. Expected Result: "invalid length"

4. It was decided that the range of valid length would be extended up to 38cm.





Test5

1. Test name: out_of _range_width_low

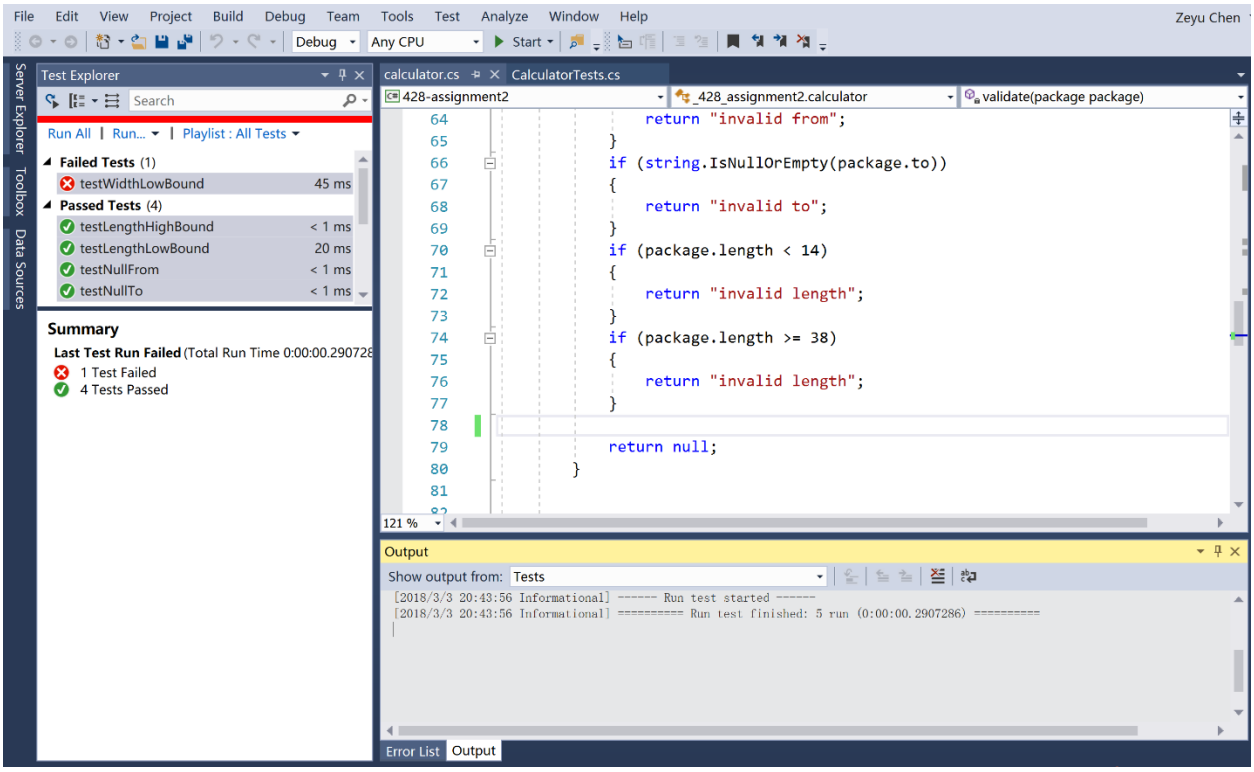
2. Call Setup: create a new package from valid package.

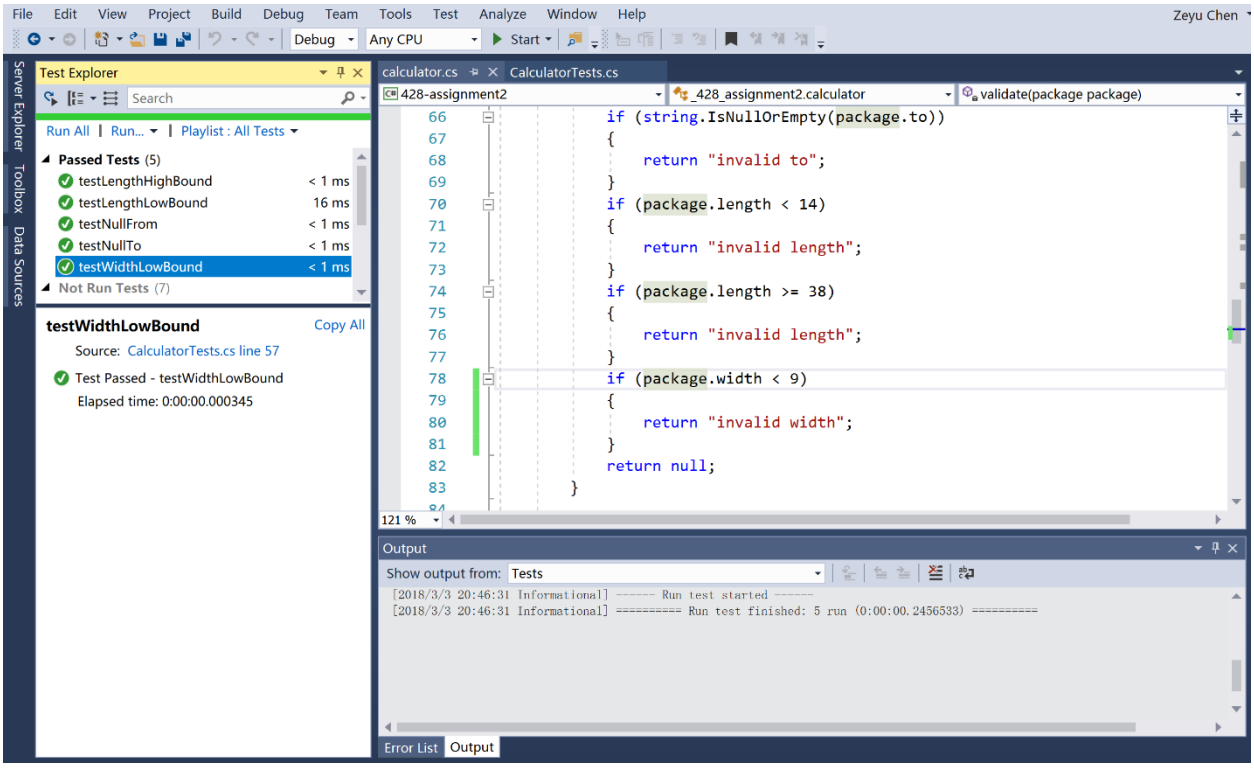
Let package.width=8.9.

Calculator.getPrice (package).

3. Expected Result: "invalid width"

4. It was decided that the range of valid width would be extended down to 9cm.


File Edit View Project Build Debug Team Tools Test Analyze Window Help
Zeyu Chen
Debug
Any CPU
Start
Test Explorer
Search
Run All | Run... | Playlist : All Tests
Failed Tests (1)
testWidthLowBound 45 ms
Passed Tests (4)
testLengthHighBound < 1 ms
testLengthLowBound 20 ms
testNullFrom < 1 ms
testNullTo < 1 ms
Summary
Last Test Run Failed (Total Run Time 0:00:00.29072
1 Test Failed
4 Tests Passed
calculator.cs
CalculatorTests.cs
428-assignment2
_428_assignment2.calculator
validate(package package)
64 return "invalid from";
65 }
66 if (string.IsNullOrEmpty(package.to))
67 {
68 return "invalid to";
69 }
70 if (package.length < 14)
71 {
72 return "invalid length";
73 }
74 if (package.length >= 38)
75 {
76 return "invalid length";
77 }
78
79 return null;
80 }
81
121 %
Output
Show output from: Tests
[2018/3/3 20:43:56 Informational] ------ Run test started ------
[2018/3/3 20:43:56 Informational] ========== Run test finished: 5 run (0:00:00.2907286) ==========
Error List
Output


File Edit View Project Build Debug Team Tools Test Analyze Window Help
Zeyu Chen
Debug
Any CPU
Start
Test Explorer
Search
Run All | Run... | Playlist : All Tests
Passed Tests (5)
testLengthHighBound < 1 ms
testLengthLowBound 16 ms
testNullFrom < 1 ms
testNullTo < 1 ms
testWidthLowBound < 1 ms
Not Run Tests (7)
testWidthLowBound
Copy All
Source: CalculatorTests.cs line 57
Test Passed - testWidthLowBound
Elapsed time: 0:00:00.000345
calculator.cs
CalculatorTests.cs
428-assignment2
_428_assignment2.calculator
validate(package package)
66 if (string.IsNullOrEmpty(package.to))
67 {
68 return "invalid to";
69 }
70 if (package.length < 14)
71 {
72 return "invalid length";
73 }
74 if (package.length >= 38)
75 {
76 return "invalid length";
77 }
78 if (package.width < 9)
79 {
80 return "invalid width";
81 }
82 return null;
83 }
121 %
Output
Show output from: Tests
[2018/3/3 20:46:31 Informational] ------ Run test started ------
[2018/3/3 20:46:31 Informational] ========== Run test finished: 5 run (0:00:00.2456533) ==========
Error List
Output

Test6

1. Test name: out_of _range_width_high

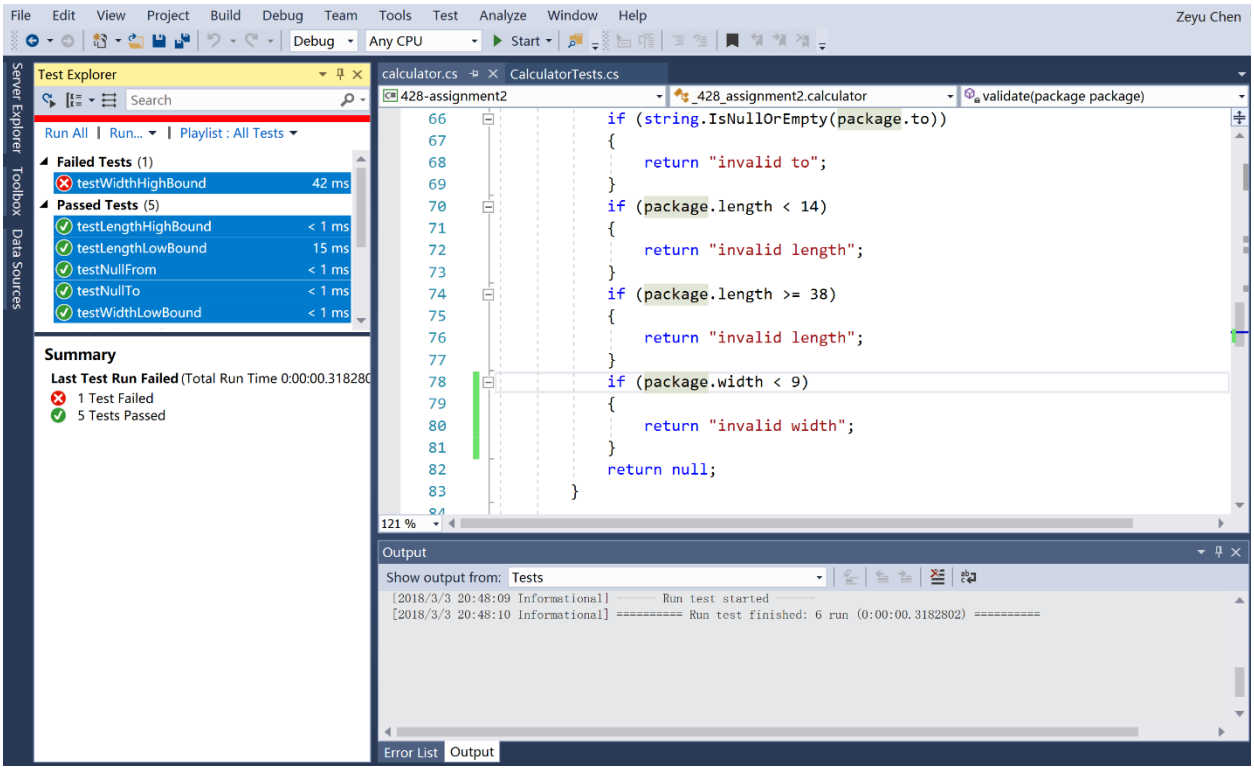
2. Call Setup: create a new package from valid package.

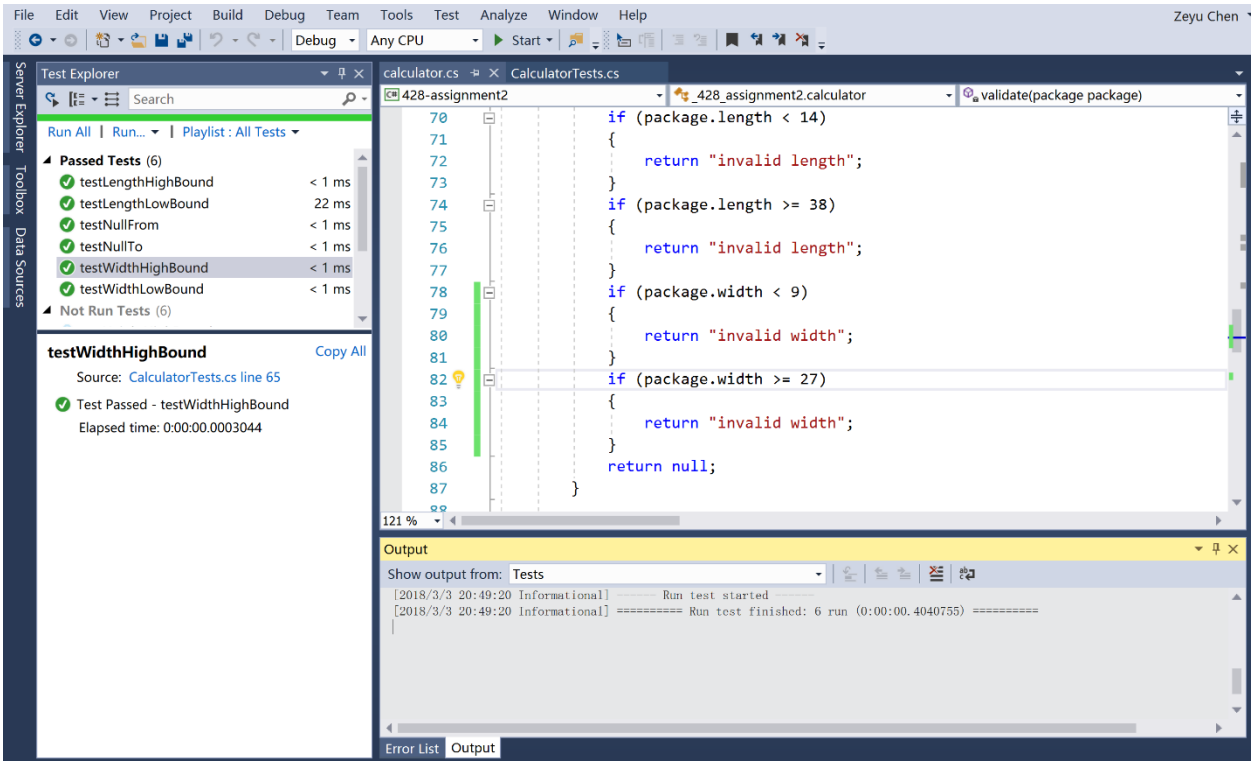
Let package.width=27.1.

