ECSE 428 – Software Engineering Practice
Assignment B: Test Driven Development
Winter 2014

Dao, Nhat-Quang 260457711
Singzon, Ryan 260397455

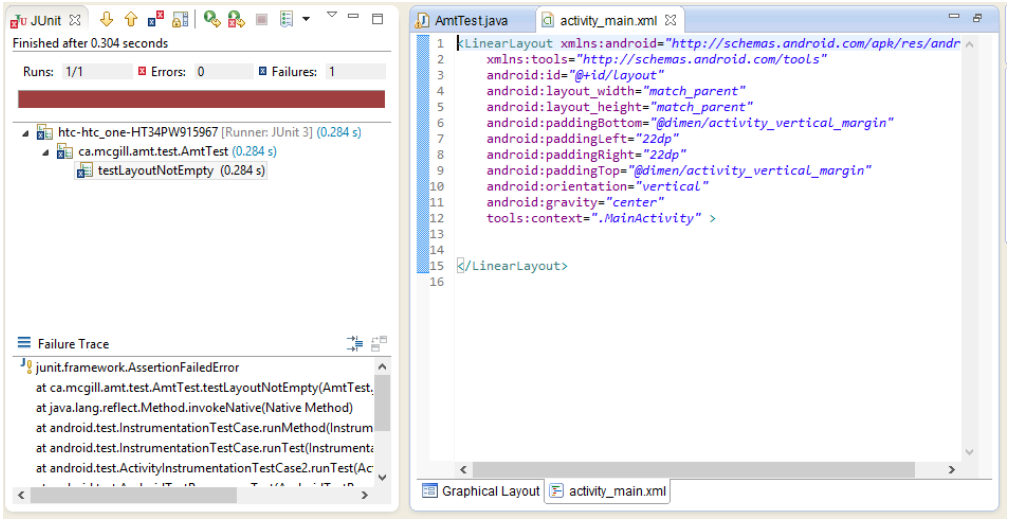
Contents

Domain Logic Tests

1. The layout contains any number of elements

Check if the application is properly formatted by verifying that that elements appear on screen
Setup: None
Expected Result: The layout should have elements in which the user can input their specifications

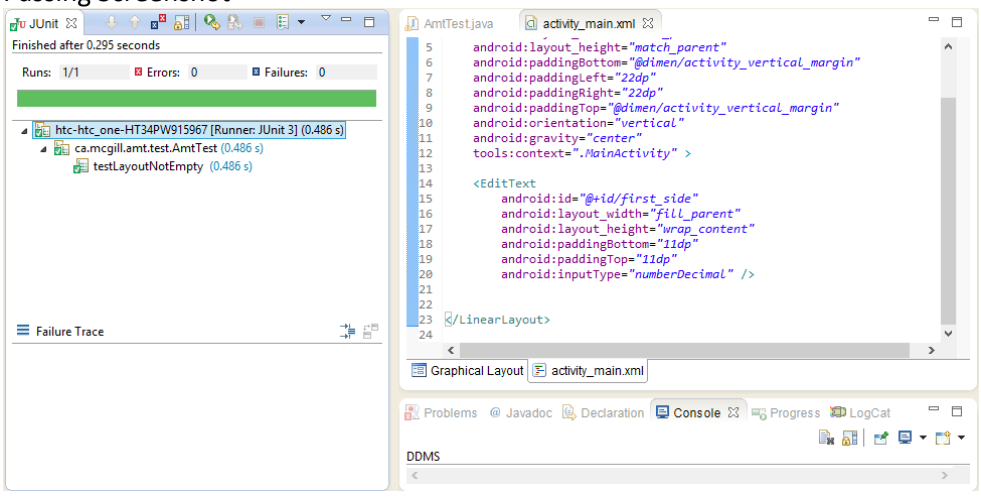
Failing Screenshot:



Code added

```
<EditText
  android:id="@+id/first_side"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:paddingBottom="11dp"
  android:paddingTop="11dp"
  android:hint="@+string/enter_first_side"
        android:inputType="numberDecimal" />
```

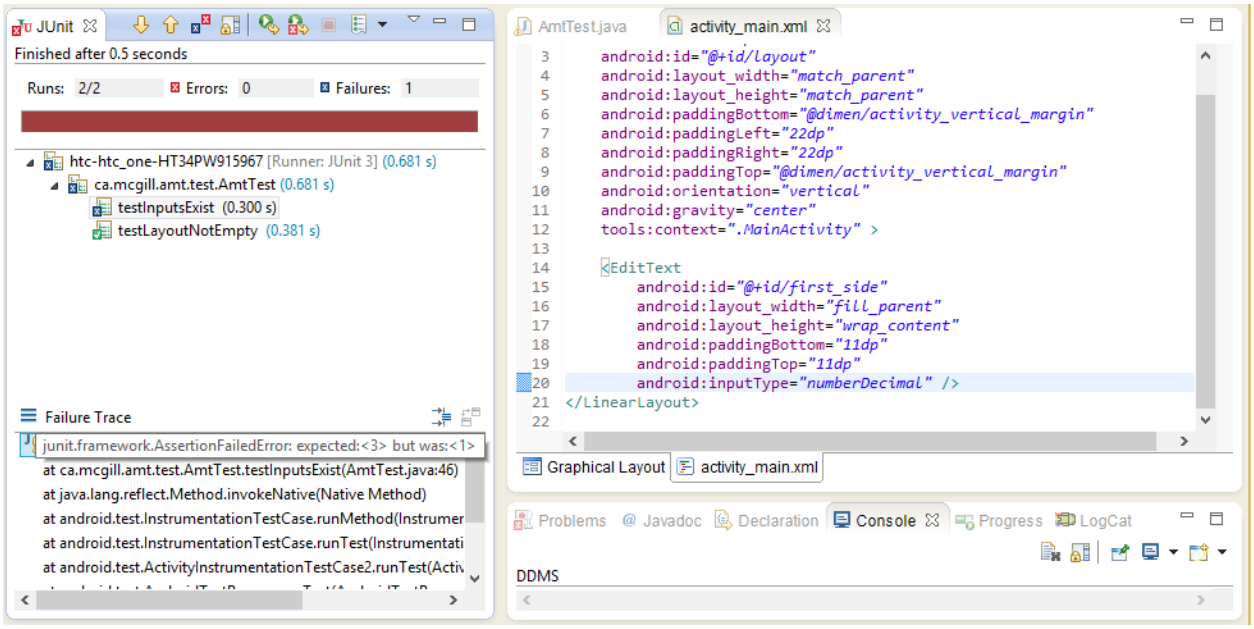
Passing Screenshot



2. Three input fields exist

Verify that the input fields for the user exist
Setup: None
Expected result: The layout should contain three input fields, one for each side of the triangle