Calculator.getPrice (package).

3. Expected Result: Usage: "invalid width"

4. It was decided that the range of valid width would be extended up to 27cm.





Test7

1. Test name: out_of _range_height_low

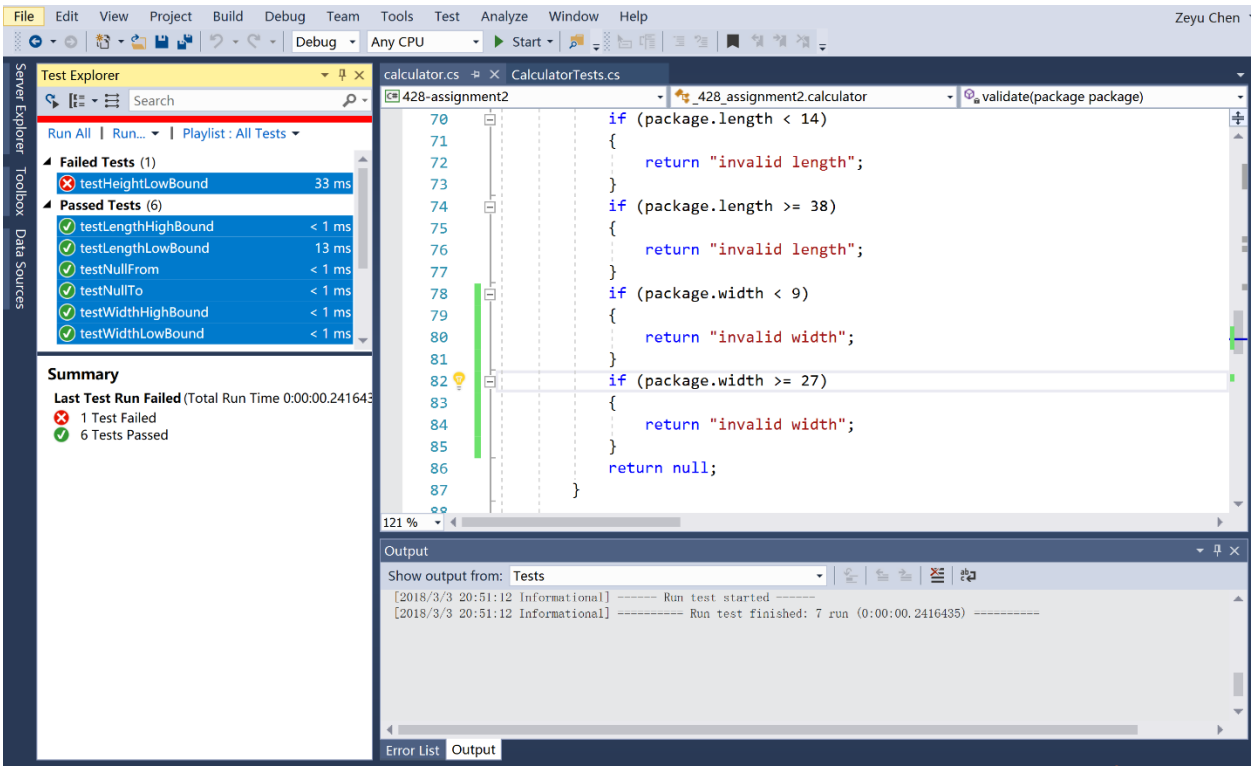
2. Call Setup: create a new package from valid package.

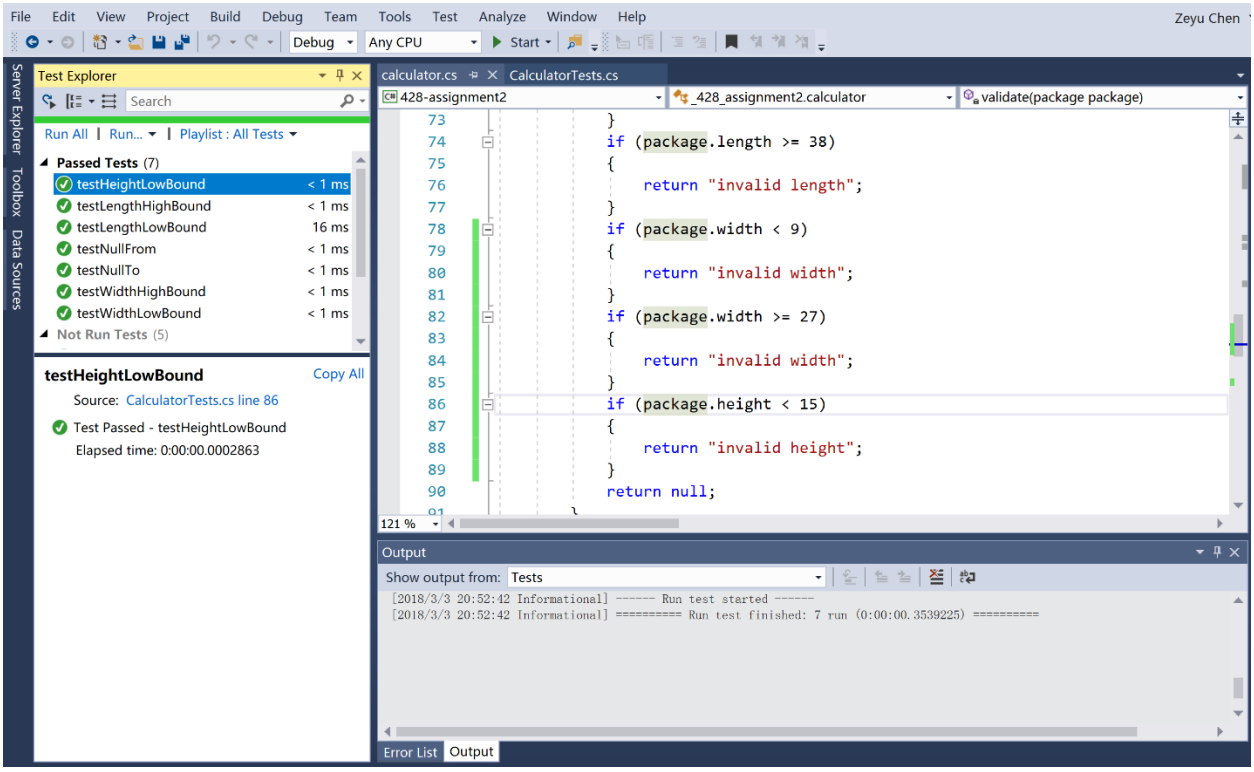
Let package.height=4.9.

Calculator.getPrice (package).

3. Expected Result: "invalid height"

4. It was decided that the range of valid height would be extended down to 5cm.





Test8

1. Test name: out_of _range_height_high

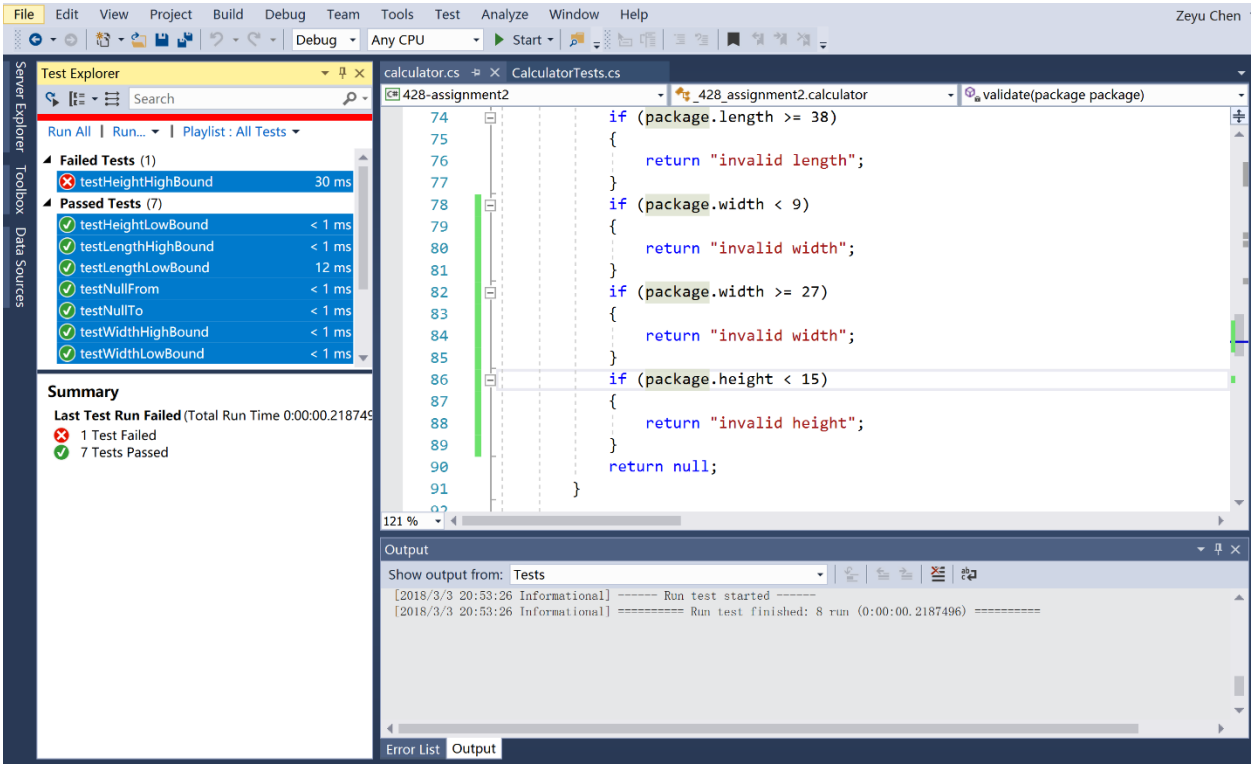
2. Call Setup: create a new package from valid package.

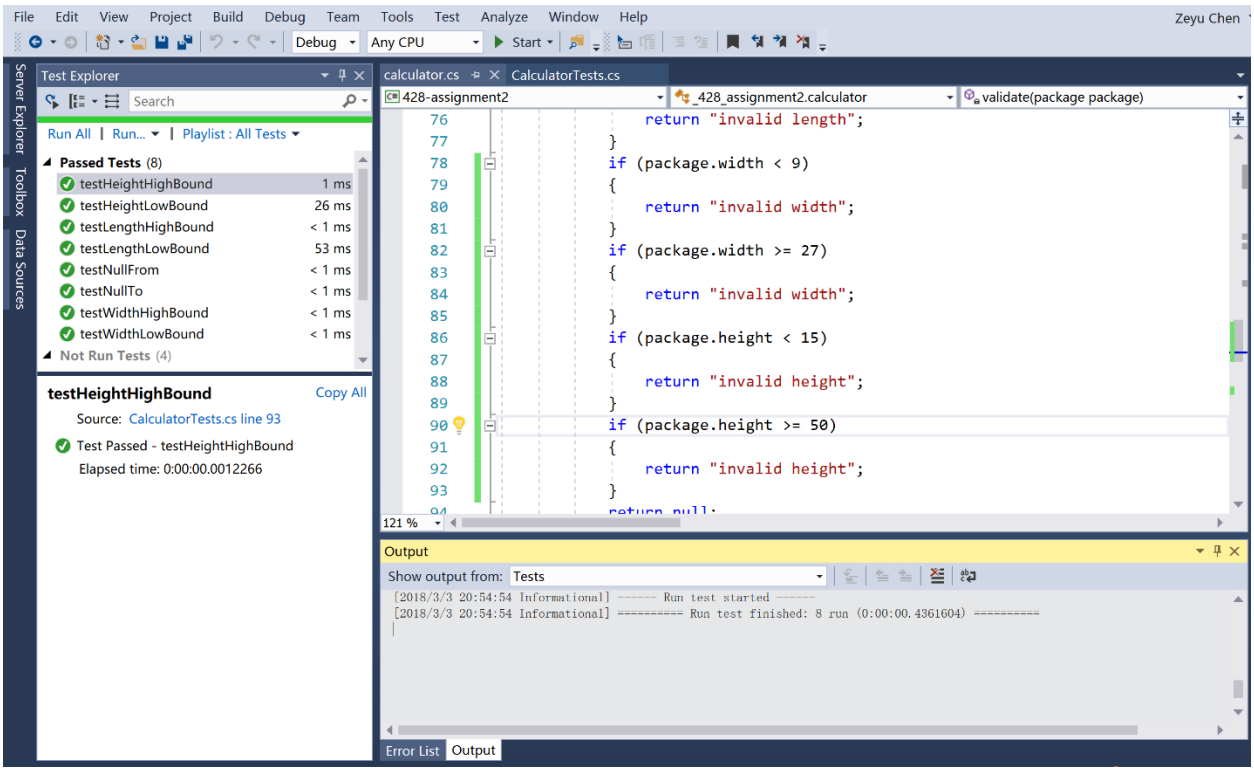
Let package.height=50.1.

Calculator.getPrice (package).

3. Expected Result: "invalid height"

4. It was decided that the range of valid height would be extended up to 50cm.





Test9

1. Test name: out_of _range_weight_low

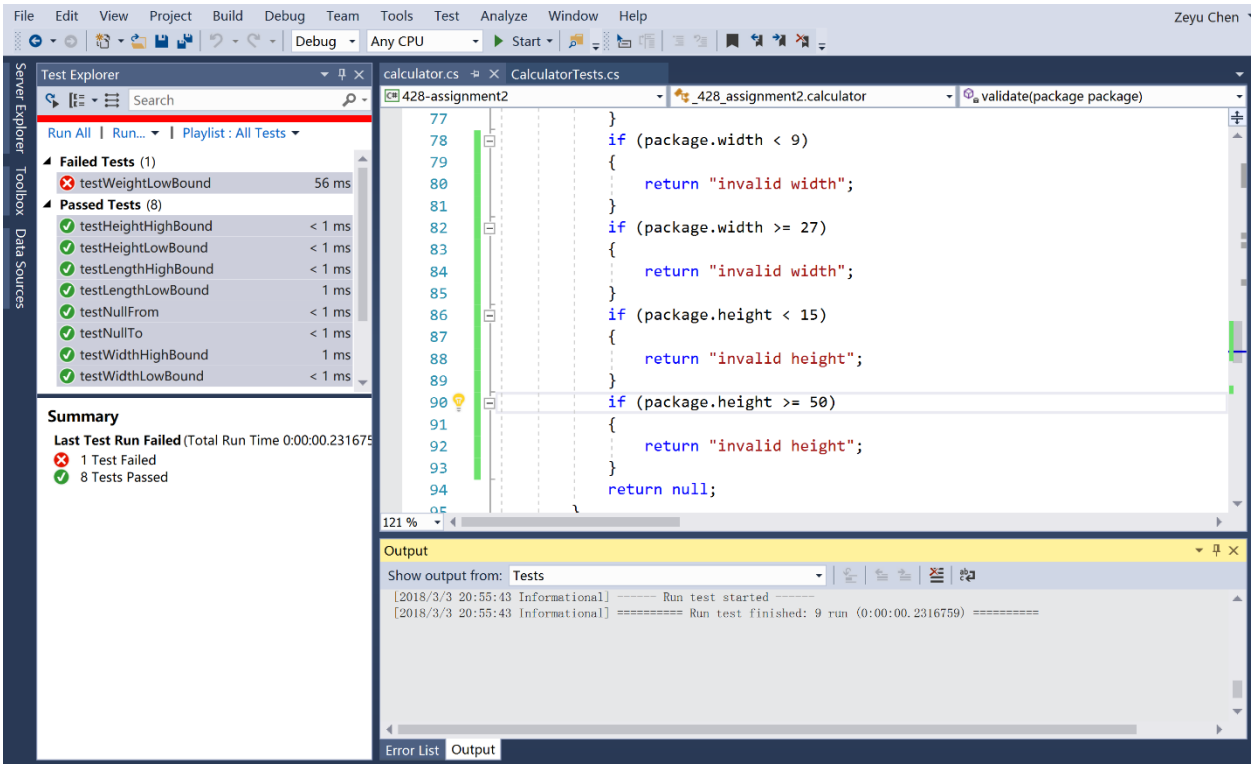
2. Call Setup: create a new package from valid package.

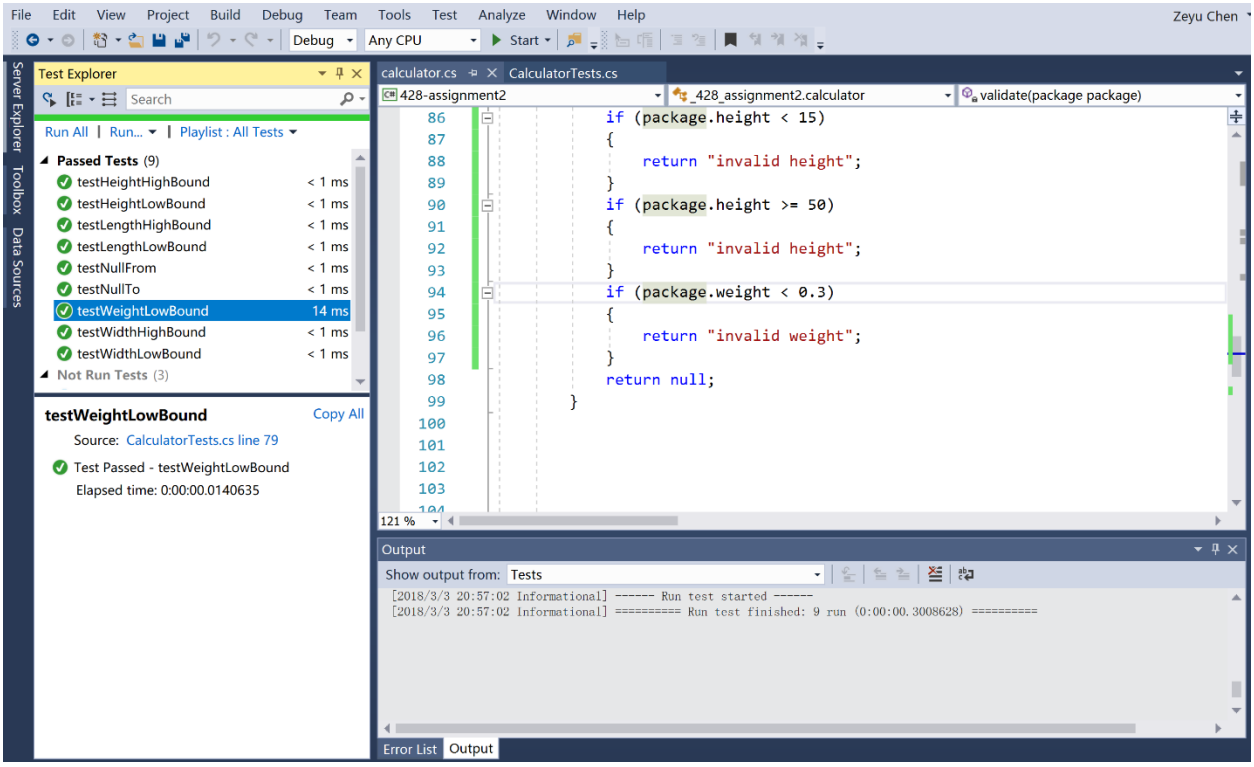
Let package.weight=0.2.

Calculator.getPrice (package).

3. Expected Result: "invalid weight"

4. It was decided that the range of valid weight would be extended down to 0.2kg.





Test10

1.Test name: out_of _range_weight_high

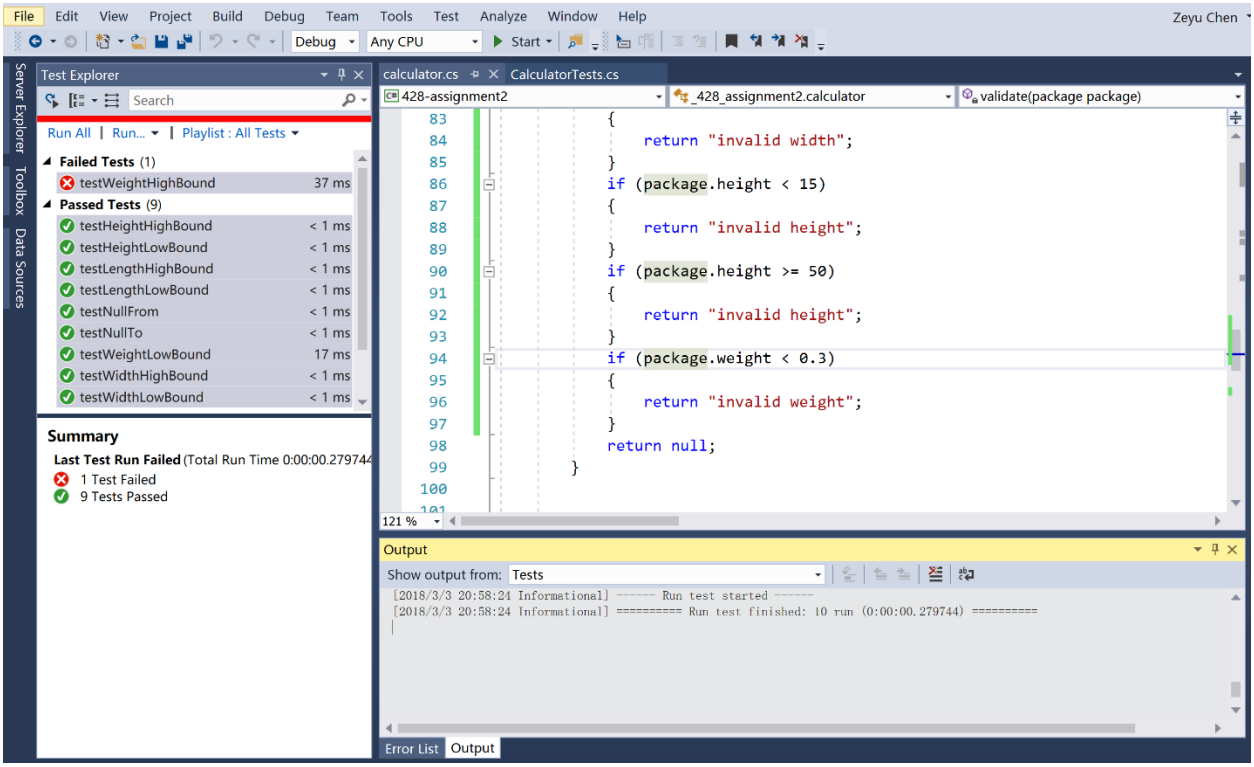
2. Call Setup: create a new package from valid package.

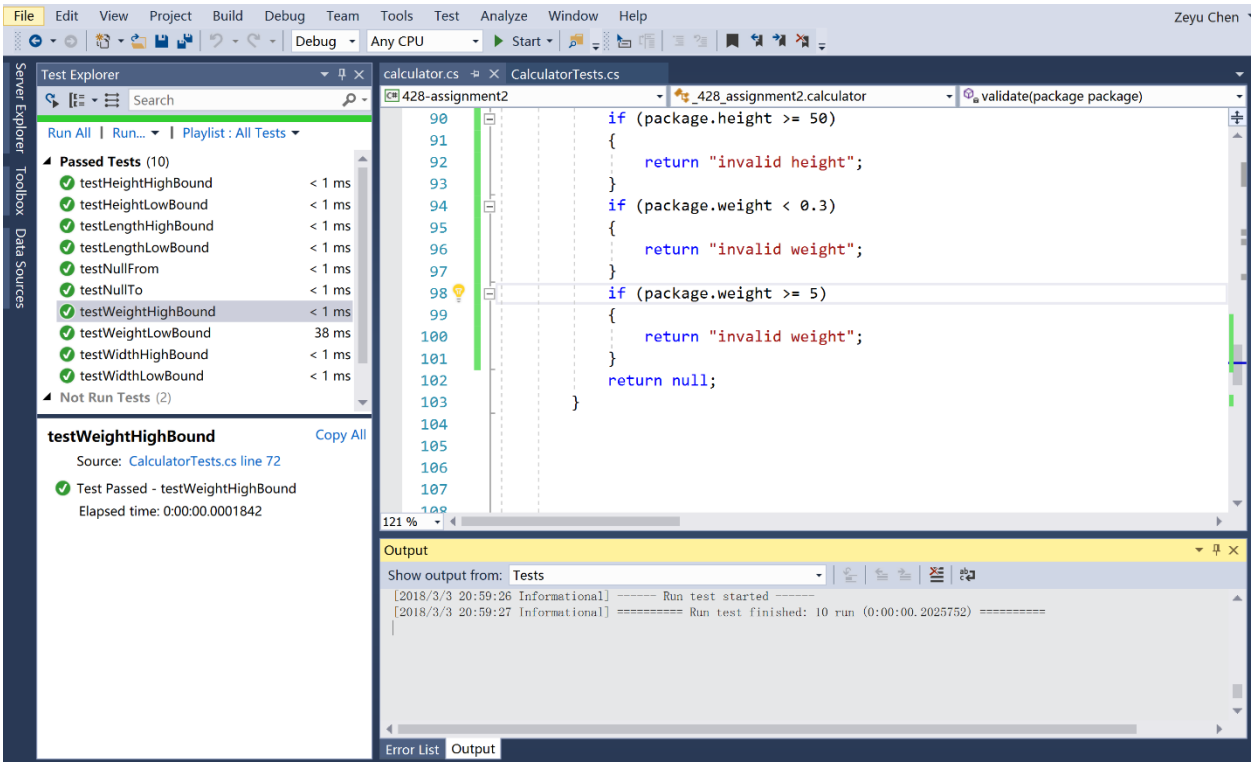
Let package.weight=5.1.

Calculator.getPrice (package).

3. Expected Result: "invalid weight"

4. It was decided that the range of valid weight would be extended up to 5kg.





Test11

1. Test name: invalid postcode_format_from

2. Call Setup: create a new package from valid package.

Let package.from="hhhhhh"

Calculator.getPrice (package).