Failing Screenshot:

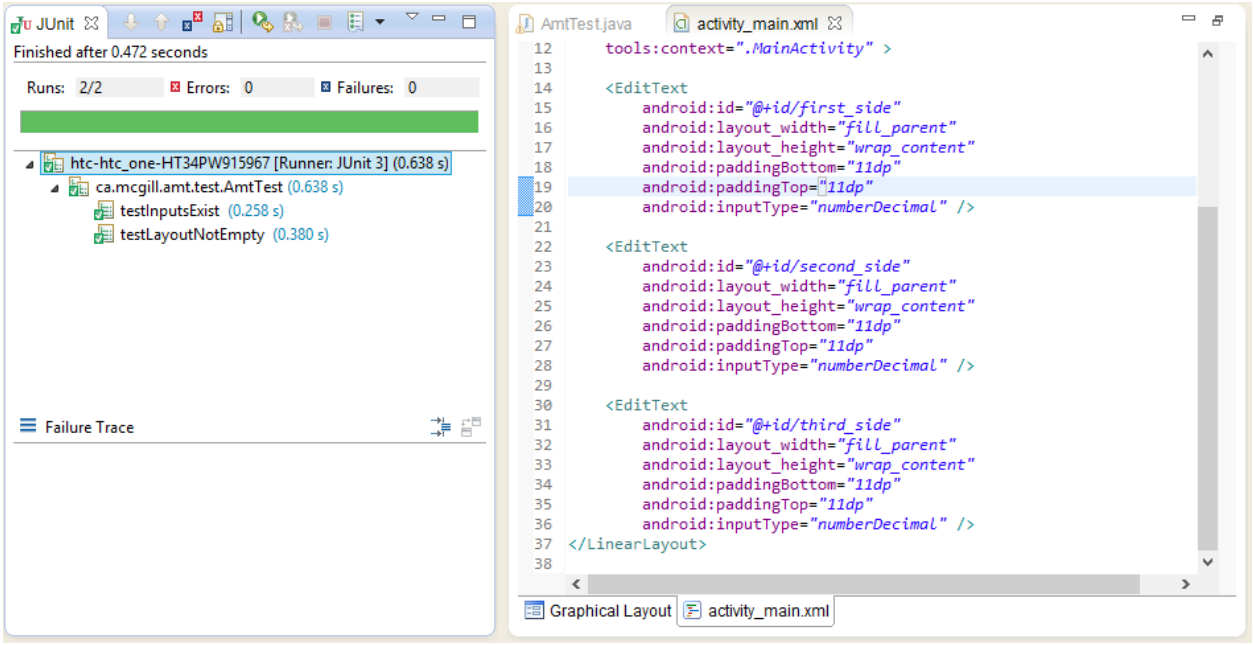


Code added:

```
<EditText
  android:id="@+id/second_side"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:paddingBottom="11dp"
  android:paddingTop="11dp"
  android:hint="@+string/enter_second_side"
  android:inputType="numberDecimal" />

 <EditText
  android:id="@+id/third_side"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:paddingBottom="11dp"
  android:paddingTop="11dp"
  android:hint="@+string/enter_third_side"
        android:inputType="numberDecimal" />
```

Passing screenshot:

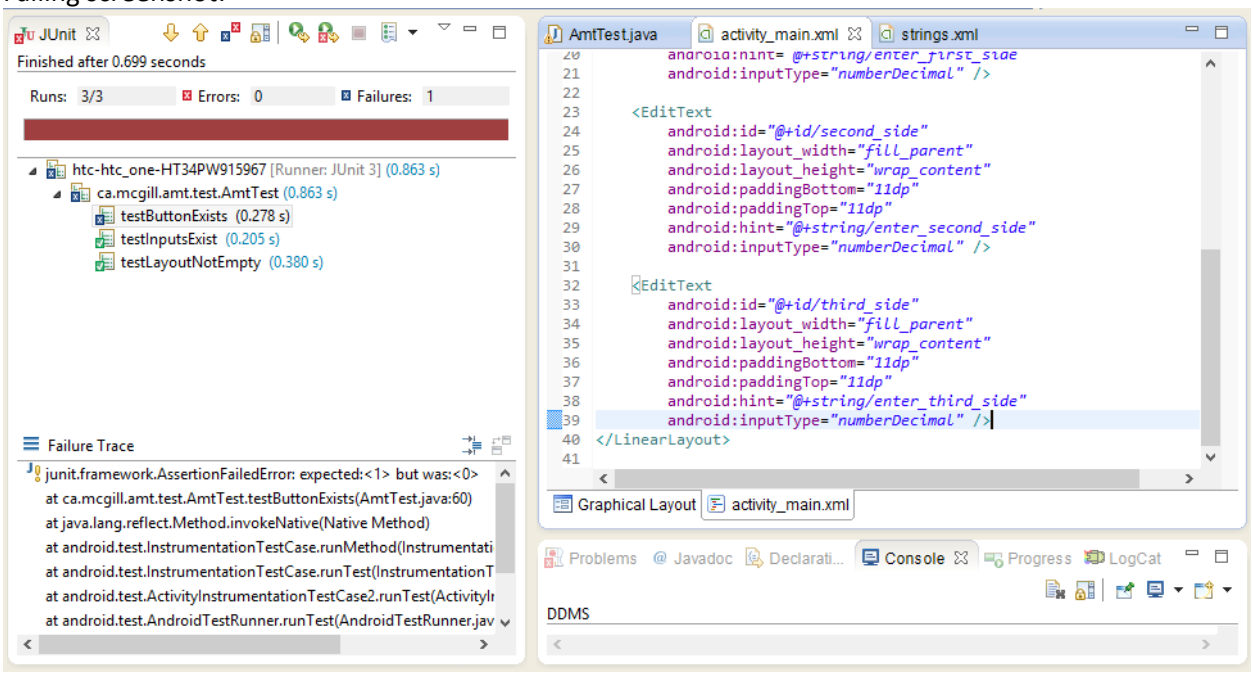


3. Submit button exists

Setup: None

Expected result: The activity should display a button that a user presses to obtain a result for their input