Let package.from="h2x2e2e"

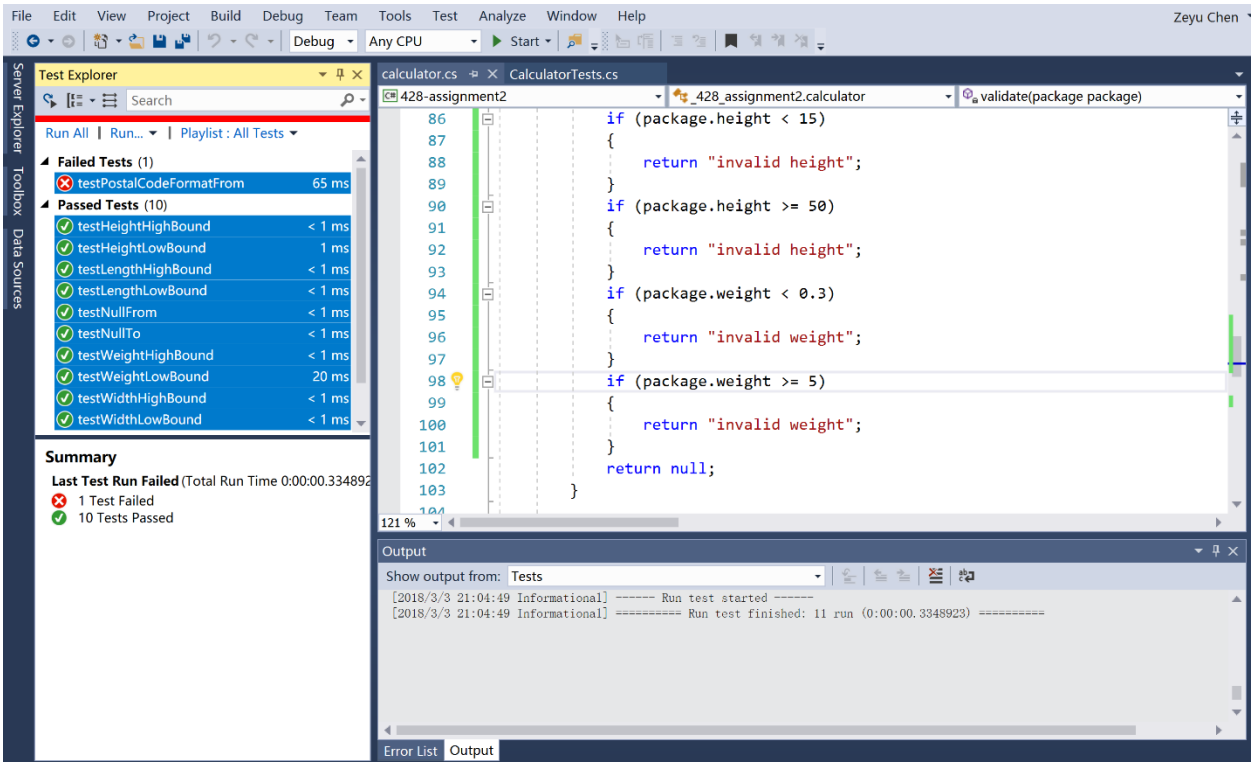
Calculator.getPrice (package).

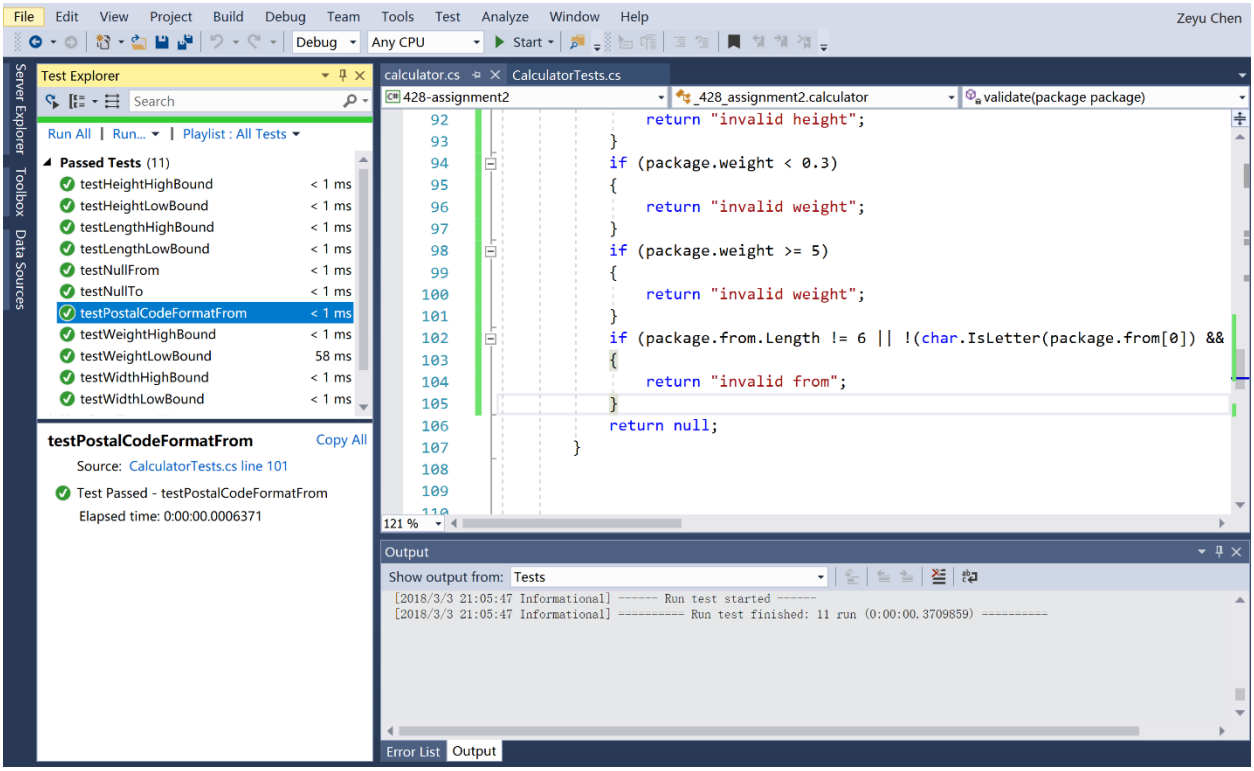
Let package.from="111111"

Calculator.getPrice (package).

3. Expected Result for all 3 situations: "invalid from"

4. This test checks whether the postcode in "from" section is right format.





Test12

1. Test name: invalid postcode_format_to

2. Call Setup: create a new package from valid package.

Let package.to="hhhhhh"

Calculator.getPrice (package).

Let package.to="h2x2e2e"

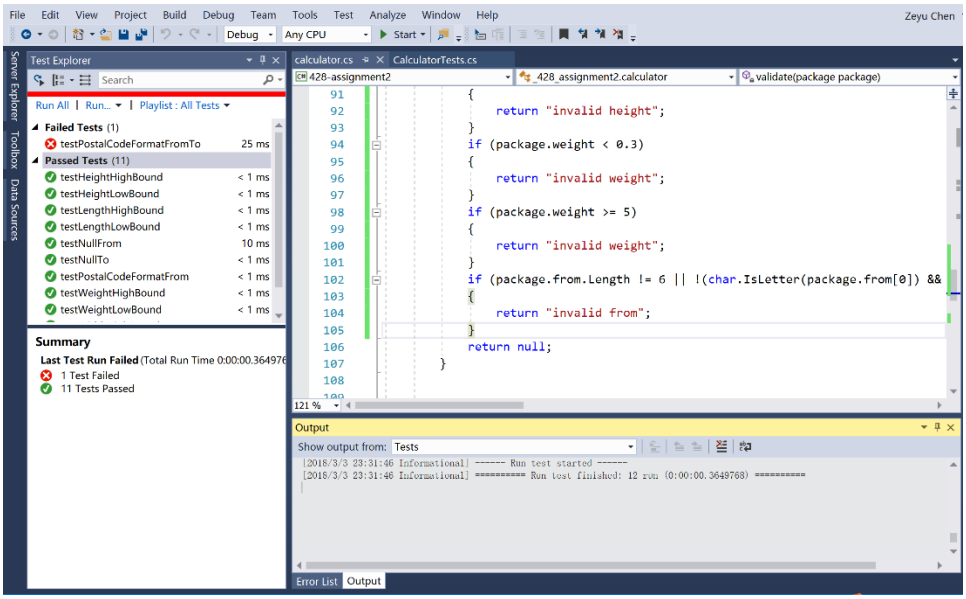
Calculator.getPrice (package).

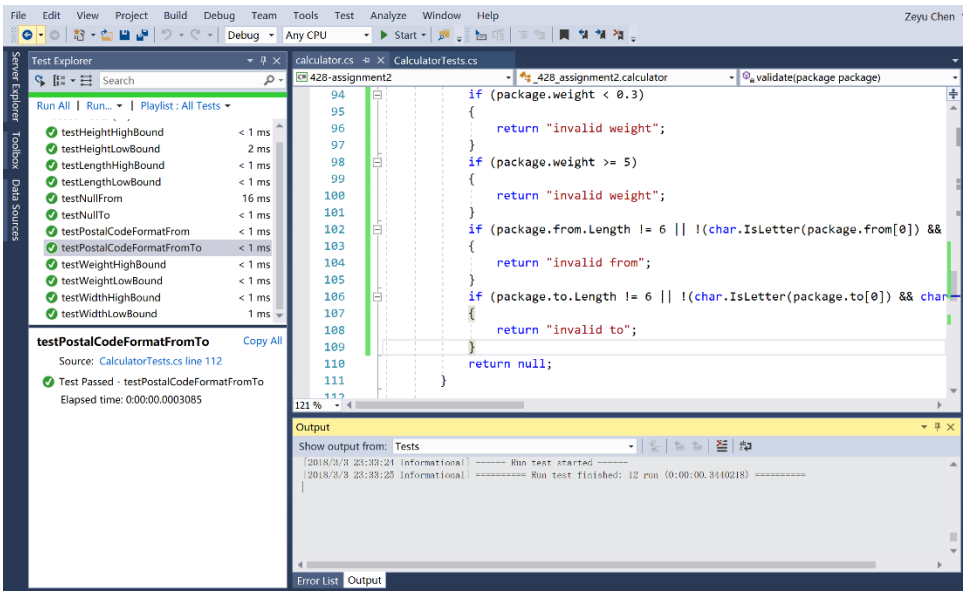
Let package.to="111111"

Calculator.getPrice (package).

3. Expected Result for all 3 situations: "invalid to"

4. This test checks whether the postcode in "from" section is right format.





Test13

1. Test name: valid_ Rule1

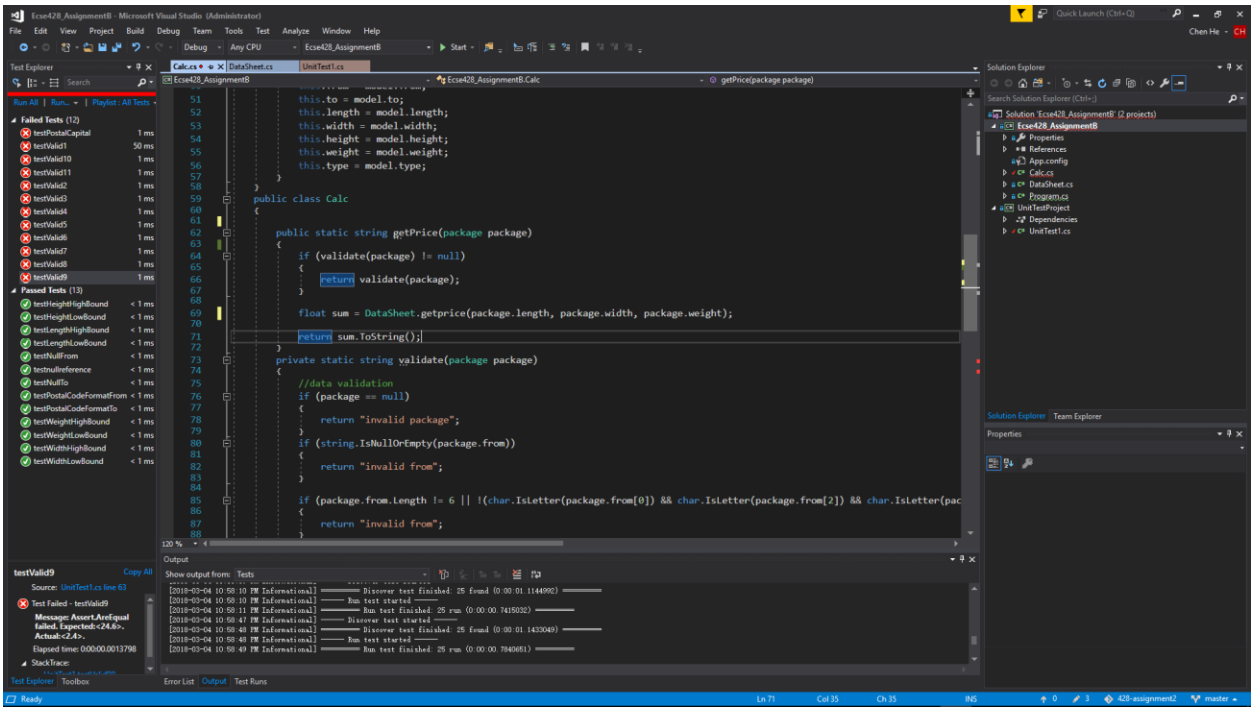
2. Call Setup: create a new package from valid package.

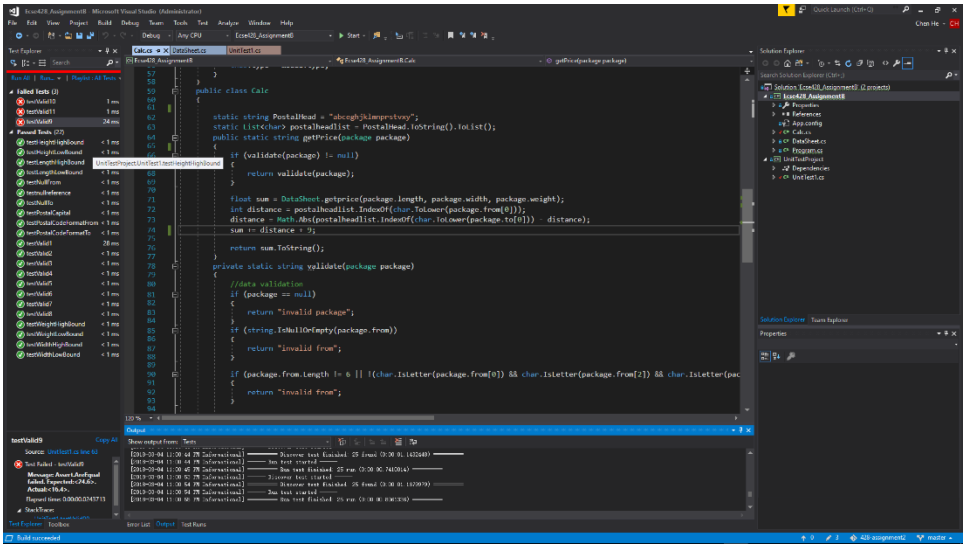
Let package=new package ("h2x2e2", "n4e6p6" , 15, 10, 20, 1, PostType.Regular)

Calculator.getPrice (package).

3. Expected Result: "14.49"

4. This test checks if the answer of the postal rate for standard packages up to 3 kg in regular type from one fixed place to a fixed destination is right.





Test14

1.Test name: valid_ Rule2

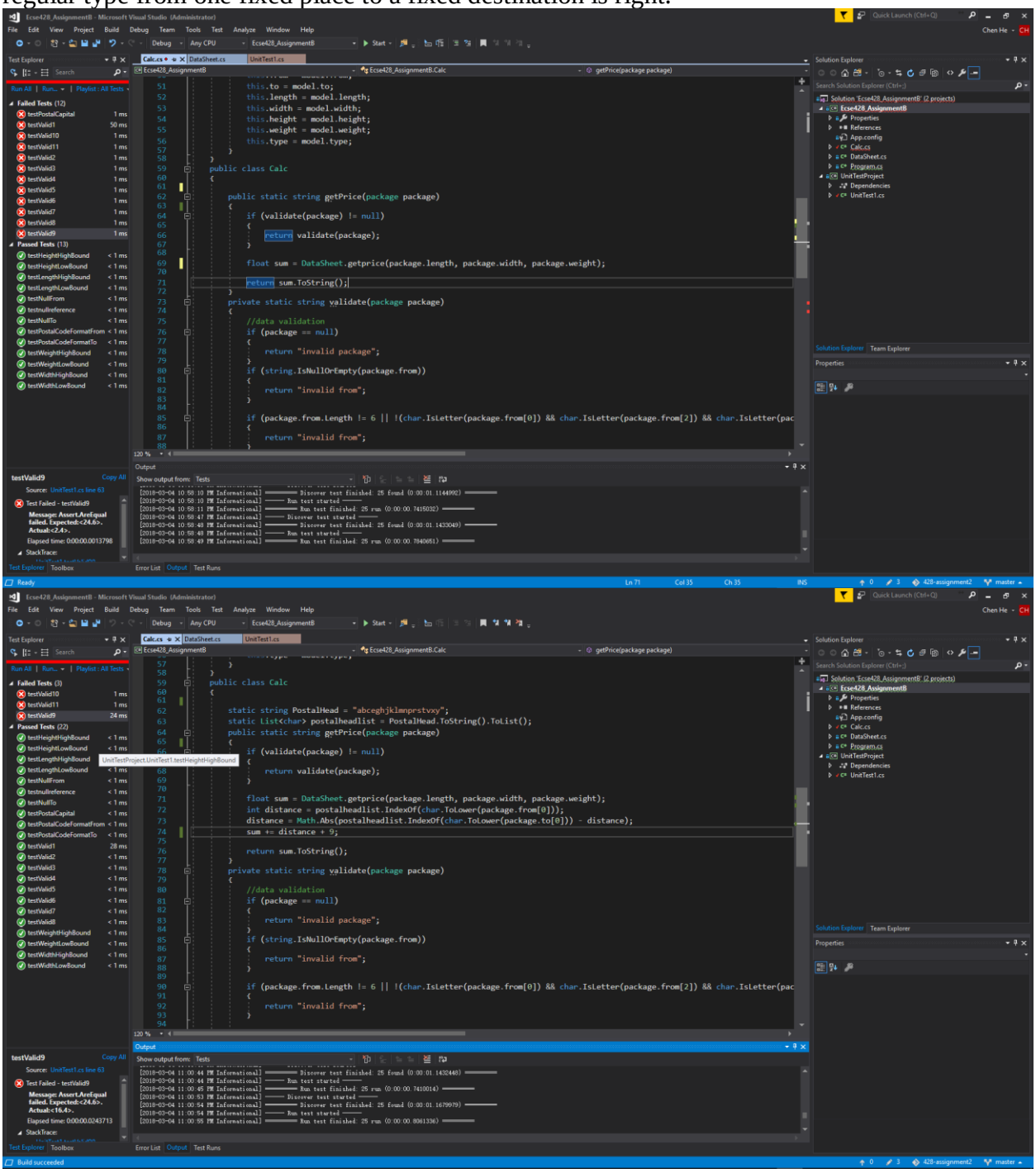
2. Call Setup: create a new package from valid package.

Let package=new package ("h2x2e2", "n4e6p6" , 15, 10, 20, 4, PostType.Regular)

Calculator.getPrice (package).

3. Expected Result: "14.8"

4. This test checks if the answer of the postal rate for standard packages over 3 kg up to 5 kg in regular type from one fixed place to a fixed destination is right.



Test15

1.Test name: valid_ Rule3

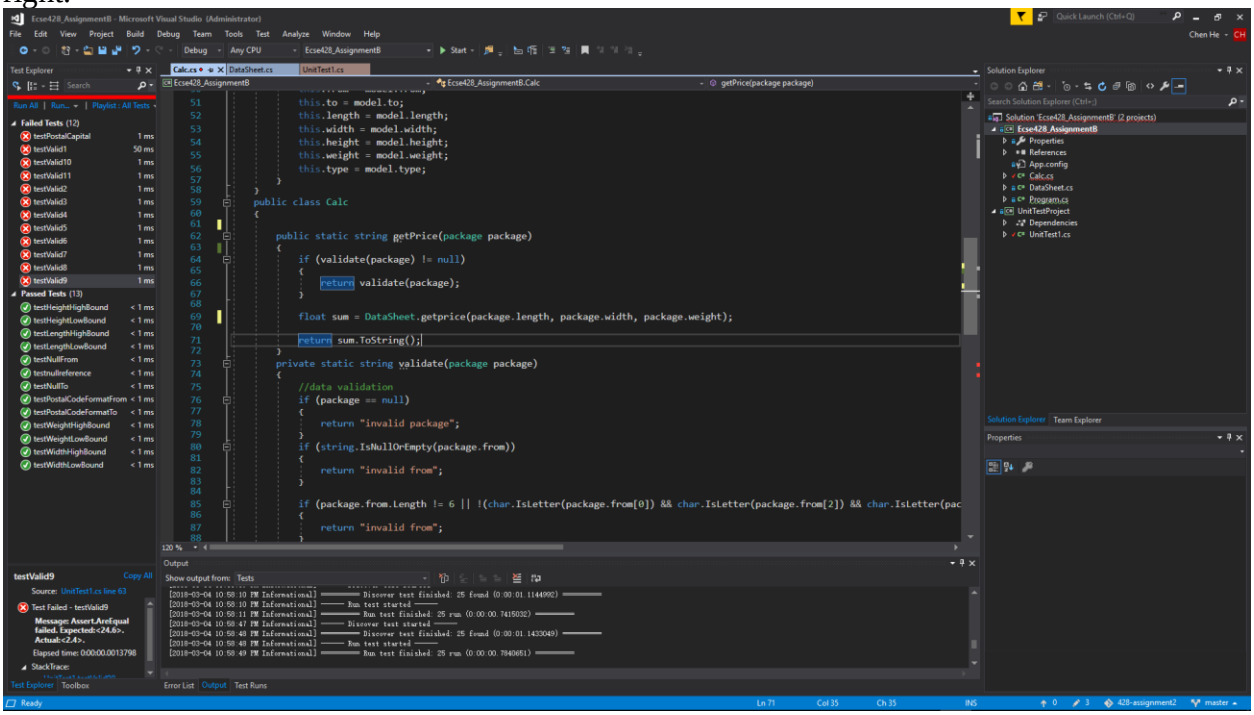
2. Call Setup: create a new package from valid package.

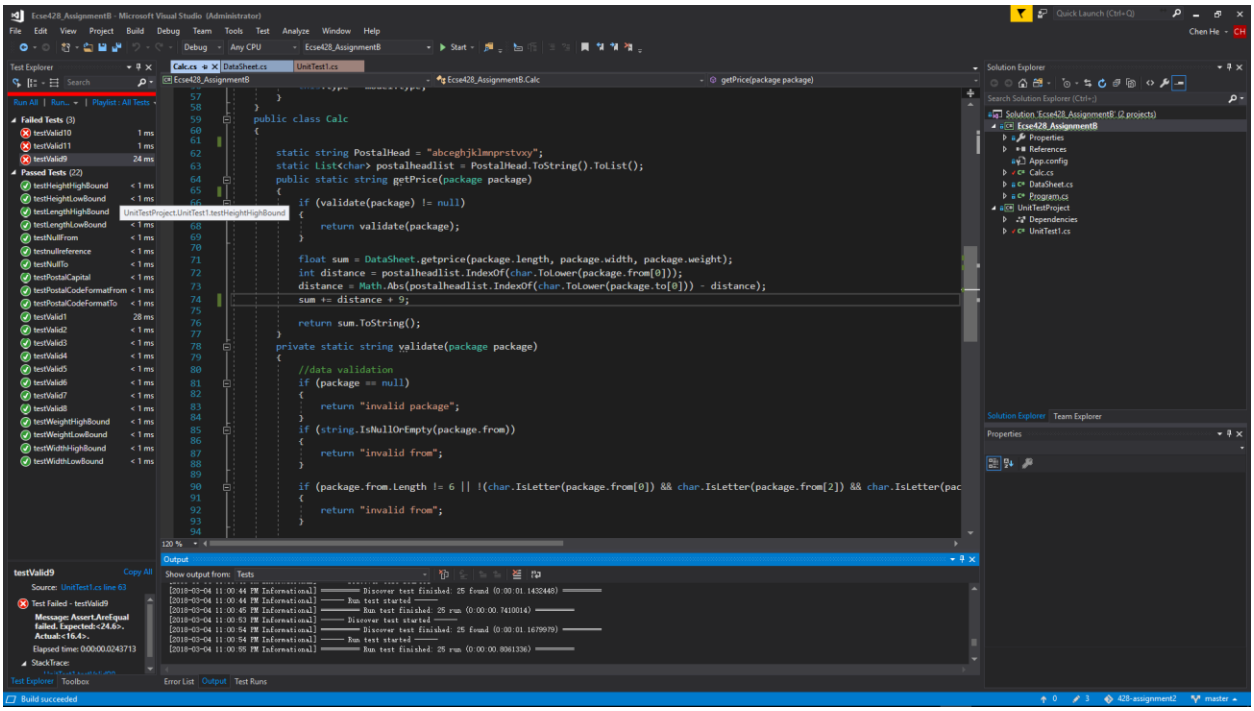
Let package=new package ("h2x2e2", "n4e6p6", 15, 10, 38, 0.8, PostType.Regular)

Calculator.getPrice (package).

3. Expected Result: "14.49"

4. This test checks if the answer of the postal rate for non-standard packages (height in the range of 35cm up to 50cm) up to 1 kg in regular type from one fixed place to a fixed destination is right.





Test16

1.Test name: valid_ Rule4

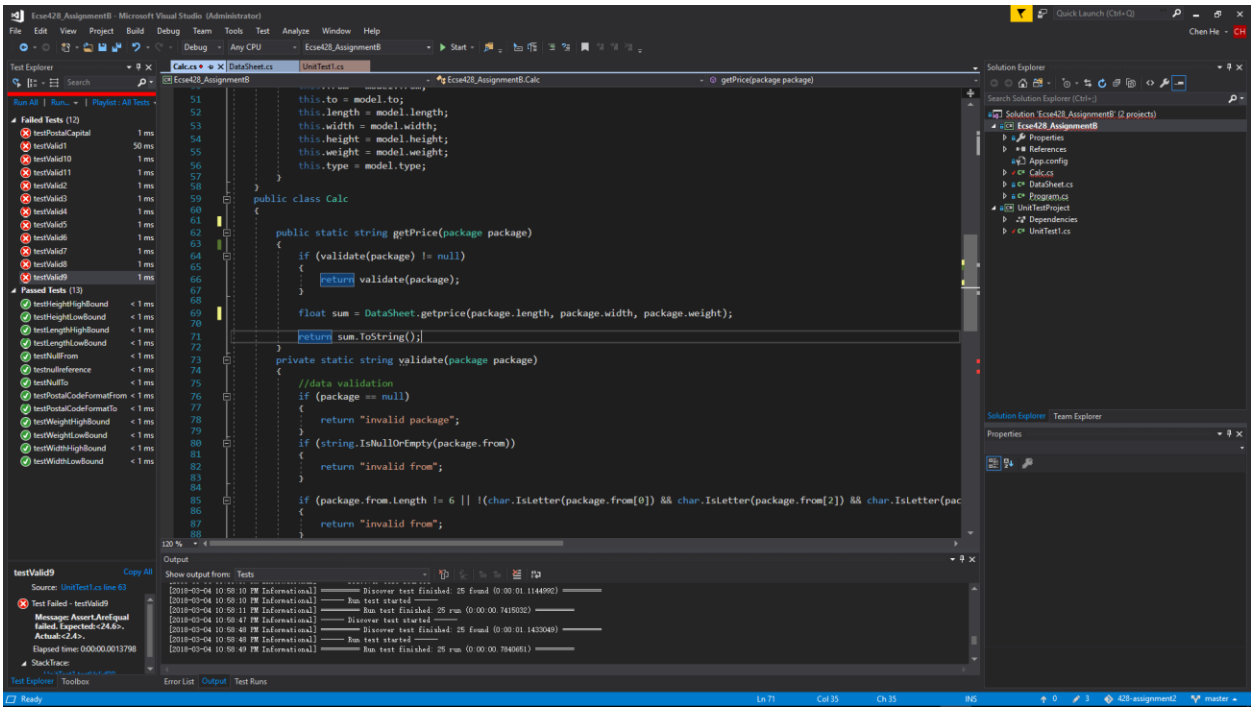
2. Call Setup: create a new package from valid package.

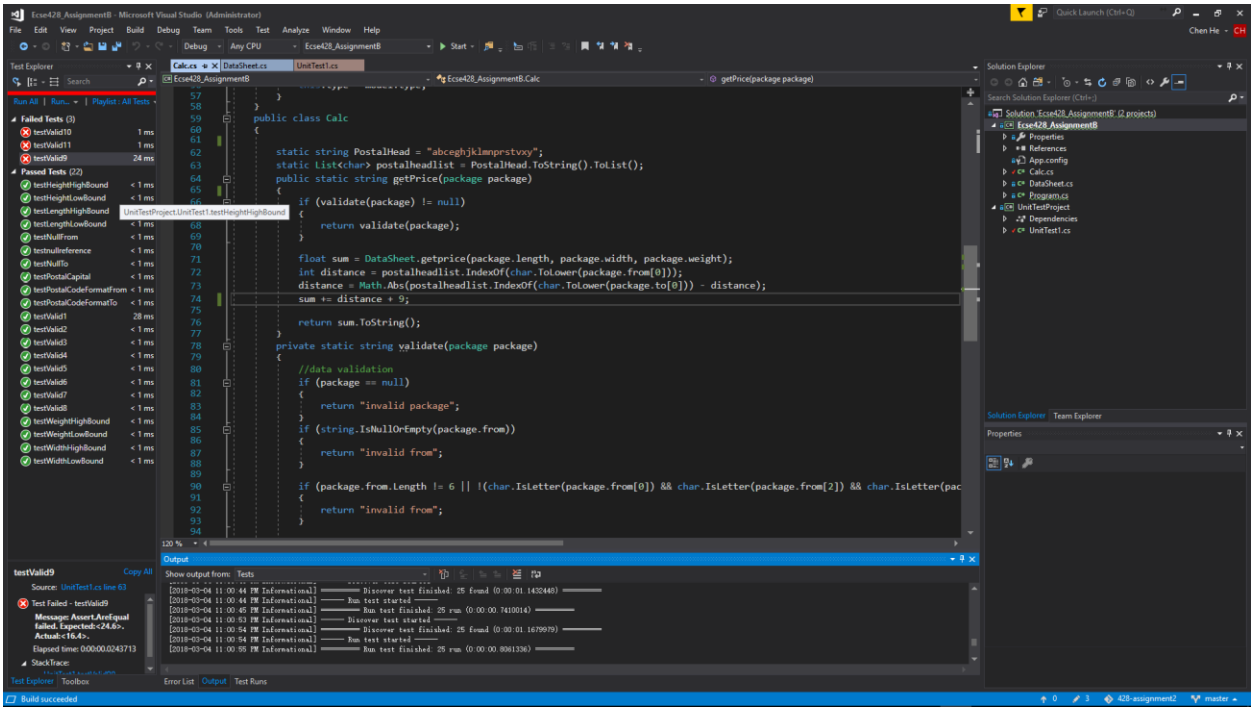
Let package=new package ("h2x2e2", "n4e6p6'' , 15, 10, 38, 2, PostType.Regular)

Calculator.getPrice (package).

3. Expected Result: "14.49"

4. This test checks if the answer of the postal rate for non-standard packages (height in the range of 35cm up to 50cm) over 1 kg up to 5 kg in regular type from one fixed place to a fixed destination is right.





Test17

1.Test name: valid_ Rule5

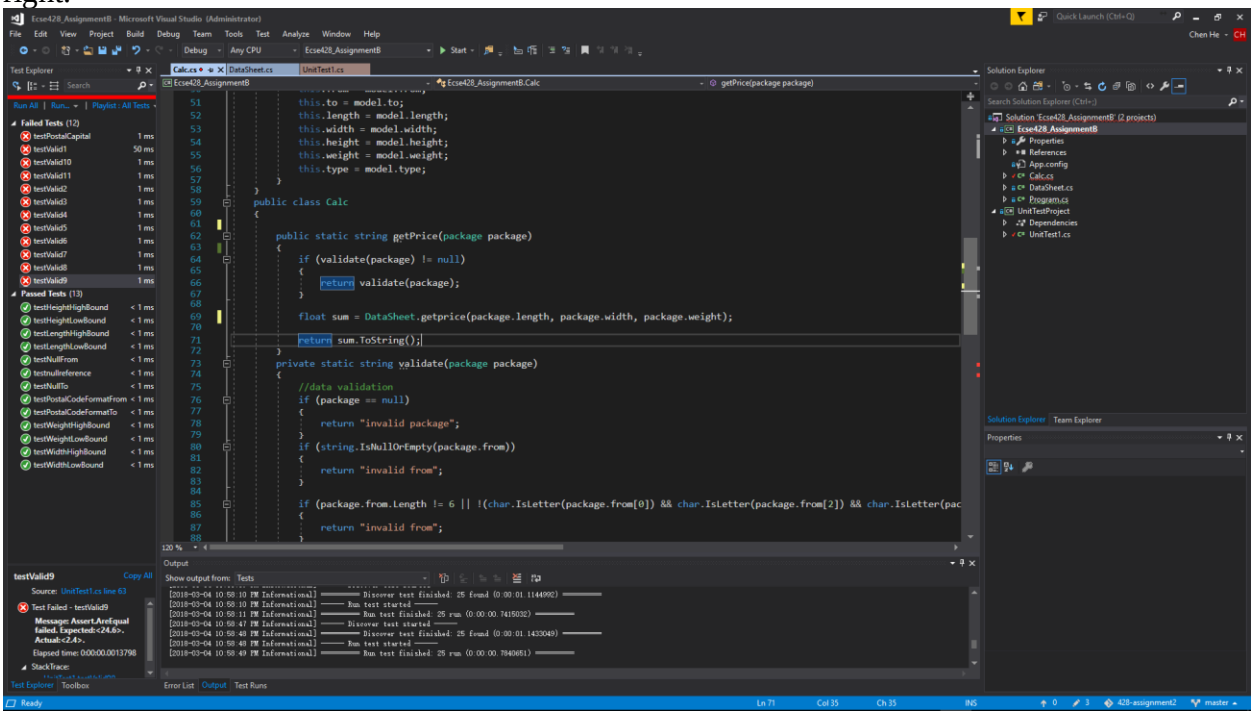
2. Call Setup: create a new package from valid package.

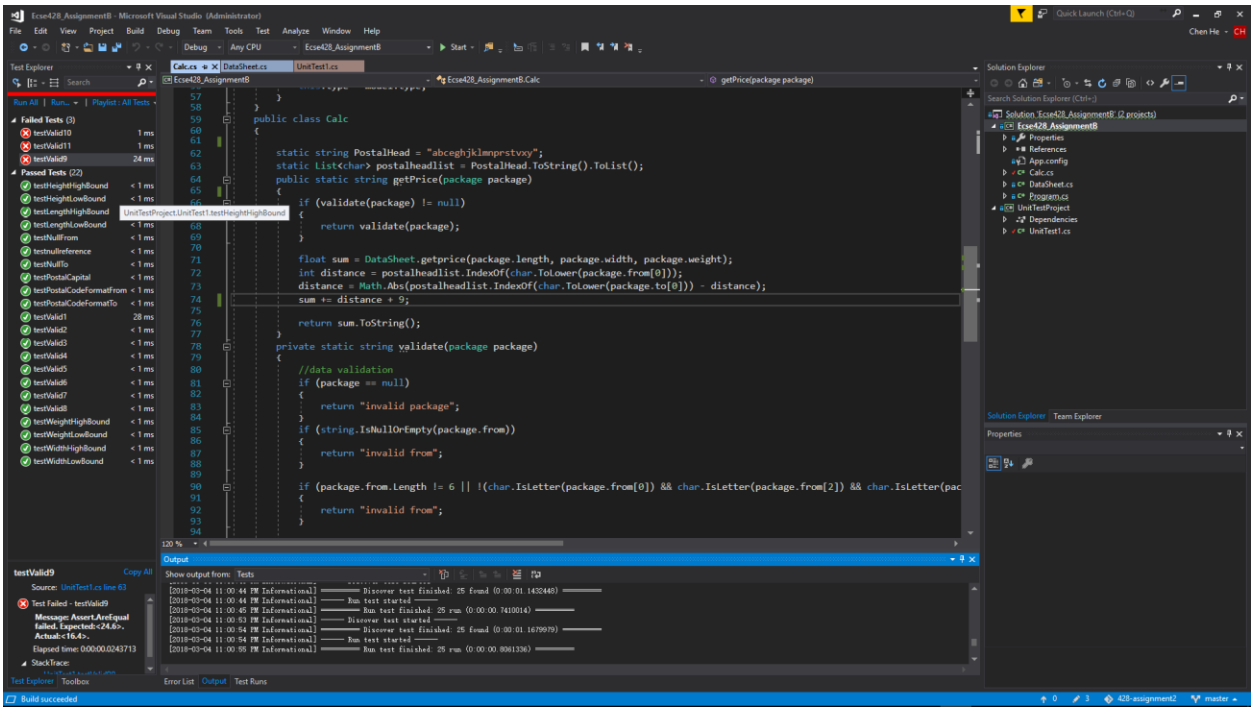
Let package=new package ("h2x2e2", "n4e6p6" , 28, 10, 20, 0.8, PostType.Regular)

Calculator.getPrice (package).

3. Expected Result: "14.98"

4. This test checks if the answer of the postal rate for non-standard packages (length in the range of 24.5cm up to 38cm) up to 1 kg in regular type from one fixed place to a fixed destination is right.





Test18

1.Test name: valid_ Rule6

2. Call Setup: create a new package from valid package.

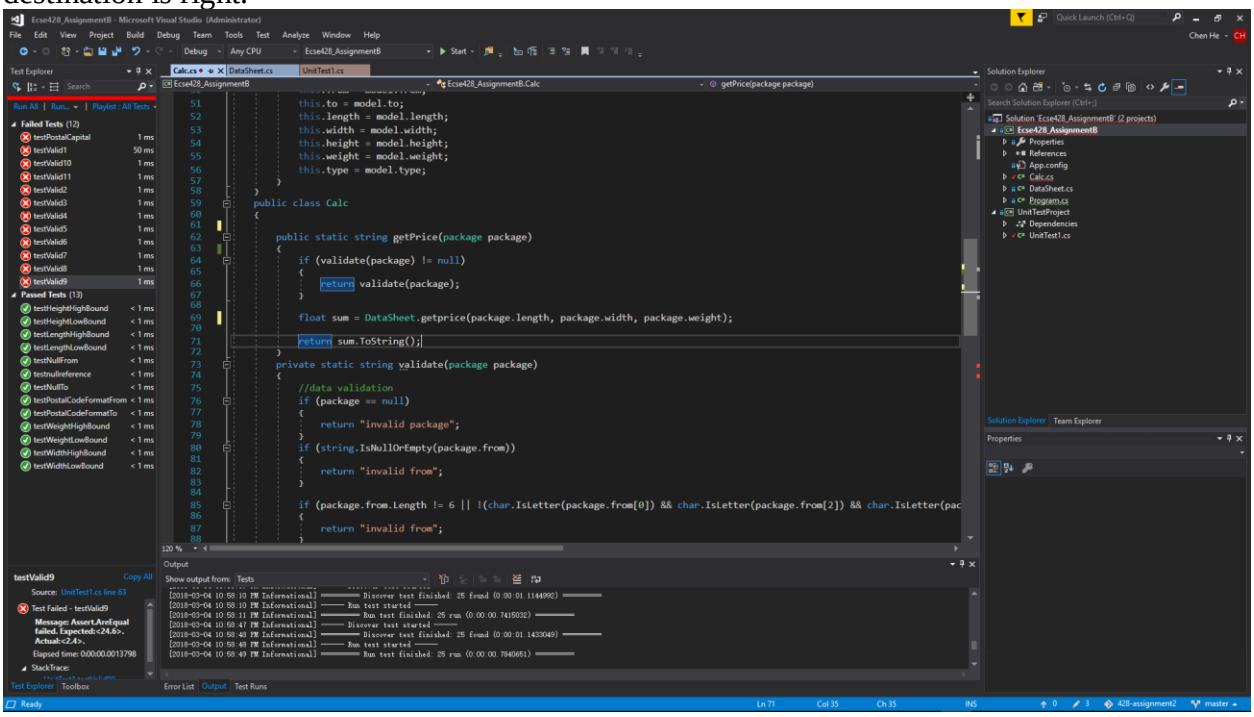
Let package=new package ("h2x2e2", "n4e6p6", 28, 10, 20, 2, PostType.Regular)

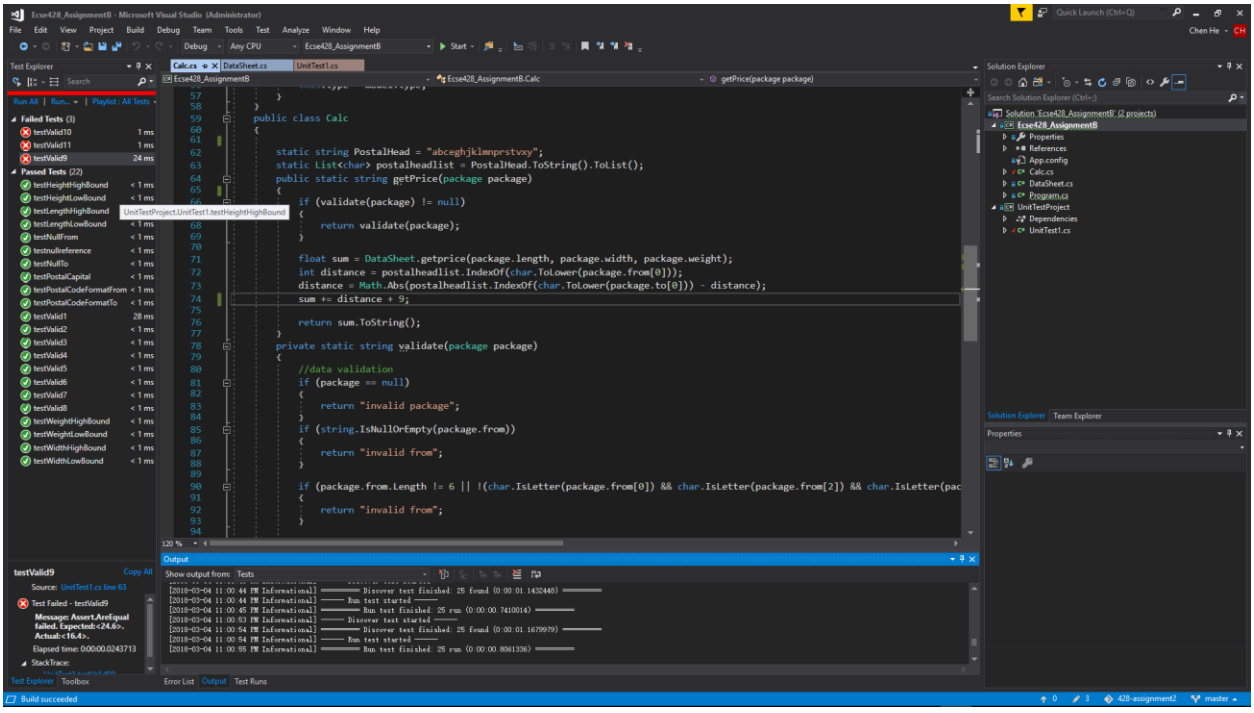
Calculator.getPrice (package).

3. Expected Result: "16.4"

4. This test checks if the answer of the postal rate for non-standard packages (width in the range of 24.5cm up to 38cm) over 1 kg up to 5 kg in regular type from one fixed place to a fixed

destination is right.





Test19

1.Test name: valid_ Rule7

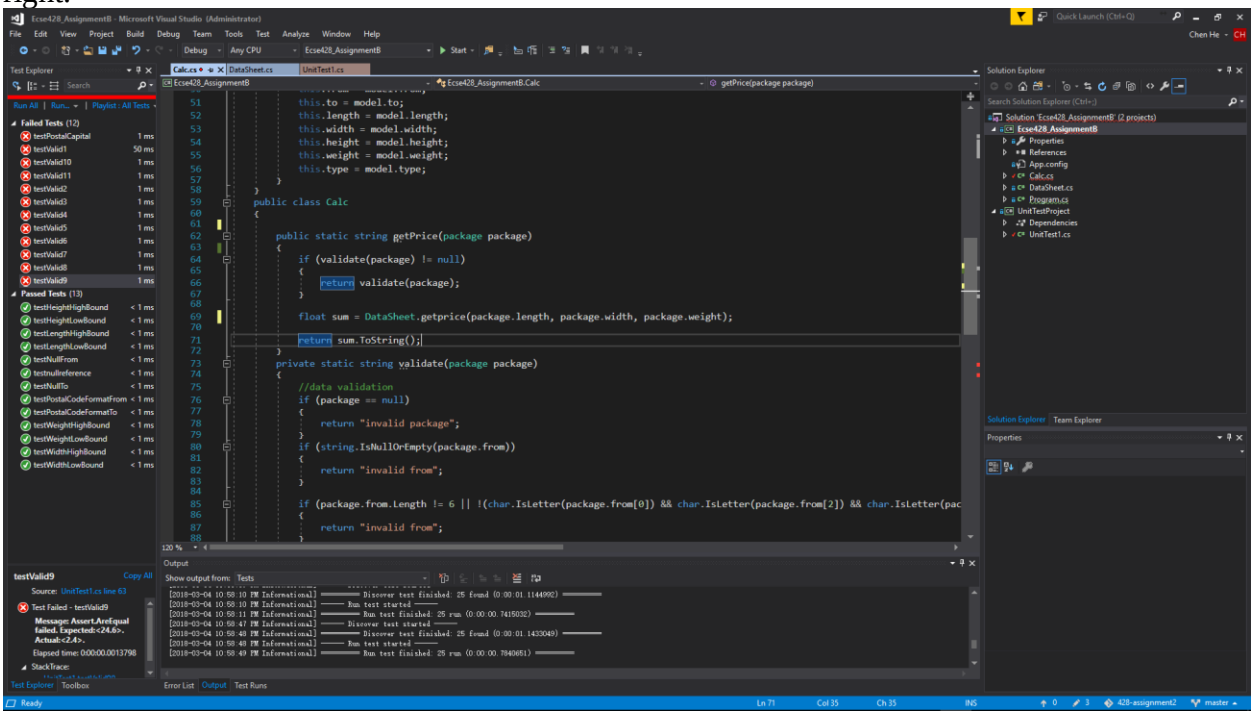
2. Call Setup: create a new package from valid package.

Let package=new package ("h2x2e2", "n4e6p6", 15, 19, 20, 0.8, PostType.Regular)

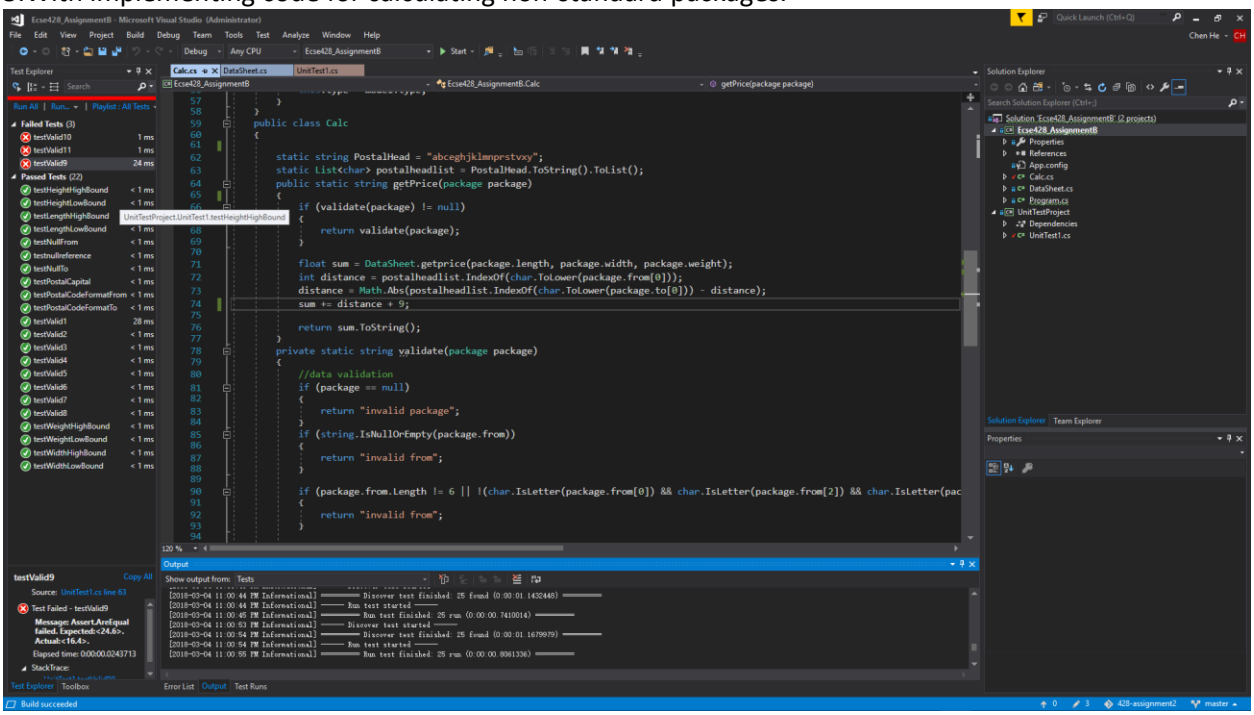
Calculator.getPrice (package).

3. Expected Result: "14.98"

4. This test checks if the answer of the postal rate for non-standard packages (width in the range of 15.6cm up to 27cm) up to 1 kg in regular type from one fixed place to a fixed destination is right.



5.With implementing code for calculating non-standard packages.



Test20

1.Test name: valid_ Rule8

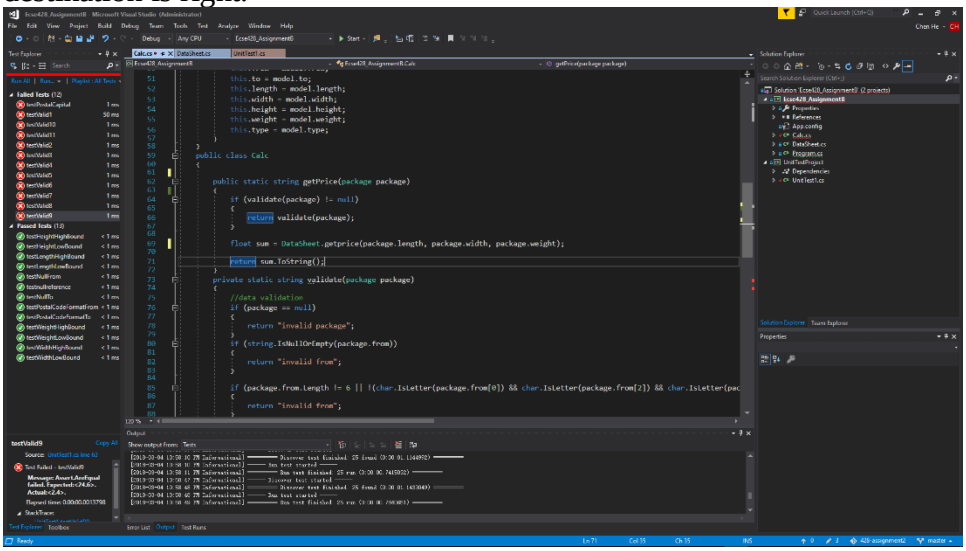
2. Call Setup: create a new package from valid package.

Let package=new package ("h2x2e2", "n4e6p6", 15, 19, 20, 2, PostType.Regular)

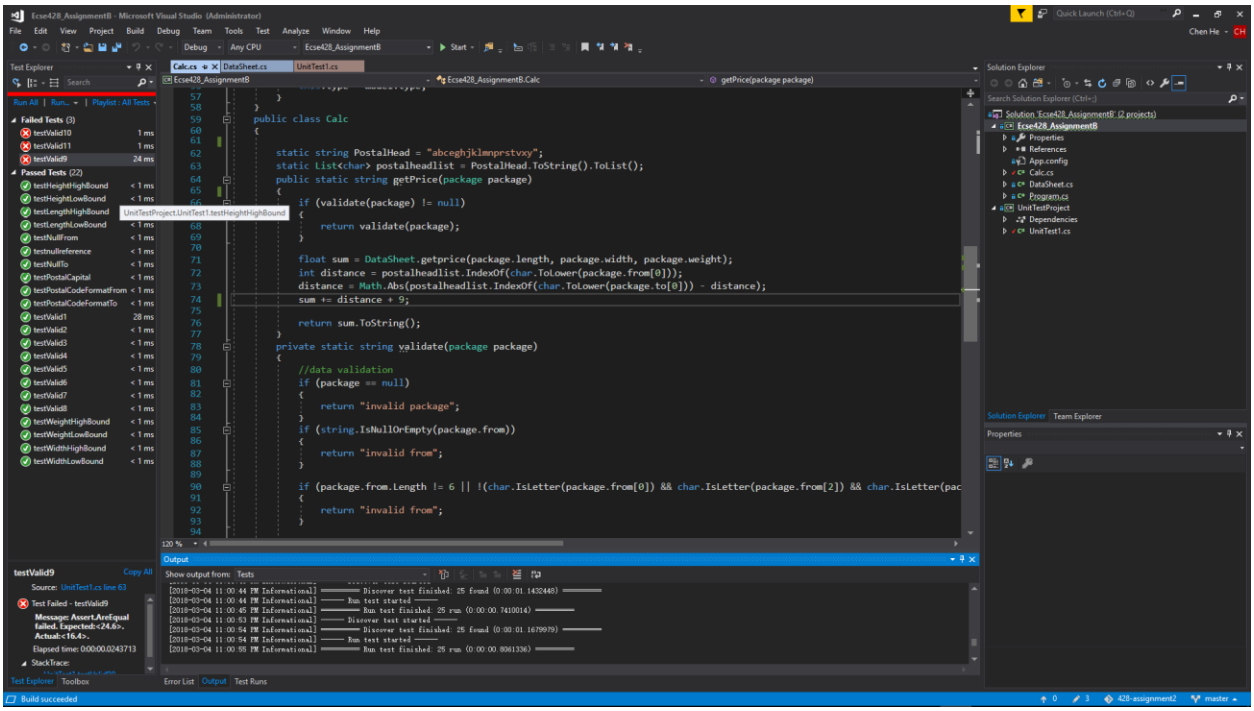
Calculator.getPrice (package).

3. Expected Result: "16.4"

4. This test checks if the answer of the postal rate for non-standard packages (width in the range of 15.6cm up to 27cm) over 1 kg up to 5 kg in regular type from one fixed place to a fixed destination is right.



5.With implementing the code



Test21

1.Test name: valid_ Rule9

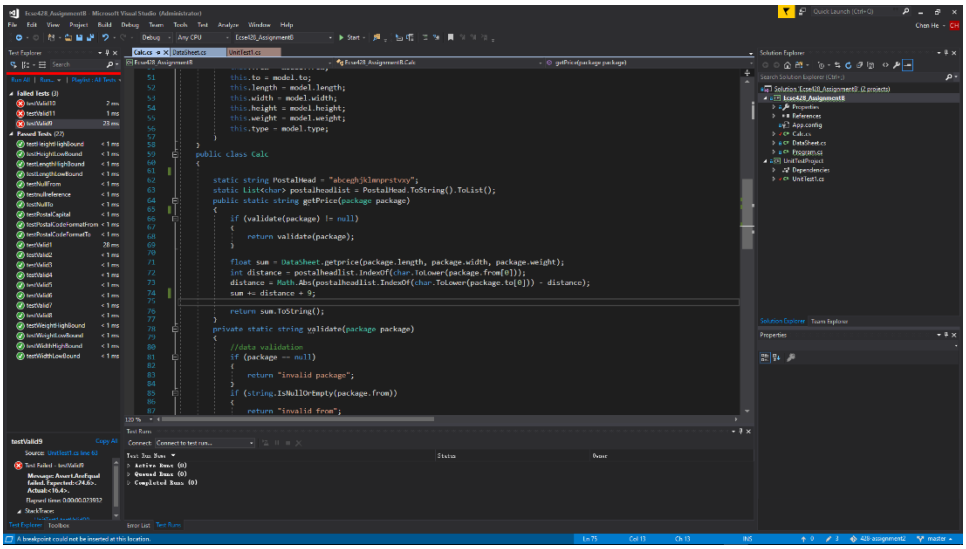
2. Call Setup: create a new package from valid package.

Let package=new package ("h2x2e2", "n4e6p6", 15, 19, 20, 2, PostType.Xpress)

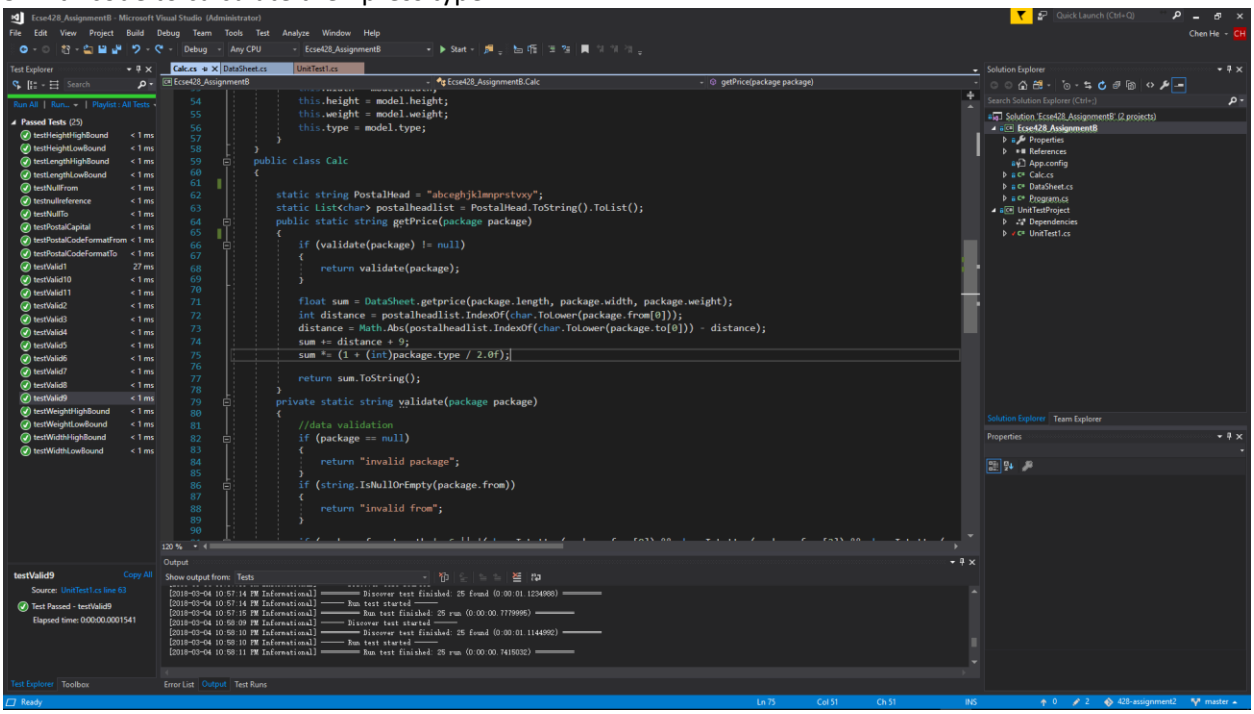
Calculator.getPrice (package).

3. Expected Result: "24.6"

4. This test checks if the answer of the postal rate for non-standard packages (width in the range of 15.6cm up to 27cm) over 1 kg up to 5 kg in Xpress type from one fixed place to a fixed destination is right.



5.With code to calculate the Xpress type



Test22

1.Test name: valid_ Rule10

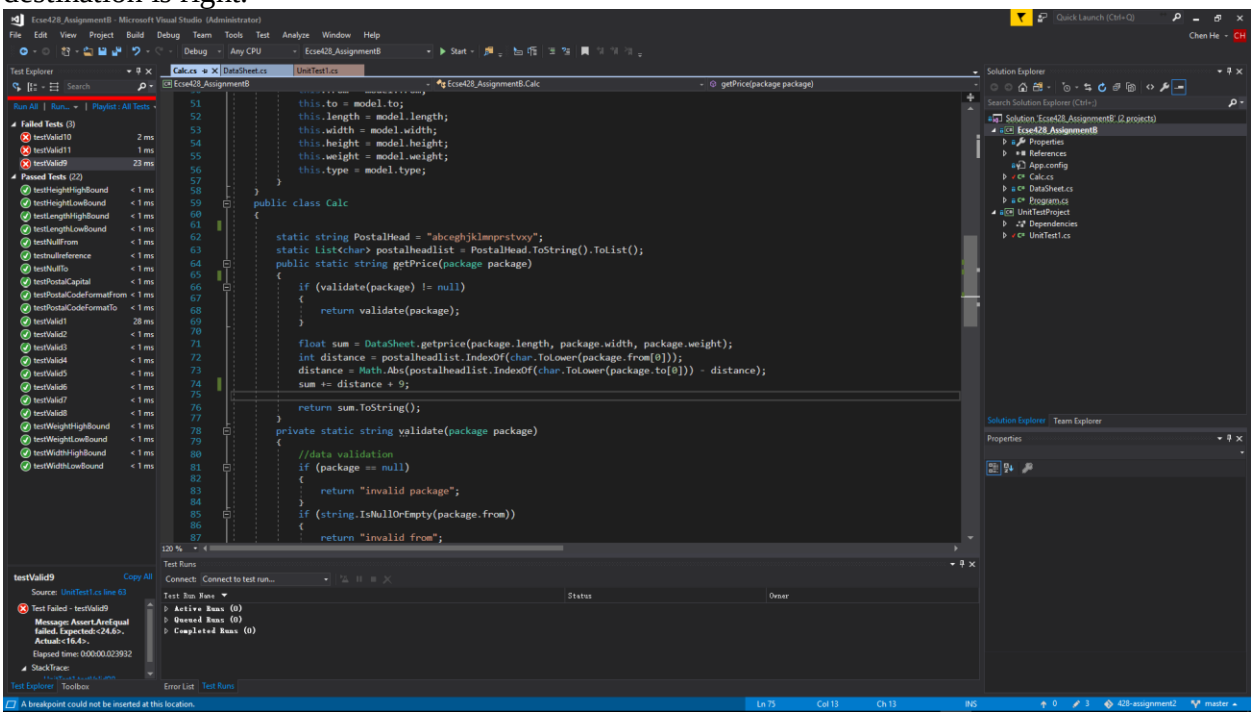
2. Call Setup: create a new package from valid package.

Let package=new package ("h2x2e2", "n4e6p6" , 15, 19, 20, 2, PostType.Priority)

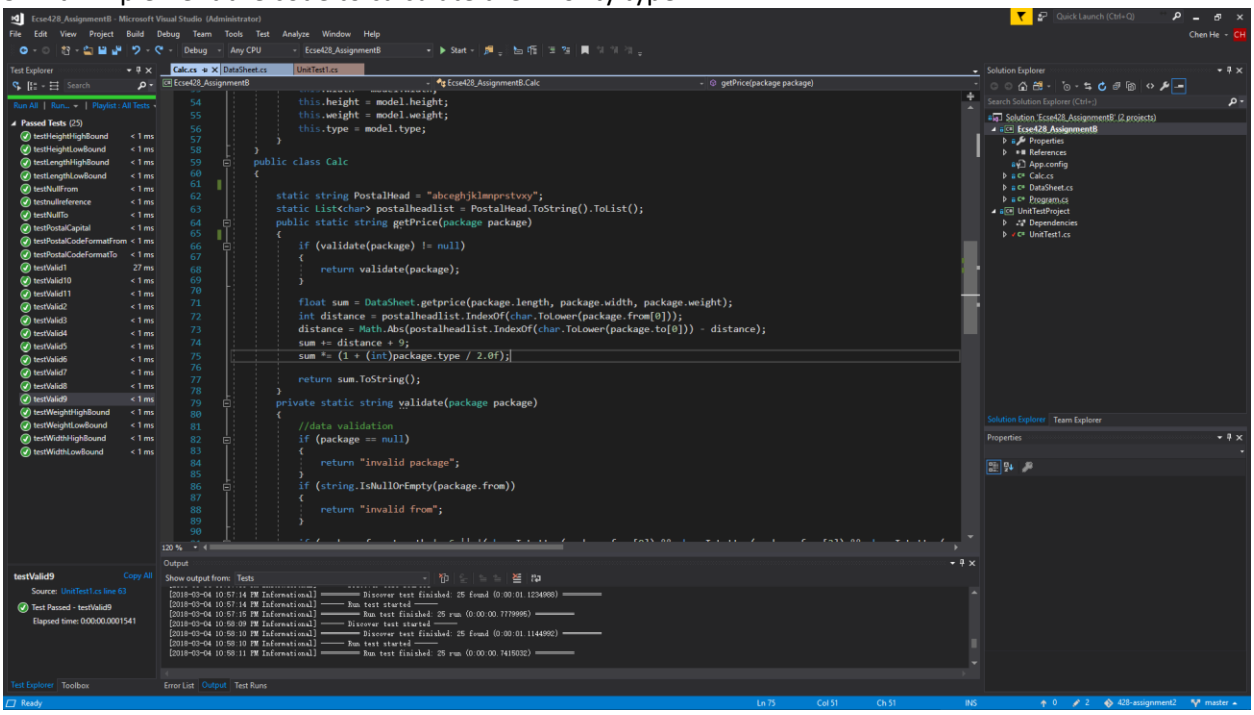
Calculator.getPrice (package).

3. Expected Result: "32.8"

4. This test checks if the answer of the postal rate for non-standard packages (width in the range of 15.6cm up to 27cm) over 1 kg up to 5 kg in Priority type from one fixed place to a fixed destination is right.



5.With implement the code to calculate the Priority type.



Test23

1.Test name: valid_ Rule11

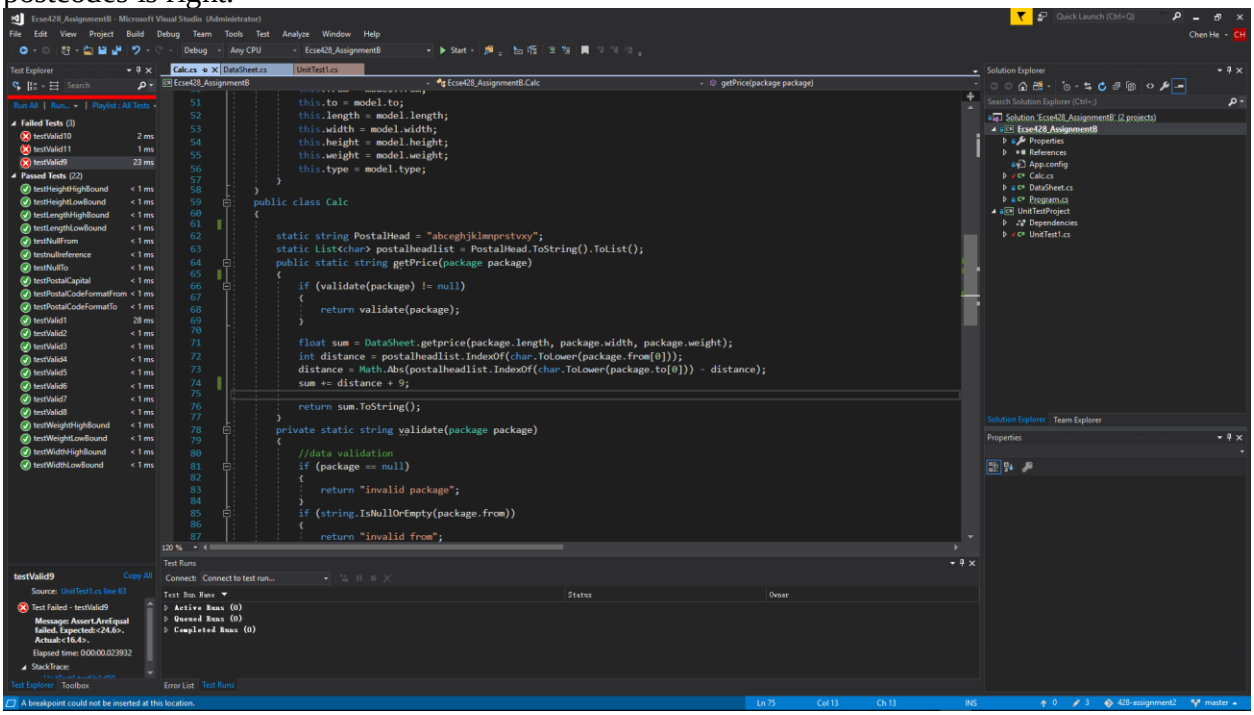
2. Call Setup: create a new package from valid package.

Let package=new package ("H2L2C9", "H2X1J5", 15, 19, 20, 2, PostType.Priority)

Calculator.getPrice (package).

3. Expected Result: "22.8"

4. This test checks if the answer of the postal rate for non-standard packages (width in the range of 15.6cm up to 27cm) over 1 kg up to 5 kg in Priority type with different "from" and "to" postcodes is right.



5.With implement the code that calculate the distance between two postcode.