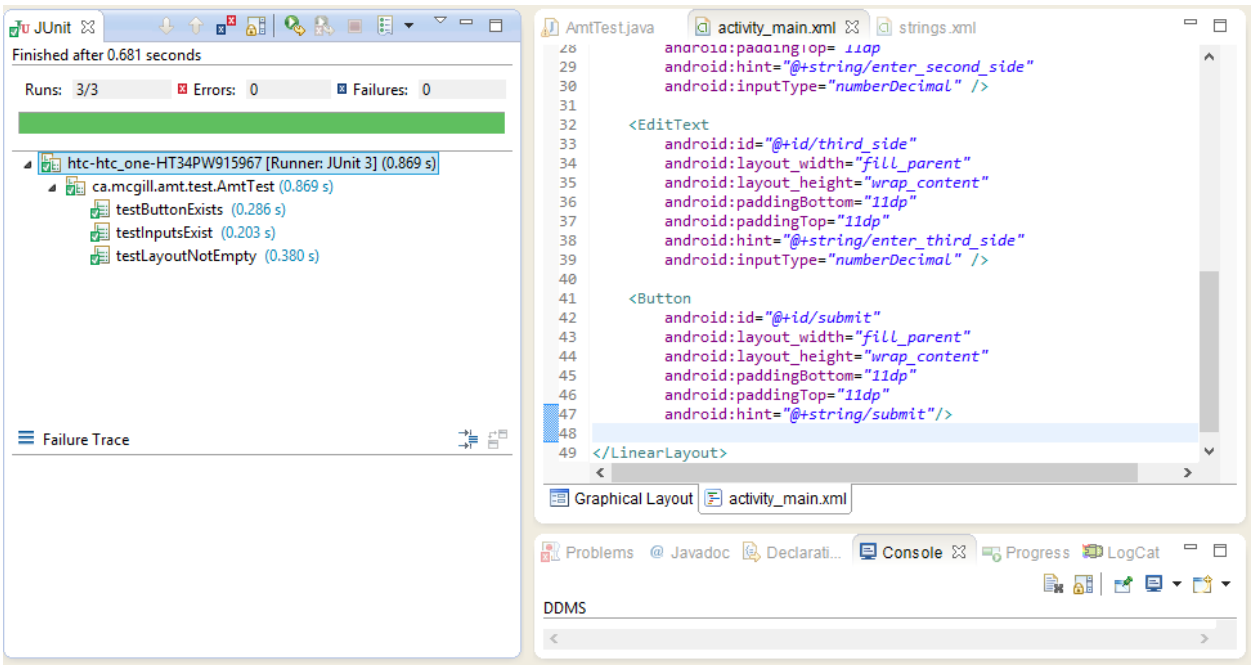
Failing screenshot:



Code added:

```
<Button
  android:id="@+id/submit"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content"
  android:paddingBottom="11dp"
  android:paddingTop="11dp"
  android:hint="@+string/submit"
        android:onClick="getTriangle"/>
```

Passing screenshot:

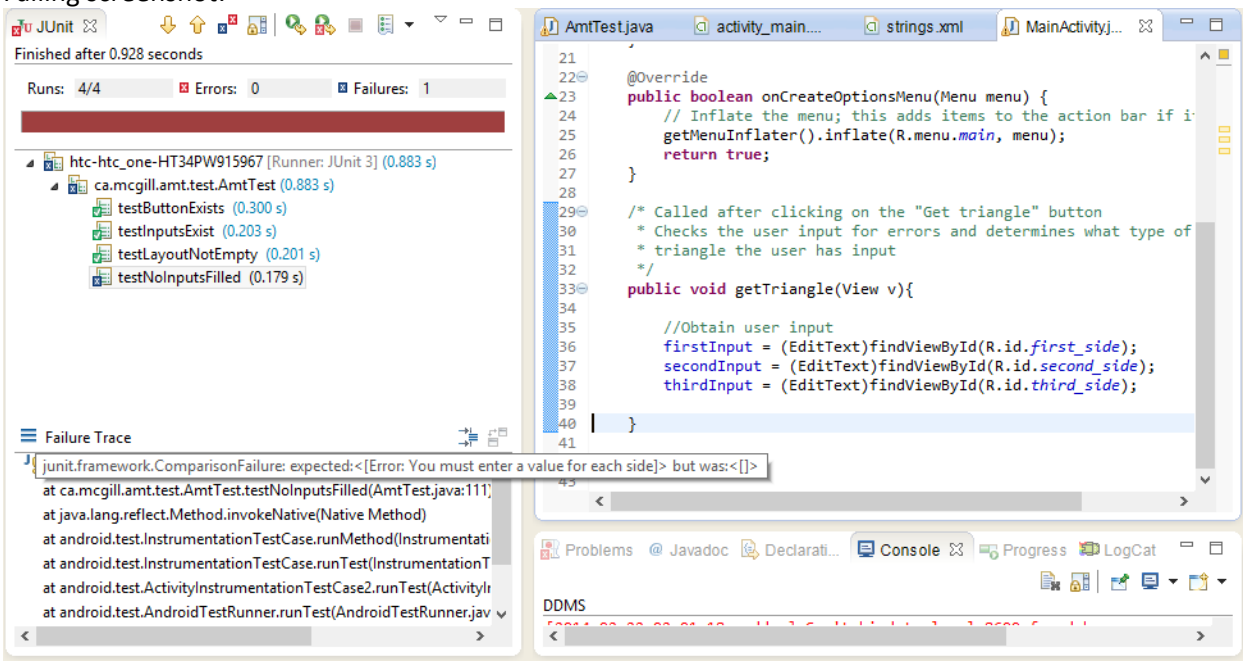

JUnit
Finished after 0.681 seconds
Runs: 3/3
Errors: 0
Failures: 0
htc-htc_one-HT34PW915967 [Runner: JUnit 3] (0.869 s)
ca.mcgill.amt.test.AmtTest (0.869 s)
testButtonExists (0.286 s)
testInputsExist (0.203 s)
testLayoutNotEmpty (0.380 s)
Failure Trace
AmtTest.java
activity_main.xml
strings.xml
android:hint="@+string/enter_second_side"
android:inputType="numberDecimal" />
<EditText
android:id="@+id/third_side"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:paddingBottom="11dp"
android:paddingTop="11dp"
android:hint="@+string/enter_third_side"
android:inputType="numberDecimal" />
<Button
android:id="@+id/submit"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:paddingBottom="11dp"
android:paddingTop="11dp"
android:hint="@+string/submit"/>
</LinearLayout>
Graphical Layout
activity_main.xml
Problems
Javadoc
Declarati...
Console
Progress
LogCat
DDMS

User Error Tests

1. None of the inputs are filled

Setup: Each input is left blank, then the submit button is pressed
Expected result: An error message appears telling the user that all inputs must be filled

Failing screenshot:

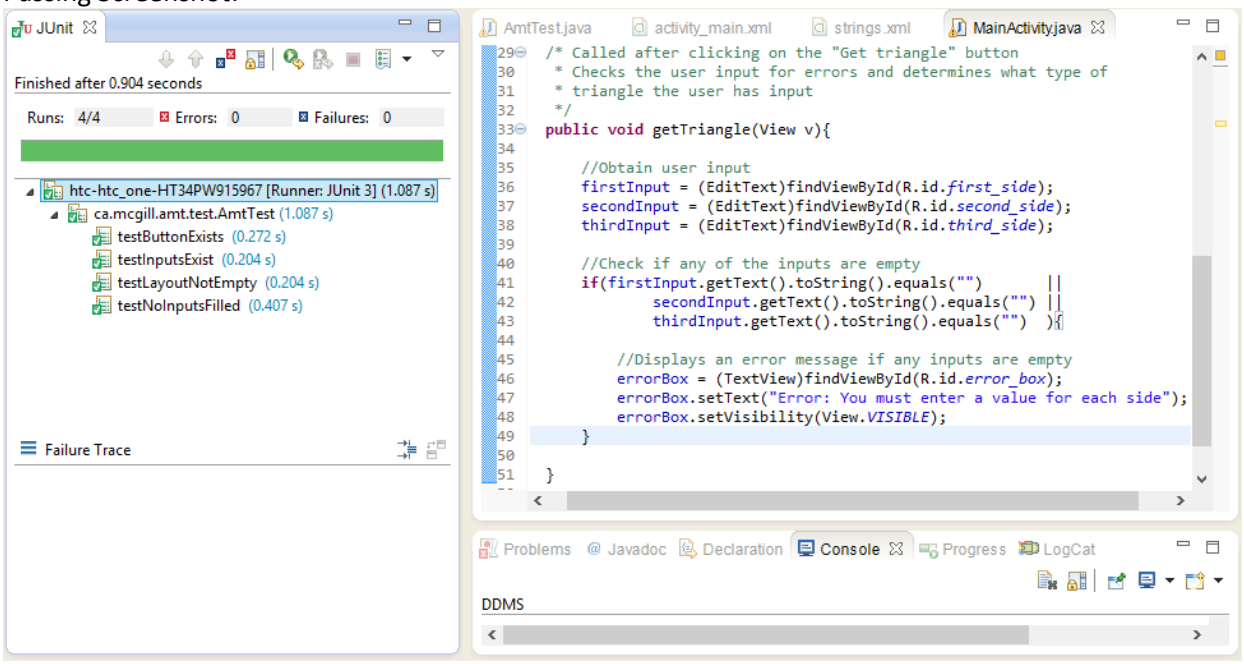


Code added:

```
//Check if any of the inputs are empty
      if(firstInput.getText().toString().equals("")          ||
                  secondInput.getText().toString().equals("")      ||
                  thirdInput.getText().toString().equals("")       ){

            //Displays an error message if any inputs are empty
                  outputBox.setText("Error: You must enter a value for
each side");
            outputBox.setVisibility(View.VISIBLE);
            return;
      }
```

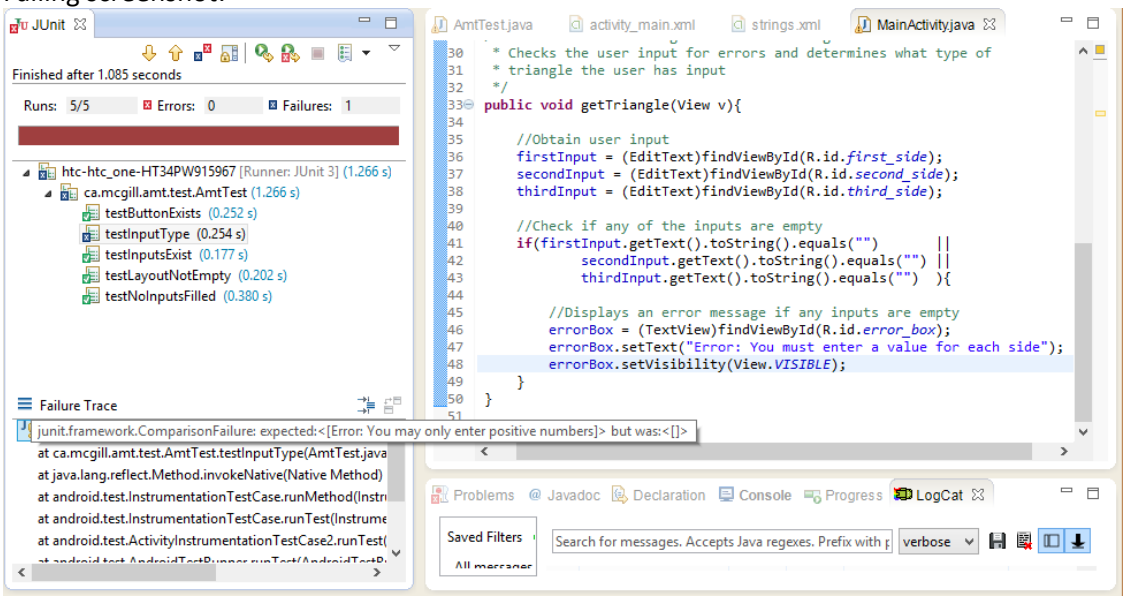
Passing Screenshot:



2. The input type is invalid

Verify that the user can only enter whole numbers
Setup: Input characters and negative numbers into the text fields
Expected Result: An error message appears telling the user they must only enter whole numbers

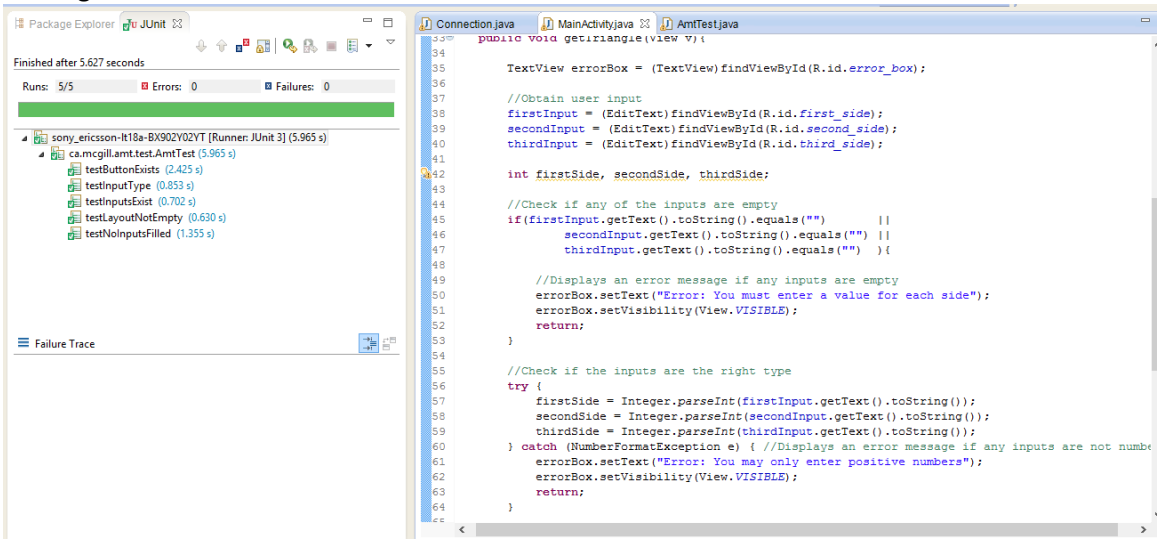
Failing screenshot:



Code added:

```
try {
            firstSide = Integer.parseInt(firstInput.getText().toString());
            secondSide = Integer.parseInt(secondInput.getText().toString());
            thirdSide = Integer.parseInt(thirdInput.getText().toString());
} catch (NumberFormatException e) {
                //Displays an error message if any inputs are not numbers
            outputBox.setText("Error: You may only enter positive numbers");
            outputBox.setVisibility(View.VISIBLE);
}
```

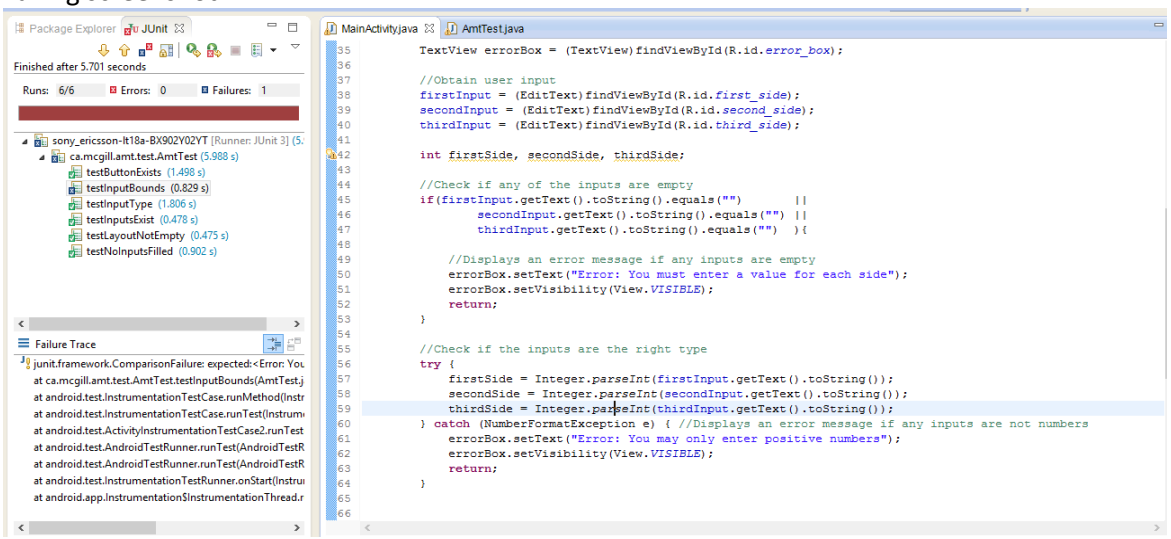
Passing screenshot:



3. The input is out of bounds

The user must input numbers between 1 and 100 inclusive
Setup: Set values to numbers outside of the valid range
Expected Result: A message appears telling the user that they must only enter numbers between 1 and 100 inclusive

Failing screenshot:

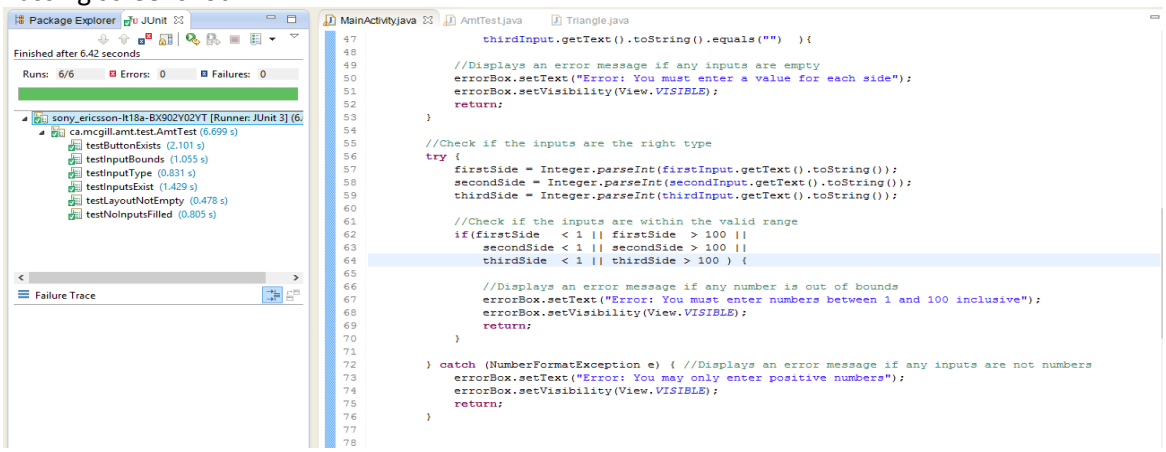


Code added:

```
//Check if the inputs are within the valid range
      if(firstSide   < 1 || firstSide  > 100 ||
            secondSide < 1 || secondSide > 100 ||
            thirdSide  < 1 || thirdSide > 100 ) {

            //Displays an error message if any number is out of bounds
            outputBox.setText("Error: You must enter numbers between 1 and
100 inclusive");
            outputBox.setVisibility(View.VISIBLE);
      }
```

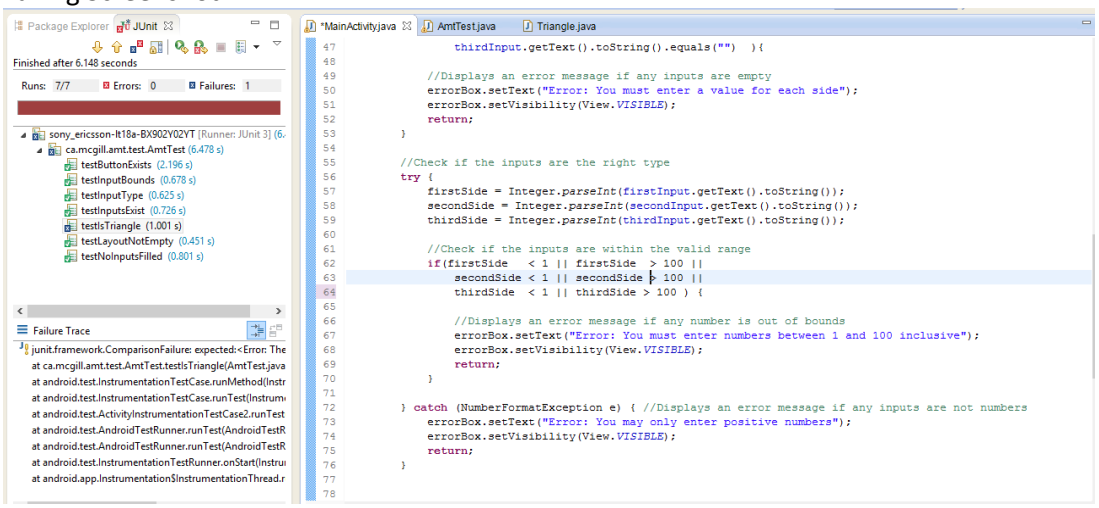
Passing screenshot:



4. The input does not create a triangle

The values of the sides the user entered does not create a triangle
Setup: Set two of the inputs such that their sum is smaller than the third input
Expected Result: An error message will appear telling the user that their inputs do not create a triangle

Failing Screenshot:

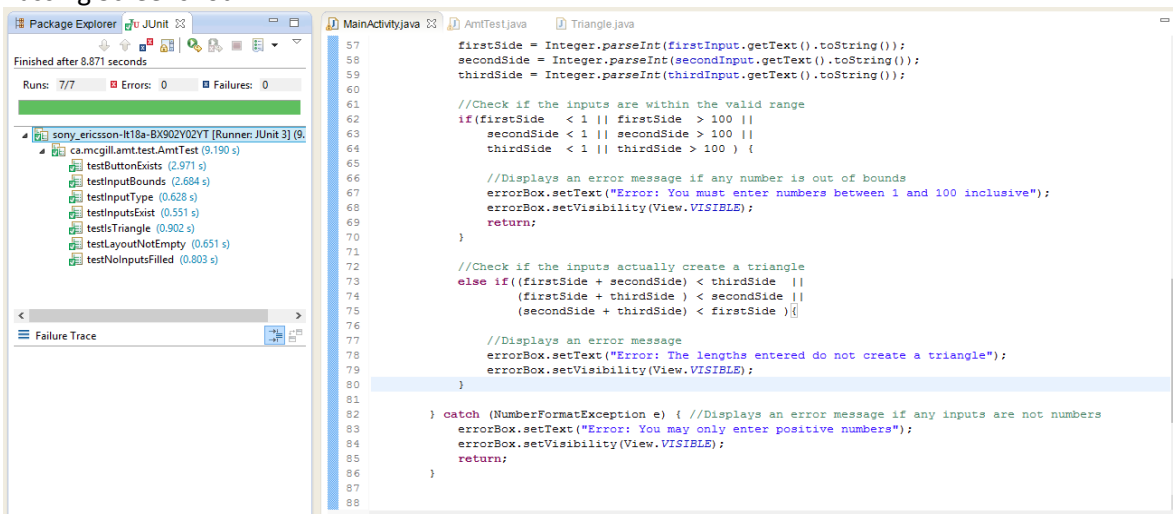


Code added:

```
//Check if the inputs actually create a triangle
      else if((firstSide + secondSide) < thirdSide  ||
            (firstSide + thirdSide ) < secondSide ||
            (secondSide + thirdSide) < firstSide ){

                  //Displays an error message
            outputBox.setText("Error: The lengths entered do not create a
triangle");
            outputBox.setVisibility(View.VISIBLE);
      }
```

Passing Screenshot:

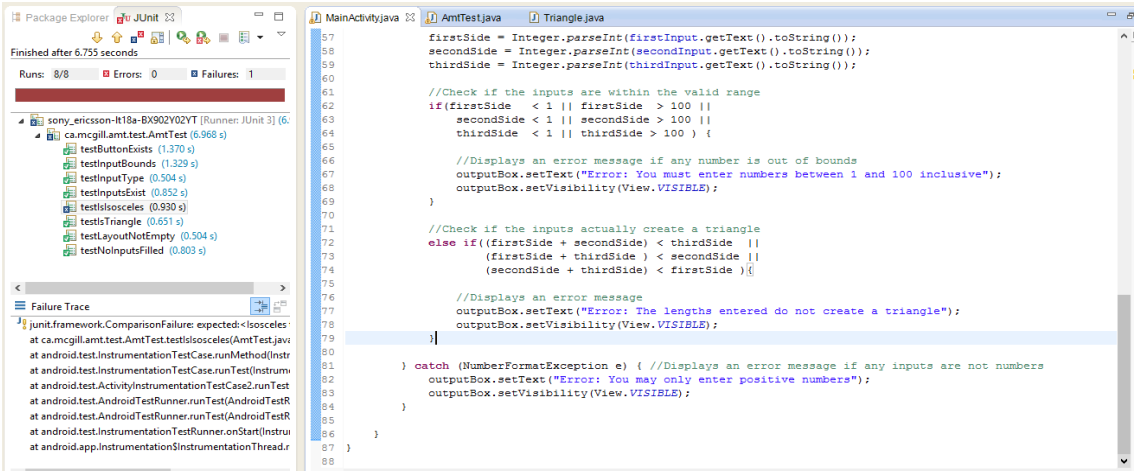


Triangle Business Rules

1. Isosceles

The user entered the same value for 2 sides.
Set up: Set 2 inputs to have the same value.
Expected result: A message will appear telling the user that the inputs form an isosceles triangle.

Failing screenshot:

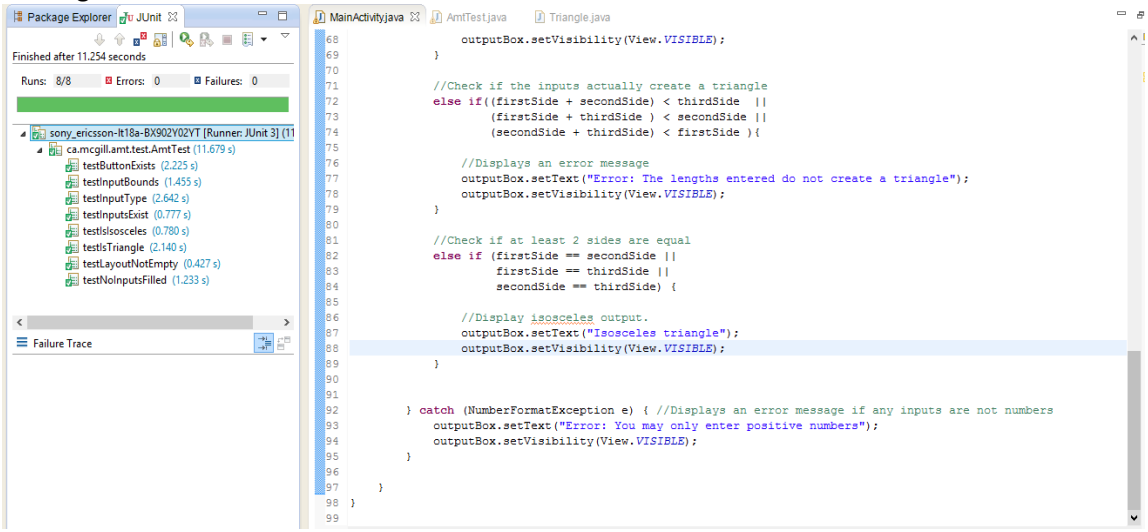


Code added:

```
else if (firstSide == secondSide ||
            firstSide == thirdSide ||
            secondSide == thirdSide) {

            //Display output isosceles triangle
            outputBox.setText("Isosceles triangle");
            outputBox.setVisibility(View.VISIBLE);
        }
```

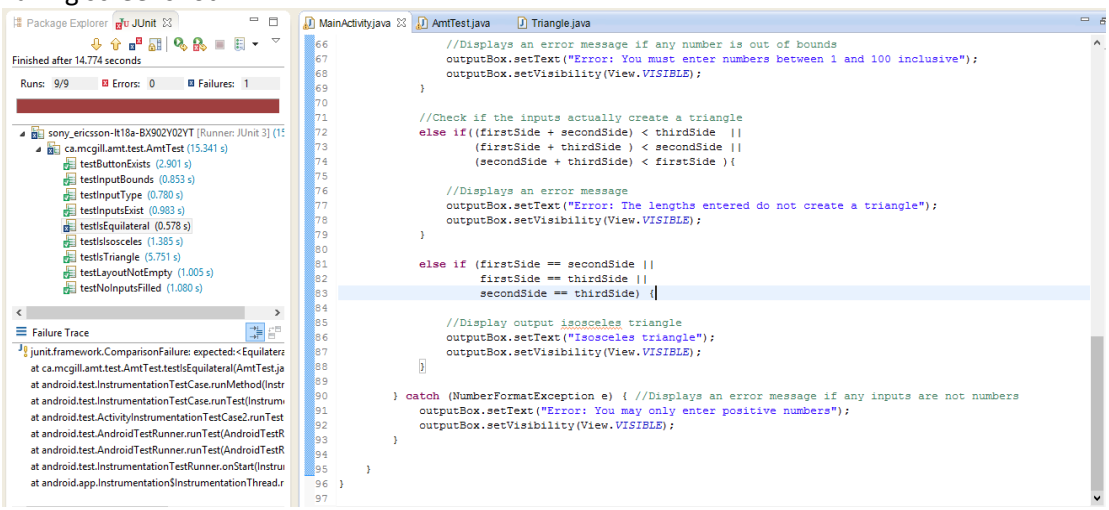
Passing screenshot:



2. Equilateral

The user entered the same value for all three sides.
Set up: Set all three inputs to have the same value.
Expected result: A message will appear telling the user that the inputs form an equilateral triangle.

Failing screenshot:

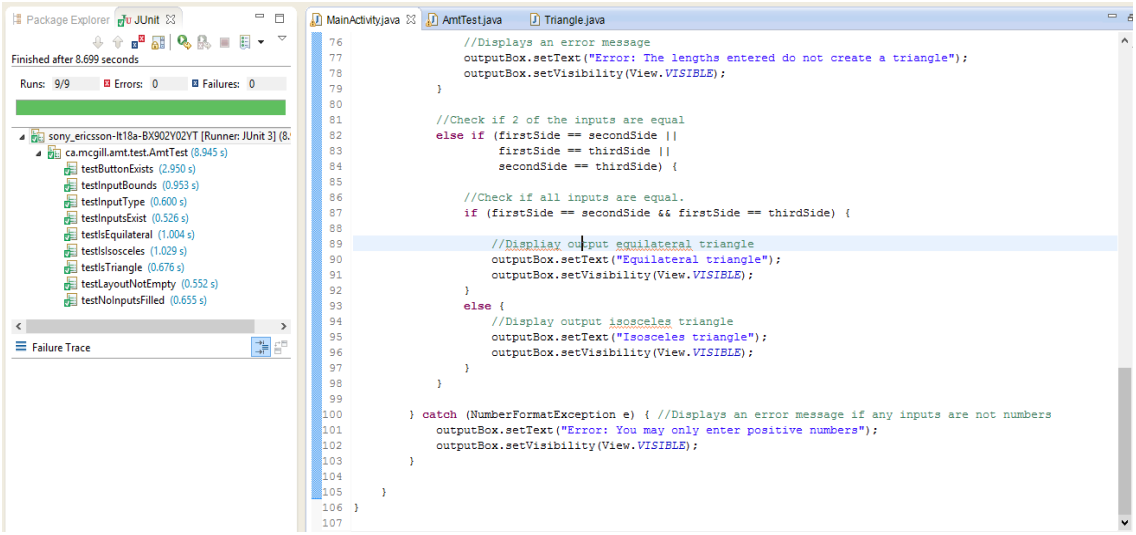


Code added:

```
//Check if all inputs are equal.
      if (firstSide == secondSide && firstSide == thirdSide) {

            //Display output equilateral triangle
            outputBox.setText("Equilateral triangle");
            outputBox.setVisibility(View.VISIBLE);
      }
      else {
```

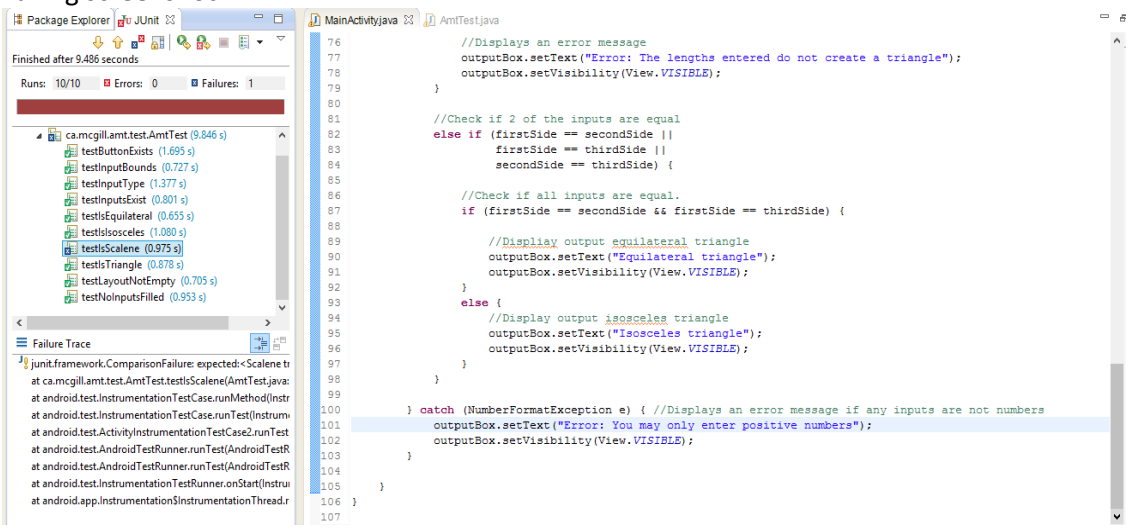
Passing screenshot:



3. Scalene

The user entered 3 different values for 3 sides.
Set up: Set 3 inputs to have different values.
Expected result: A message will appear telling the user that the inputs form a scalene triangle.

Failing screenshot:



Code added:

```
else {
                //Display output scalene triangle
                outputBox.setText("Scalene triangle");
                outputBox.setVisibility(View.VISIBLE);
                }
```

Passing screenshot: