Karim El-Baba
260582332

ECSE 428 – Software Engineering Practice
Assignment B

Canada Post Rate Calculator

The Application:

A command line application that calculates postage rates for a parcel based on inputs from the user regarding the parcel's: weight, height, length, width, Type of postage (Regular, Xpress, or Priority), and the destination postal code. The postage rate is not calculated based on real values.

How To Run:

1. On the command line, navigate to the directory where the runnable jar file is.
2. Ensure that the jar file is in the same directory as the csv file.
3. Run: java -jar AssignmentB.jar

Assumptions:

- The origin postal code will always start with an H (the parcel is sent from Montreal).
- Postal code inputs will be limited to those within Canada.
- Destination postal codes starting with H represent Montreal, G or J to represent the remainder of Quebec. Other letters represent the rest of Canada.
- The user inputs are in the form of Strings obtained from the console.

How Rates are Calculated:

- The CSV file provided contains 6 sets of sub-rates, for each of the input parameters.
- The final rate is calculated as the sum of 1 sub rate from each of the set.
- The sub rates are selected from each set based on the user input and the range it falls into (with respect to the type of attribute).
- The ranges for each attribute are specified in the CSV file and are parsed in the code.

Test 1: getDimensionSubRateTest

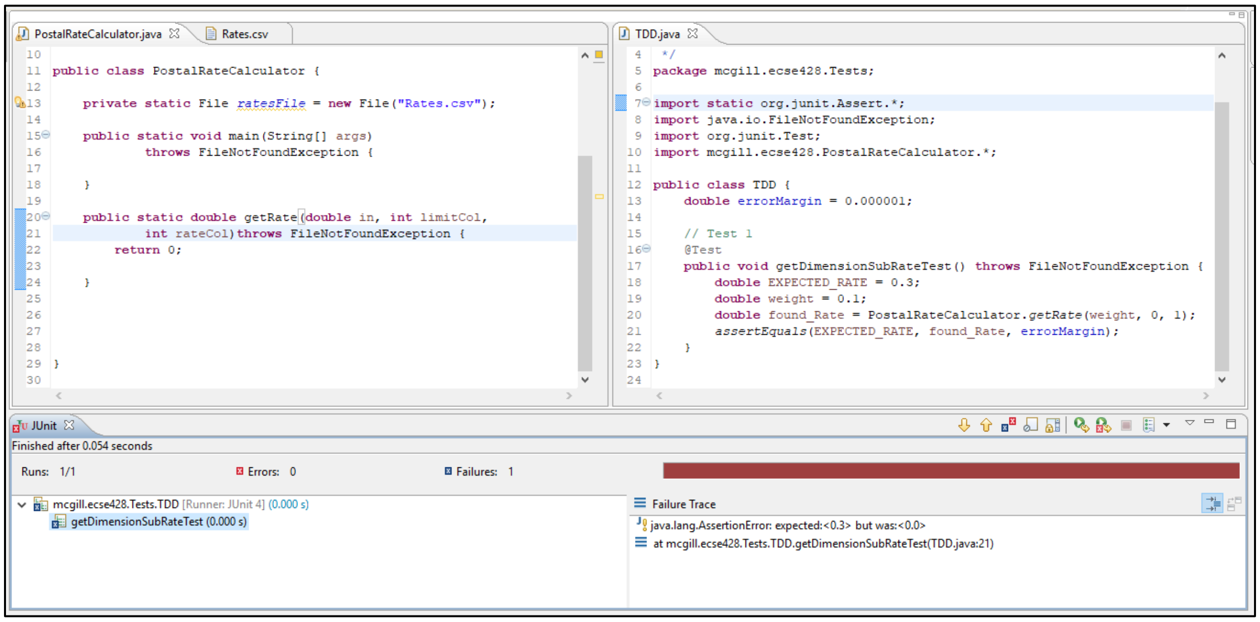
Purpose:

- Test the ability to parse rates from sub tables associated with weight, height, length and width.

Inputs and Expected Output:

- The value (0.1Kg) of the attribute being tested.
- Values for the columns with the ranges, and their respective rates.
- The output is the rate (0.3$) associated with the range of the attribute.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

```java

public class PostalRateCalculator {

    private static File ratesFile = new File("Rates.csv");

    public static void main(String[] args)
            throws FileNotFoundException {

    }

    public static double getRate(double in, int limitCol,
            int rateCol) throws FileNotFoundException {
        return 0;

    }




}

```

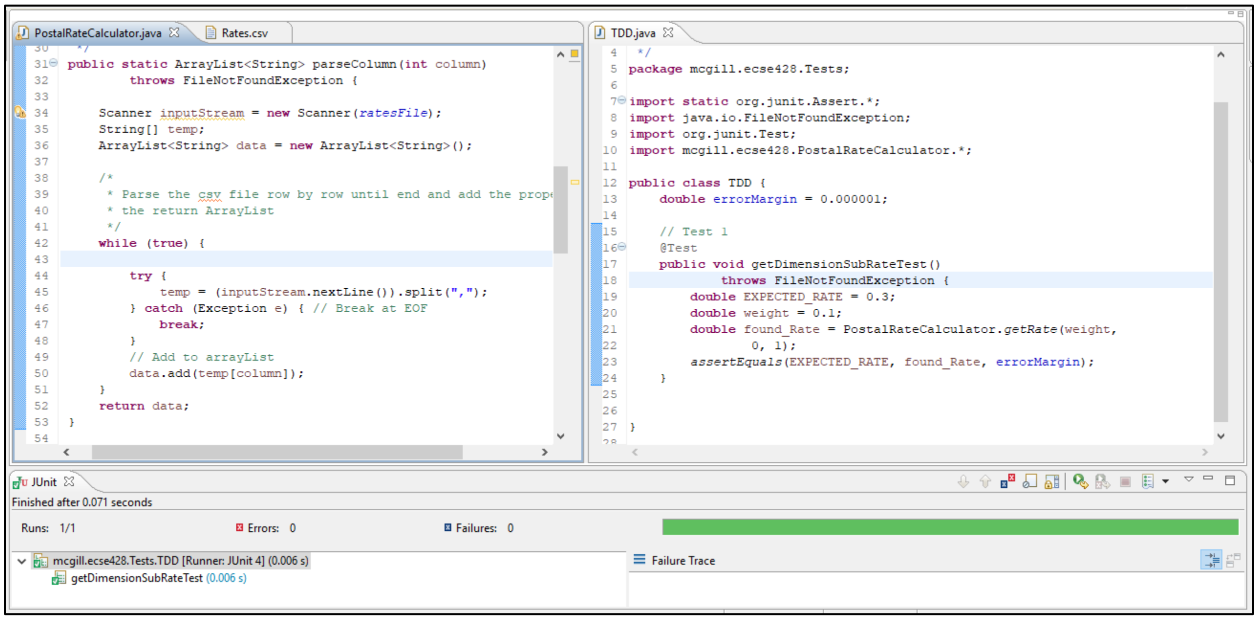
TDD.java

```java
 */
package mcgill.ecse428.Tests;

import static org.junit.Assert.*;
import java.io.FileNotFoundException;
import org.junit.Test;
import mcgill.ecse428.PostalRateCalculator.*;

public class TDD {
    double errorMargin = 0.000001;

    // Test 1
    @Test
    public void getDimensionSubRateTest() throws FileNotFoundException {
        double EXPECTED_RATE = 0.3;
        double weight = 0.1;
        double found_Rate = PostalRateCalculator.getRate(weight, 0, 1);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
}

```

JUnit

```
Finished after 0.054 seconds

Runs: 1/1        Errors: 0        Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.000 s)
    getDimensionSubRateTest (0.000 s)

Failure Trace
java.lang.AssertionError: expected:<0.3> but was:<0.0>
at mcgill.ecse428.Tests.TDD.getDimensionSubRateTest(TDD.java:21)
```

Passing Screenshot:


PostalRateCalculator.java | Rates.csv

```java
    public static ArrayList<String> parseColumn(int column)
            throws FileNotFoundException {

        Scanner inputStream = new Scanner(ratesFile);
        String[] temp;
        ArrayList<String> data = new ArrayList<String>();

        /*
         * Parse the csv file row by row until end and add the prope
         * the return ArrayList
         */
        while (true) {

            try {
                temp = (inputStream.nextLine()).split(",");
            } catch (Exception e) { // Break at EOF
                break;
            }
            // Add to arrayList
            data.add(temp[column]);
        }
        return data;
    }
```

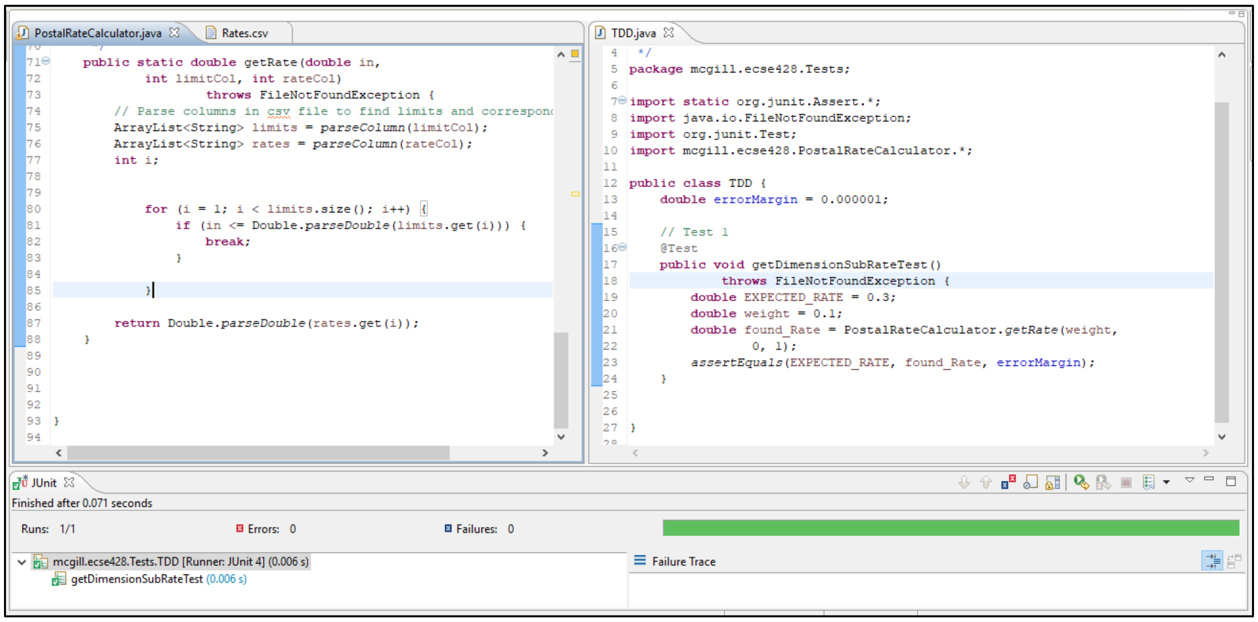
TDD.java

```java
 */
package mcgill.ecse428.Tests;

import static org.junit.Assert.*;
import java.io.FileNotFoundException;
import org.junit.Test;
import mcgill.ecse428.PostalRateCalculator.*;

public class TDD {
    double errorMargin = 0.000001;

    // Test 1
    @Test
    public void getDimensionSubRateTest()
            throws FileNotFoundException {
        double EXPECTED_RATE = 0.3;
        double weight = 0.1;
        double found_Rate = PostalRateCalculator.getRate(weight,
                0, 1);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }


}
```

JUnit

```
Finished after 0.071 seconds
Runs: 1/1        Errors: 0        Failures: 0
mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.006 s)
    getDimensionSubRateTest (0.006 s)
```

Failure Trace


PostalRateCalculator.java | Rates.csv

```java
    public static double getRate(double in,
            int limitCol, int rateCol)
                throws FileNotFoundException {
        // Parse columns in csv file to find limits and correspond
        ArrayList<String> limits = parseColumn(limitCol);
        ArrayList<String> rates = parseColumn(rateCol);
        int i;


            for (i = 1; i < limits.size(); i++) {
                if (in <= Double.parseDouble(limits.get(i))) {
                    break;
                }

            }

        return Double.parseDouble(rates.get(i));
    }





}
```

TDD.java

```java
 */
package mcgill.ecse428.Tests;

import static org.junit.Assert.*;
import java.io.FileNotFoundException;
import org.junit.Test;
import mcgill.ecse428.PostalRateCalculator.*;

public class TDD {
    double errorMargin = 0.000001;

    // Test 1
    @Test
    public void getDimensionSubRateTest()
            throws FileNotFoundException {
        double EXPECTED_RATE = 0.3;
        double weight = 0.1;
        double found_Rate = PostalRateCalculator.getRate(weight,
                0, 1);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }


}
```

JUnit

```
Finished after 0.071 seconds
Runs: 1/1        Errors: 0        Failures: 0
mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.006 s)
    getDimensionSubRateTest (0.006 s)
```

Failure Trace

Test 2: typeRate

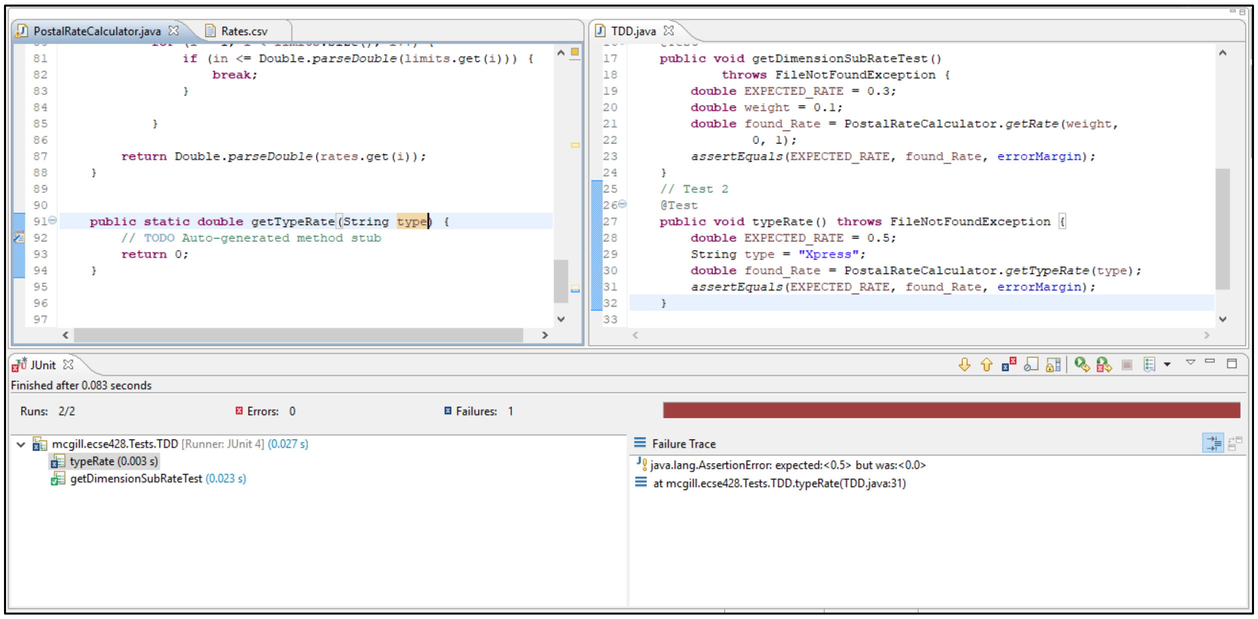
Purpose:

- Test the ability to parse the sub rates associated with the type of postage.

Inputs and Expected Output:

- The type of postage (Xpress).
- The output is the respective rate (0.5$) associated with the postage type.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

```
                    if (in <= Double.parseDouble(limits.get(i))) {
                        break;
                    }

                }

            return Double.parseDouble(rates.get(i));
        }


    public static double getTypeRate(String type) {
        // TODO Auto-generated method stub
        return 0;
    }
```

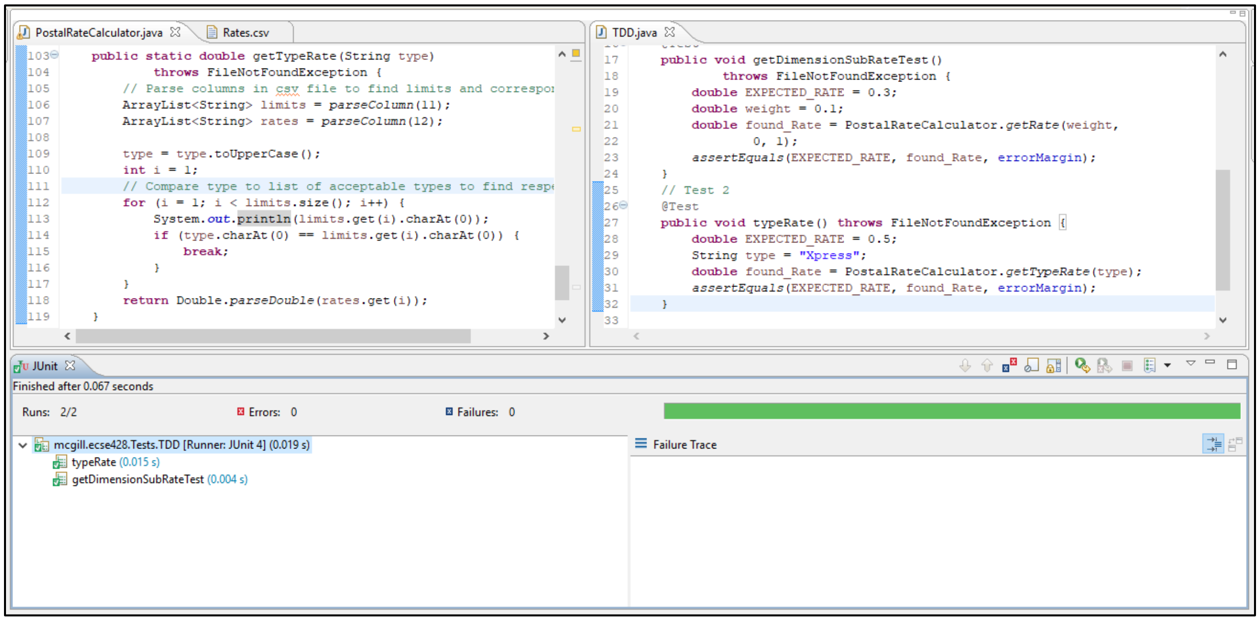
TDD.java

```
    public void getDimensionSubRateTest()
            throws FileNotFoundException {
        double EXPECTED_RATE = 0.3;
        double weight = 0.1;
        double found_Rate = PostalRateCalculator.getRate(weight,
                0, 1);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 2
    @Test
    public void typeRate() throws FileNotFoundException {
        double EXPECTED_RATE = 0.5;
        String type = "Xpress";
        double found_Rate = PostalRateCalculator.getTypeRate(type);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
```

JUnit

```
Finished after 0.083 seconds

Runs: 2/2        Errors: 0        Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.027 s)
    typeRate (0.003 s)
    getDimensionSubRateTest (0.023 s)

Failure Trace
java.lang.AssertionError: expected:<0.5> but was:<0.0>
at mcgill.ecse428.Tests.TDD.typeRate(TDD.java:31)
```

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
    public static double getTypeRate(String type)
            throws FileNotFoundException {
        // Parse columns in csv file to find limits and correspo
        ArrayList<String> limits = parseColumn(11);
        ArrayList<String> rates = parseColumn(12);

        type = type.toUpperCase();
        int i = 1;
        // Compare type to list of acceptable types to find respe
        for (i = 1; i < limits.size(); i++) {
            System.out.println(limits.get(i).charAt(0));
            if (type.charAt(0) == limits.get(i).charAt(0)) {
                break;
            }
        }
        return Double.parseDouble(rates.get(i));
    }
```

TDD.java

```
    public void getDimensionSubRateTest()
            throws FileNotFoundException {
        double EXPECTED_RATE = 0.3;
        double weight = 0.1;
        double found_Rate = PostalRateCalculator.getRate(weight,
                0, 1);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 2
    @Test
    public void typeRate() throws FileNotFoundException {
        double EXPECTED_RATE = 0.5;
        String type = "Xpress";
        double found_Rate = PostalRateCalculator.getTypeRate(type);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
```

JUnit

```
Finished after 0.067 seconds

Runs: 2/2        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.019 s)
    typeRate (0.015 s)
    getDimensionSubRateTest (0.004 s)

Failure Trace
```

Test 3: toPostalCodeRate

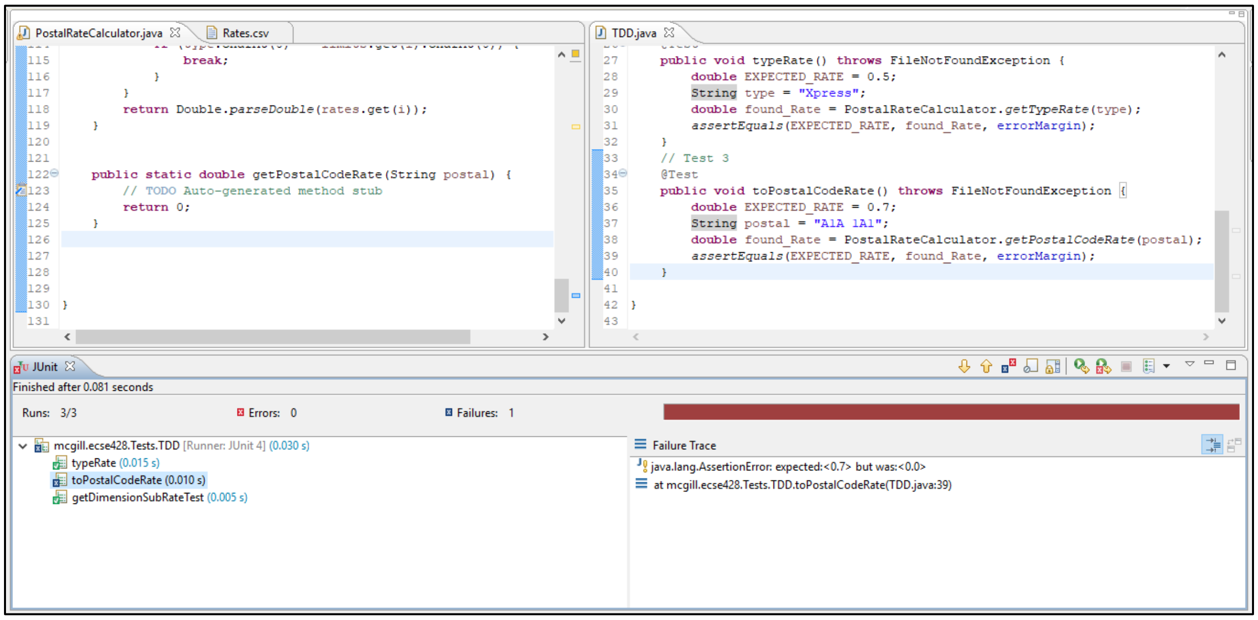
Purpose:

- Test the ability to parse the sub rates associated with the destination postal code of the parcel.

Inputs and Expected Output:

- The destination postal code (A1A 1A1).
- The output is the rate (0.7$) associated with the input postal code.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

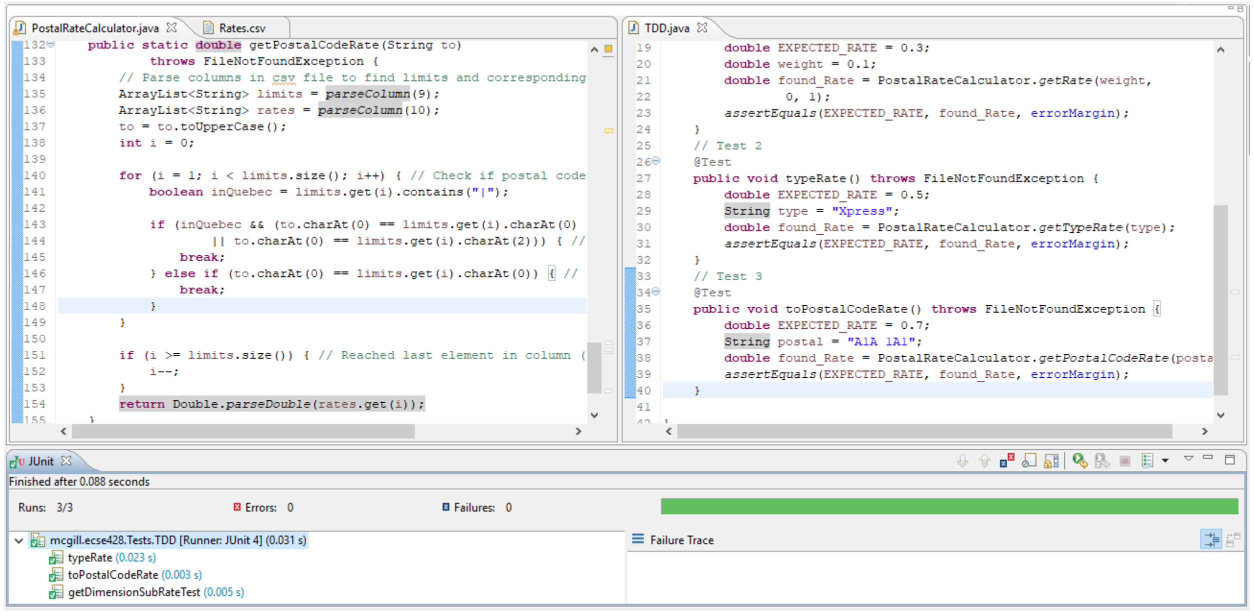
```java
                break;
            }
        }
        return Double.parseDouble(rates.get(i));
    }


    public static double getPostalCodeRate(String postal) {
        // TODO Auto-generated method stub
        return 0;
    }




}
```

TDD.java

```java
    public void typeRate() throws FileNotFoundException {
        double EXPECTED_RATE = 0.5;
        String type = "Xpress";
        double found_Rate = PostalRateCalculator.getTypeRate(type);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 3
    @Test
    public void toPostalCodeRate() throws FileNotFoundException {
        double EXPECTED_RATE = 0.7;
        String postal = "A1A 1A1";
        double found_Rate = PostalRateCalculator.getPostalCodeRate(postal);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

}
```

```
JUnit
Finished after 0.081 seconds
Runs: 3/3          Errors: 0          Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.030 s)
    typeRate (0.015 s)
    toPostalCodeRate (0.010 s)
    getDimensionSubRateTest (0.005 s)

Failure Trace
java.lang.AssertionError: expected:<0.7> but was:<0.0>
at mcgill.ecse428.Tests.TDD.toPostalCodeRate(TDD.java:39)
```

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```java
    public static double getPostalCodeRate(String to)
            throws FileNotFoundException {
        // Parse columns in csv file to find limits and corresponding
        ArrayList<String> limits = parseColumn(9);
        ArrayList<String> rates = parseColumn(10);
        to = to.toUpperCase();
        int i = 0;

        for (i = 1; i < limits.size(); i++) { // Check if postal code
            boolean inQuebec = limits.get(i).contains("|");

            if (inQuebec && (to.charAt(0) == limits.get(i).charAt(0)
                    || to.charAt(0) == limits.get(i).charAt(2))) { //
                break;
            } else if (to.charAt(0) == limits.get(i).charAt(0)) { //
                break;
            }
        }

        if (i >= limits.size()) { // Reached last element in column (
            i--;
        }
        return Double.parseDouble(rates.get(i));
```

TDD.java

```java
        double EXPECTED_RATE = 0.3;
        double weight = 0.1;
        double found_Rate = PostalRateCalculator.getRate(weight,
                0, 1);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 2
    @Test
    public void typeRate() throws FileNotFoundException {
        double EXPECTED_RATE = 0.5;
        String type = "Xpress";
        double found_Rate = PostalRateCalculator.getTypeRate(type);
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 3
    @Test
    public void toPostalCodeRate() throws FileNotFoundException {
        double EXPECTED_RATE = 0.7;
        String postal = "A1A 1A1";
        double found_Rate = PostalRateCalculator.getPostalCodeRate(posta
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

```

```
JUnit
Finished after 0.088 seconds
Runs: 3/3          Errors: 0          Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.031 s)
    typeRate (0.023 s)
    toPostalCodeRate (0.003 s)
    getDimensionSubRateTest (0.005 s)

Failure Trace
```

Test 4: baseFullRateTest

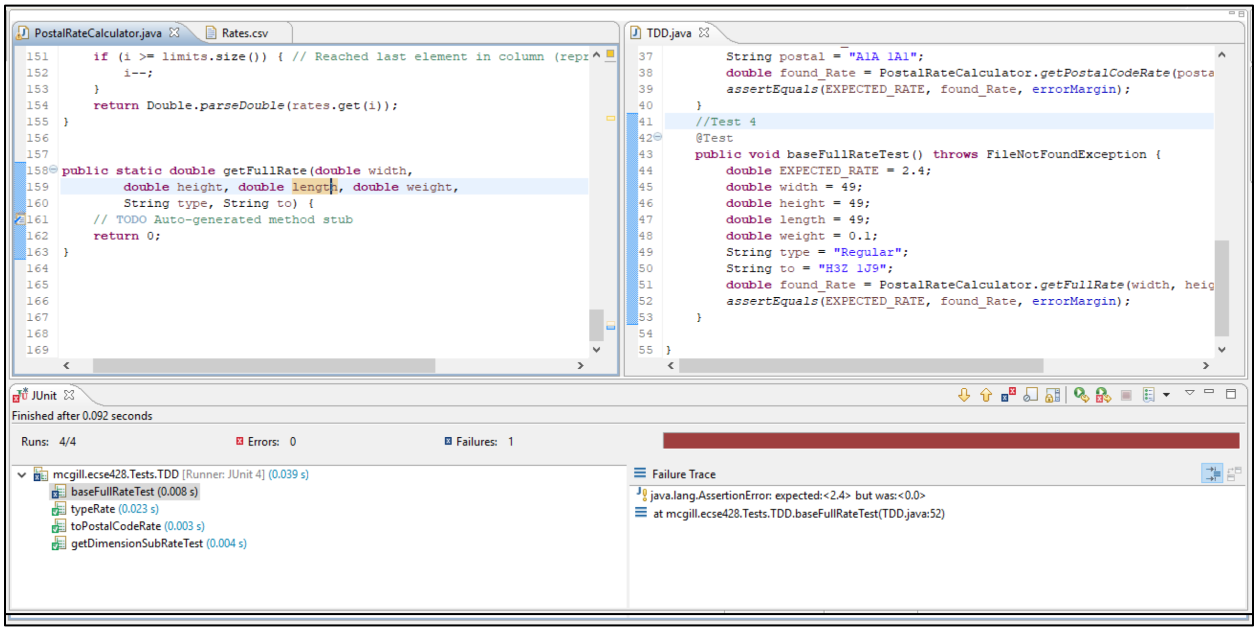
Purpose:

- Test the ability to get the full rate of a parcel based on the sub rate of all it's attributes.

Inputs and Expected Output:

- ***weight*** (0.1 Kg), ***height*** (49 cm), ***width*** (49 cm), ***length*** (49 cm), and the ***destination postal code*** (H3Z 1J9) within Montreal, and ***type of postage*** (Regular).
- The output is the proper parcel rate (2.4$).

Failing Screenshot:



PostalRateCalculator.java

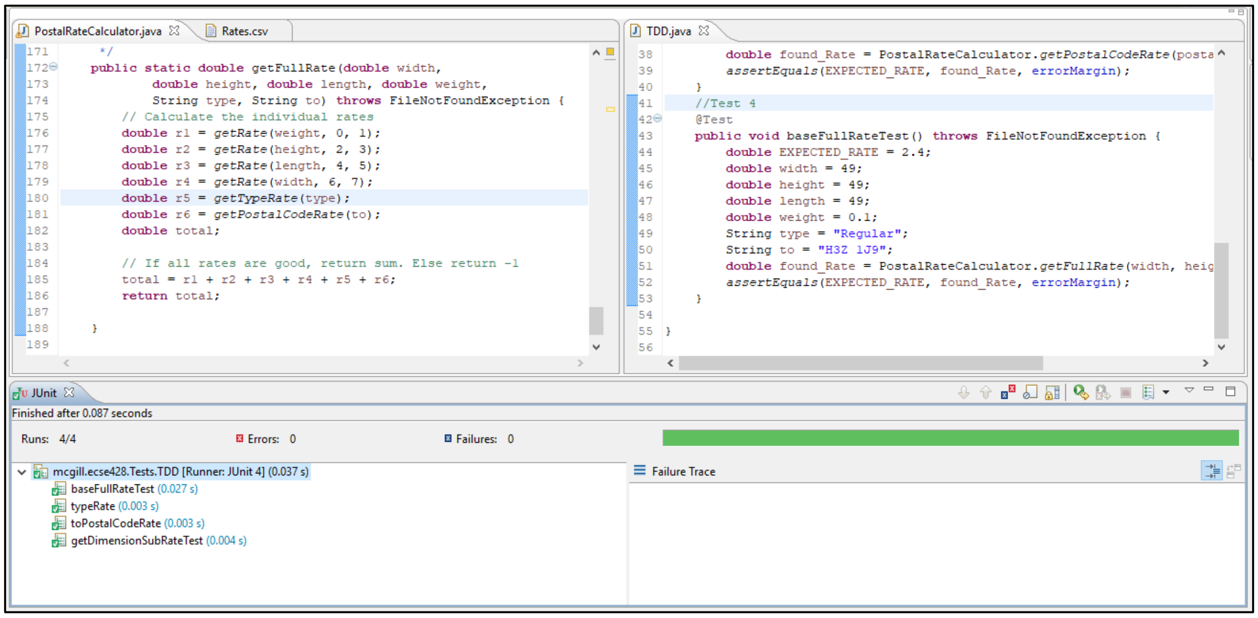
```
        if (i >= limits.size()) { // Reached last element in column (repr
            i--;
        }
        return Double.parseDouble(rates.get(i));
    }


public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) {
        // TODO Auto-generated method stub
        return 0;
    }
```

TDD.java

```
        String postal = "A1A 1A1";
        double found_Rate = PostalRateCalculator.getPostalCodeRate(posta
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    //Test 4
    @Test
    public void baseFullRateTest() throws FileNotFoundException {
        double EXPECTED_RATE = 2.4;
        double width = 49;
        double height = 49;
        double length = 49;
        double weight = 0.1;
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, heig
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

}
```

```
Finished after 0.092 seconds
Runs: 4/4    Errors: 0    Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.039 s)
    baseFullRateTest (0.008 s)
    typeRate (0.023 s)
    toPostalCodeRate (0.003 s)
    getDimensionSubRateTest (0.004 s)

Failure Trace
java.lang.AssertionError: expected:<2.4> but was:<0.0>
at mcgill.ecse428.Tests.TDD.baseFullRateTest(TDD.java:52)
```

Passing Screenshot:



PostalRateCalculator.java

```
     */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;

    }
```

TDD.java

```
        double found_Rate = PostalRateCalculator.getPostalCodeRate(posta
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    //Test 4
    @Test
    public void baseFullRateTest() throws FileNotFoundException {
        double EXPECTED_RATE = 2.4;
        double width = 49;
        double height = 49;
        double length = 49;
        double weight = 0.1;
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, heig
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

}
```

```
Finished after 0.087 seconds
Runs: 4/4    Errors: 0    Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.037 s)
    baseFullRateTest (0.027 s)
    typeRate (0.003 s)
    toPostalCodeRate (0.003 s)
    getDimensionSubRateTest (0.004 s)

Failure Trace
```

Test 5: fullRateWeightChange

Purpose:

- Test the ability to get the full rate of a parcel with the ***weight*** varied.

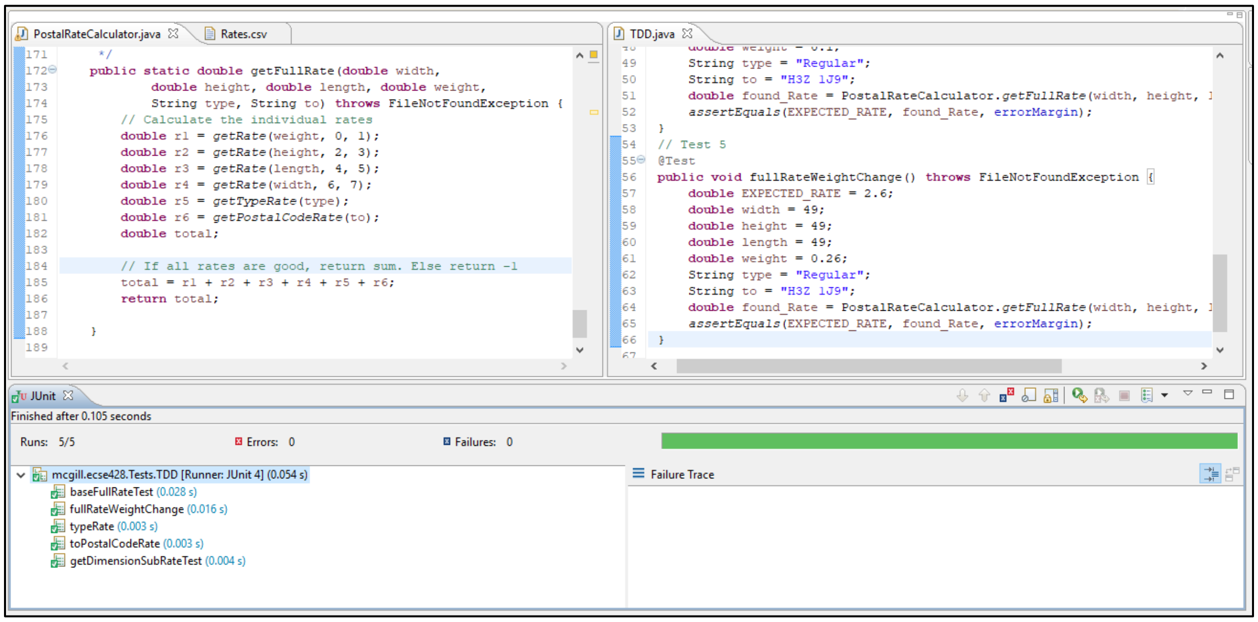
Inputs and Expected Output:

- ***weight (0.26 Kg), height*** (49 cm), ***width*** (49 cm), ***length*** (49 cm), and ***destination postal code*** (H3Z 1J9), and ***type of postage*** (Regular).
- The output is the proper parcel rate (2.6$).

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
     */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;

    }
```

TDD.java

```
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, height,
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 5
    @Test
    public void fullRateWeightChange() throws FileNotFoundException {
        double EXPECTED_RATE = 2.6;
        double width = 49;
        double height = 49;
        double length = 49;
        double weight = 0.26;
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, height,
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
```

```
JUnit
Finished after 0.105 seconds

Runs: 5/5        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.054 s)
    baseFullRateTest (0.028 s)
    fullRateWeightChange (0.016 s)
    typeRate (0.003 s)
    toPostalCodeRate (0.003 s)
    getDimensionSubRateTest (0.004 s)

Failure Trace
```

Test 6: fullRateHeightChange

Purpose:

- Test the ability to get the full rate of a parcel with the ***height*** varied.

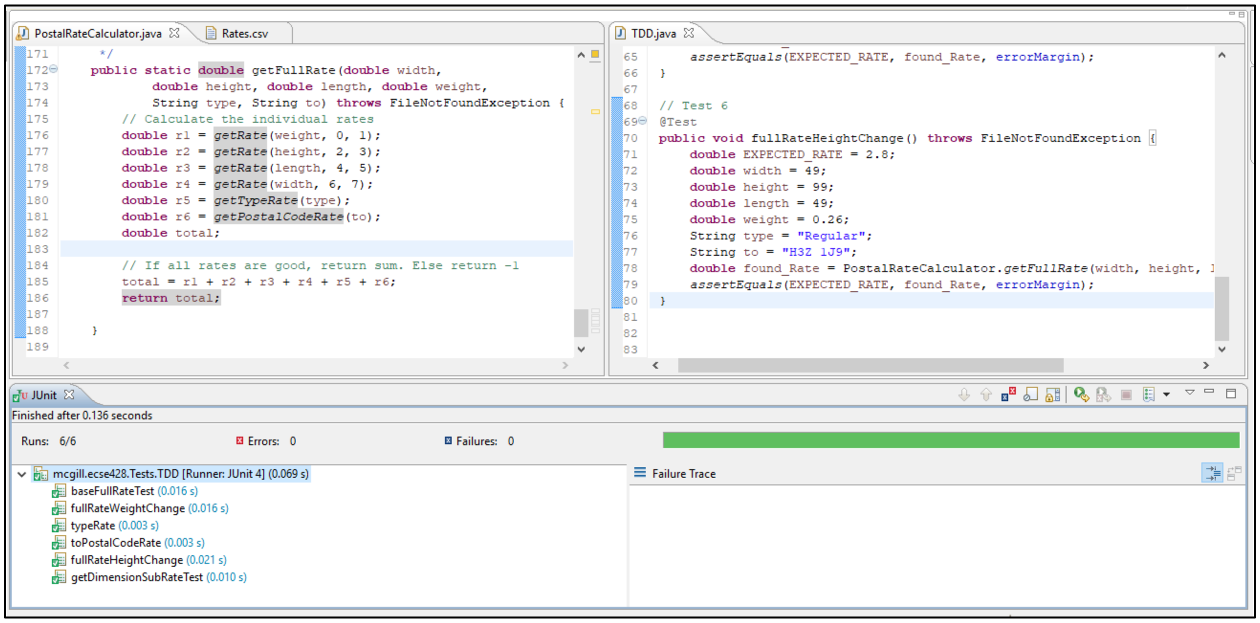
Inputs and Expected Output:

- ***weight* (*0.26 Kg*), *height* (*99 cm*), *width*** (49 cm), ***length*** (49 cm), and the ***destination postal code*** (H3Z 1J9), and ***type of postage*** (Regular).
- The output is the proper parcel rate (2.8$).

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
         */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;

    }

```

TDD.java

```
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

    // Test 6
    @Test
    public void fullRateHeightChange() throws FileNotFoundException {
        double EXPECTED_RATE = 2.8;
        double width = 49;
        double height = 99;
        double length = 49;
        double weight = 0.26;
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, height, 
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }



```

JUnit

```
Finished after 0.136 seconds

Runs: 6/6        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.069 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.016 s)
    typeRate (0.003 s)
    toPostalCodeRate (0.003 s)
    fullRateHeightChange (0.021 s)
    getDimensionSubRateTest (0.010 s)
```

Test 7: fullRateLengthChange

Purpose:

- Test the ability to get the full rate of a parcel with the ***length*** varied.

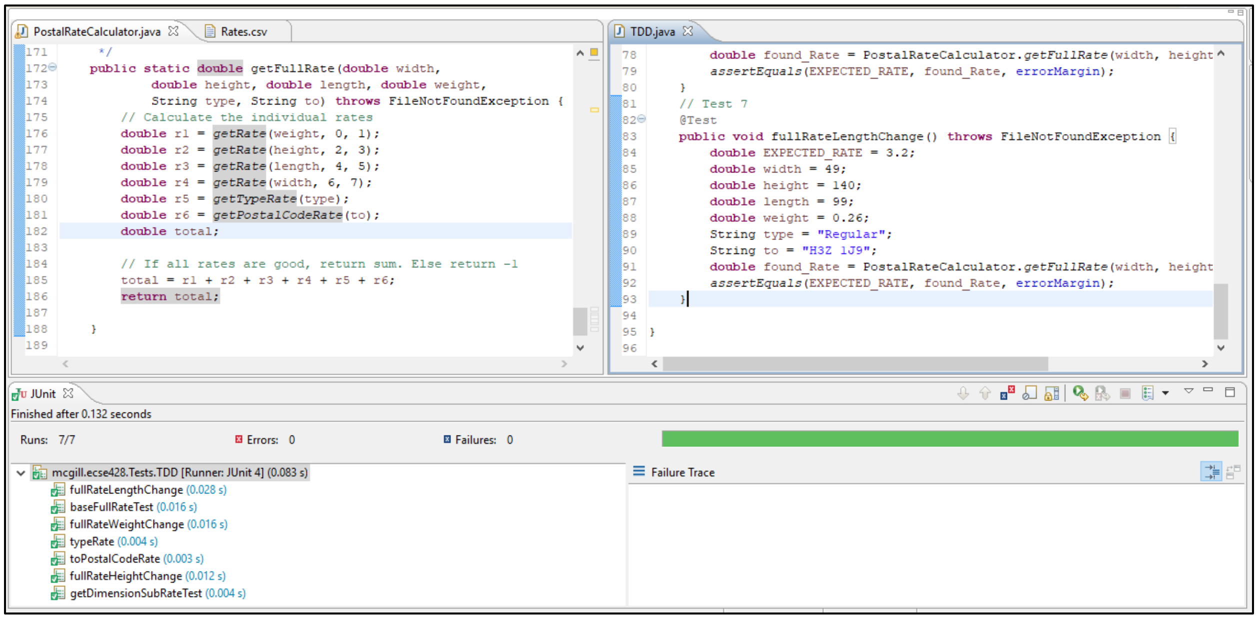
Inputs and Expected Output:

- ***weight (0.26 Kg), height (140 cm), width*** (49 cm), ***length (99 cm),*** and ***destination postal code*** (H3Z 1J9), and ***type of postage*** (Regular).
- The output is the proper parcel rate (sum of all sub rates).

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



```java
    */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;

    }
```

```java
        double found_Rate = PostalRateCalculator.getFullRate(width, height
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 7
    @Test
    public void fullRateLengthChange() throws FileNotFoundException {
        double EXPECTED_RATE = 3.2;
        double width = 49;
        double height = 140;
        double length = 99;
        double weight = 0.26;
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, height
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

}
```

```
Finished after 0.132 seconds
Runs: 7/7        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.083 s)
    fullRateLengthChange (0.028 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.016 s)
    typeRate (0.004 s)
    toPostalCodeRate (0.003 s)
    fullRateHeightChange (0.012 s)
    getDimensionSubRateTest (0.004 s)
```

Test 8: fullRateWidthChange

Purpose:

- Test the ability to get the full rate of a parcel with the ***width*** varied.

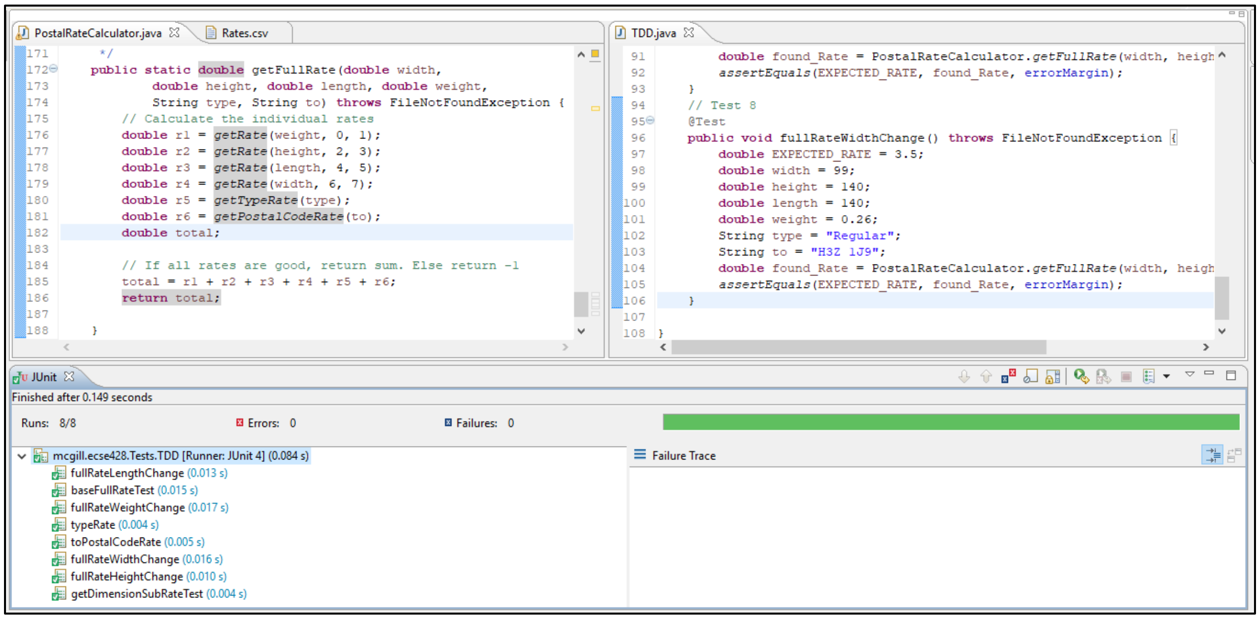
Inputs and Expected Output:

- ***weight (0.26 Kg)***, ***height (140 cm)***, ***width (99 cm)***, ***length (140 cm)***, and strings for the ***destination postal code*** (H3Z 1J9), and ***type of postage*** (Regular).
- The output is the proper parcel rate (sum of all sub rates).

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



PostalRateCalculator.java Rates.csv

```
     */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;

    }
```

TDD.java

```
        double found_Rate = PostalRateCalculator.getFullRate(width, heigh
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 8
    @Test
    public void fullRateWidthChange() throws FileNotFoundException {
        double EXPECTED_RATE = 3.5;
        double width = 99;
        double height = 140;
        double length = 140;
        double weight = 0.26;
        String type = "Regular";
        String to = "H3Z 1J9";
        double found_Rate = PostalRateCalculator.getFullRate(width, heigh
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }

}
```

JUnit

```
Finished after 0.149 seconds

Runs: 8/8        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.084 s)
    fullRateLengthChange (0.013 s)
    baseFullRateTest (0.015 s)
    fullRateWeightChange (0.017 s)
    typeRate (0.004 s)
    toPostalCodeRate (0.005 s)
    fullRateWidthChange (0.016 s)
    fullRateHeightChange (0.010 s)
    getDimensionSubRateTest (0.004 s)
```

Test 9: fullRateTypeChange

Purpose:

- Test the ability to get the full rate of a parcel with the ***type*** of postage varied.

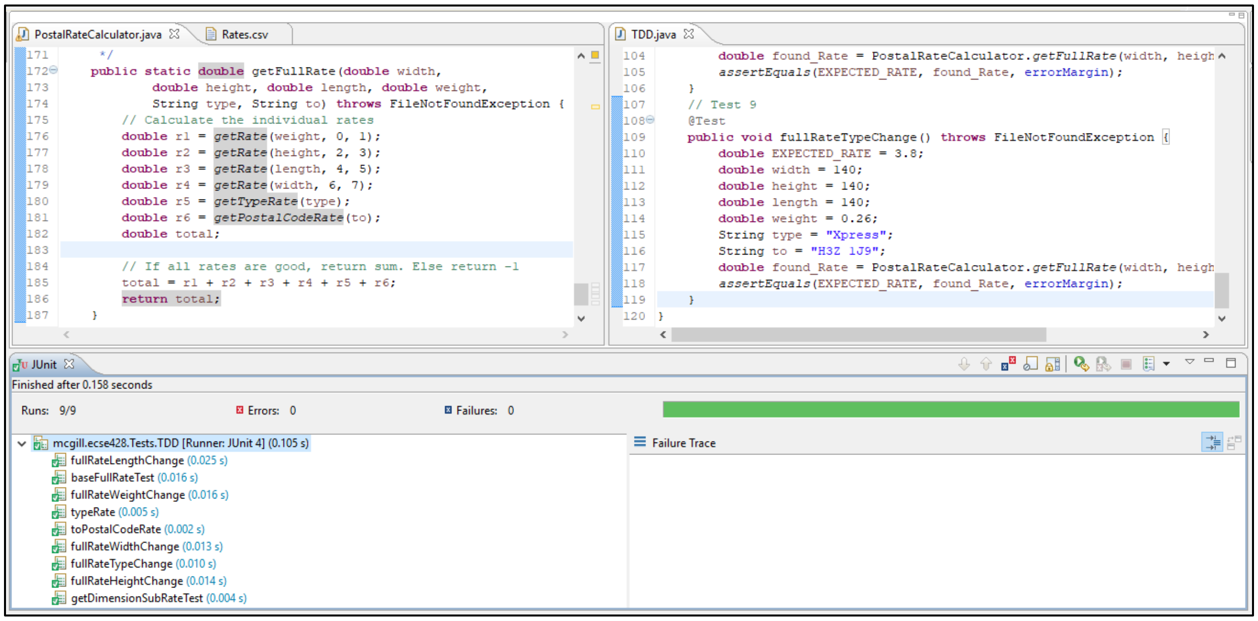
Inputs and Expected Output:

- ***weight (0.26 Kg)***, ***height (140 cm)***, ***width (99 cm)***, ***length (140 cm)***, and ***destination postal code*** (H3Z 1J9), and ***type of postage (Xpress).***
- The expected output is the proper parcel rate (sum of all sub rates).

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```java
     */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;
    }
```

TDD.java

```java
            double found_Rate = PostalRateCalculator.getFullRate(width, heigh
            assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 9
    @Test
    public void fullRateTypeChange() throws FileNotFoundException {
            double EXPECTED_RATE = 3.8;
            double width = 140;
            double height = 140;
            double length = 140;
            double weight = 0.26;
            String type = "Xpress";
            String to = "H3Z 1J9";
            double found_Rate = PostalRateCalculator.getFullRate(width, heigh
            assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
}
```

JUnit

```
Finished after 0.158 seconds
Runs: 9/9        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.105 s)
    fullRateLengthChange (0.025 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.016 s)
    typeRate (0.005 s)
    toPostalCodeRate (0.002 s)
    fullRateWidthChange (0.013 s)
    fullRateTypeChange (0.010 s)
    fullRateHeightChange (0.014 s)
    getDimensionSubRateTest (0.004 s)
```

Test 10: fullRateToPostalCodeChange

Purpose:

- Test the ability to get the full rate of a parcel with the ***destination*** postal code varied.

Inputs and Expected Output:

- ***weight (0.26 Kg)***, ***height (140 cm)***, ***width (99 cm)***, ***length (140 cm)***, and the ***destination postal code (J3Q 1R9)*** within Quebec, and ***type of postage*** (Regular).
- The expected output is the proper parcel rate (sum of all sub rates).

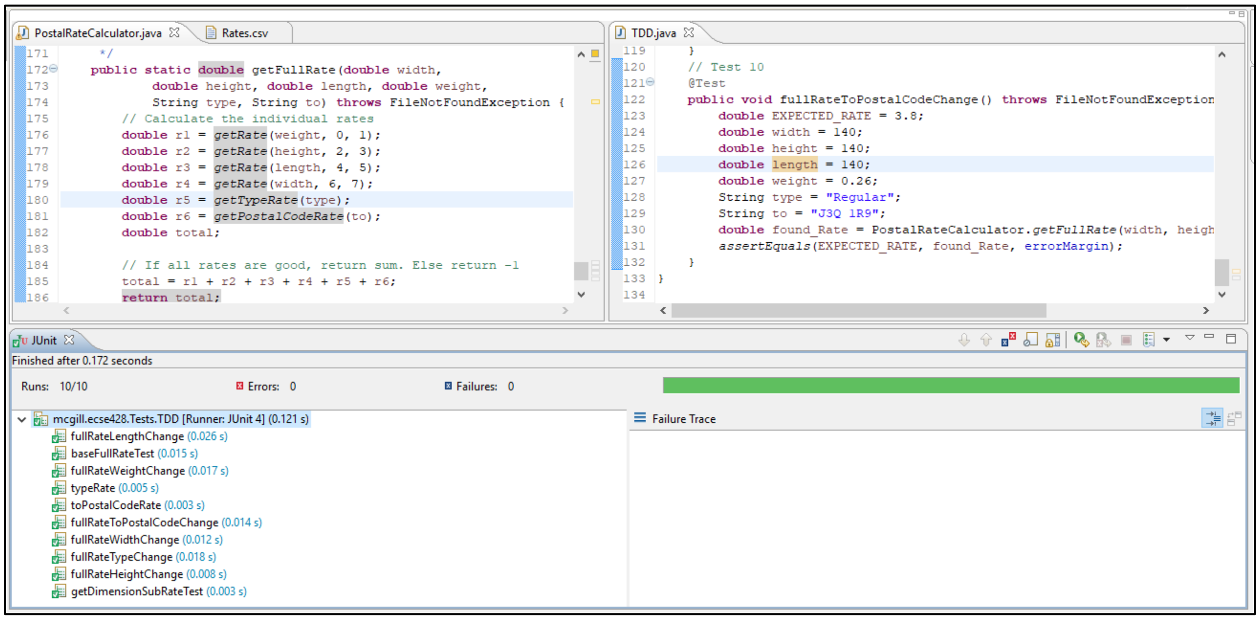
Assumptions:

- All the double input values are within an accepted range, and of proper format.
- The origin postal code has been verified start with an H.
- The origin and destination postal code match the proper postal code patterns.
- That the type of postage string has been verified.

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
     */
    public static double getFullRate(double width,
            double height, double length, double weight,
            String type, String to) throws FileNotFoundException {
        // Calculate the individual rates
        double r1 = getRate(weight, 0, 1);
        double r2 = getRate(height, 2, 3);
        double r3 = getRate(length, 4, 5);
        double r4 = getRate(width, 6, 7);
        double r5 = getTypeRate(type);
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;
```

TDD.java

```
    }
    // Test 10
    @Test
    public void fullRateToPostalCodeChange() throws FileNotFoundException
        double EXPECTED_RATE = 3.8;
        double width = 140;
        double height = 140;
        double length = 140;
        double weight = 0.26;
        String type = "Regular";
        String to = "J3Q 1R9";
        double found_Rate = PostalRateCalculator.getFullRate(width, heigh
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
}
```

JUnit

```
Finished after 0.172 seconds
Runs: 10/10    Errors: 0    Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.121 s)
    fullRateLengthChange (0.026 s)
    baseFullRateTest (0.015 s)
    fullRateWeightChange (0.017 s)
    typeRate (0.005 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.014 s)
    fullRateWidthChange (0.012 s)
    fullRateTypeChange (0.018 s)
    fullRateHeightChange (0.008 s)
    getDimensionSubRateTest (0.003 s)
```

Test 11: inValidDimensionFormat

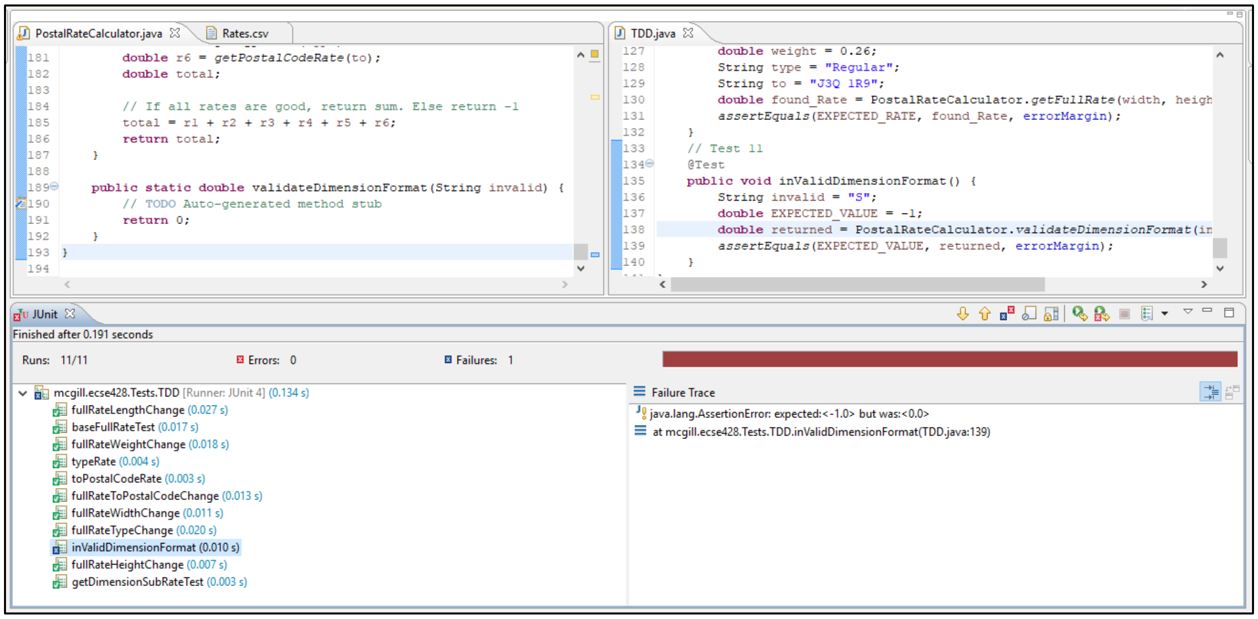
Purpose:

- To verify that incorrect numeric format (e.g an alphanumeric value) is detected.

Inputs and Expected Output:

- An alphanumeric string not representing a number.
- The expected output is a double of value -1.

Failing Screenshot:



PostalRateCalculator.java   Rates.csv

```
        double r6 = getPostalCodeRate(to);
        double total;

        // If all rates are good, return sum. Else return -1
        total = r1 + r2 + r3 + r4 + r5 + r6;
        return total;
    }

    public static double validateDimensionFormat(String invalid) {
        // TODO Auto-generated method stub
        return 0;
    }
}
```

TDD.java

```
        double weight = 0.26;
        String type = "Regular";
        String to = "J3Q 1R9";
        double found_Rate = PostalRateCalculator.getFullRate(width, heigh
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 11
    @Test
    public void inValidDimensionFormat() {
        String invalid = "S";
        double EXPECTED_VALUE = -1;
        double returned = PostalRateCalculator.validateDimensionFormat(in
        assertEquals(EXPECTED_VALUE, returned, errorMargin);
    }
```

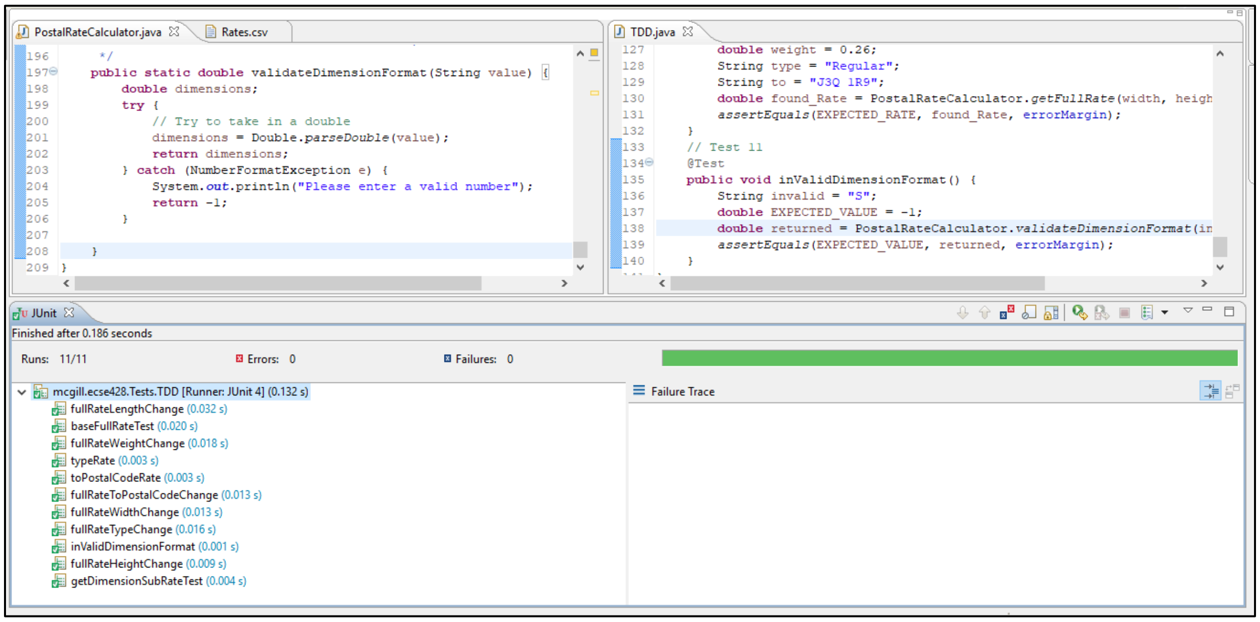
JUnit
Finished after 0.191 seconds
Runs: 11/11   Errors: 0   Failures: 1

```
mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.134 s)
    fullRateLengthChange (0.027 s)
    baseFullRateTest (0.017 s)
    fullRateWeightChange (0.018 s)
    typeRate (0.004 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.013 s)
    fullRateWidthChange (0.011 s)
    fullRateTypeChange (0.020 s)
    inValidDimensionFormat (0.010 s)
    fullRateHeightChange (0.007 s)
    getDimensionSubRateTest (0.003 s)
```

Failure Trace

```
java.lang.AssertionError: expected:<-1.0> but was:<0.0>
at mcgill.ecse428.Tests.TDD.inValidDimensionFormat(TDD.java:139)
```

Passing Screenshot:



PostalRateCalculator.java   Rates.csv

```
     */
    public static double validateDimensionFormat(String value) {
        double dimensions;
        try {
            // Try to take in a double
            dimensions = Double.parseDouble(value);
            return dimensions;
        } catch (NumberFormatException e) {
            System.out.println("Please enter a valid number");
            return -1;
        }

    }
}
```

TDD.java

```
        double weight = 0.26;
        String type = "Regular";
        String to = "J3Q 1R9";
        double found_Rate = PostalRateCalculator.getFullRate(width, heigh
        assertEquals(EXPECTED_RATE, found_Rate, errorMargin);
    }
    // Test 11
    @Test
    public void inValidDimensionFormat() {
        String invalid = "S";
        double EXPECTED_VALUE = -1;
        double returned = PostalRateCalculator.validateDimensionFormat(in
        assertEquals(EXPECTED_VALUE, returned, errorMargin);
    }
```

JUnit
Finished after 0.186 seconds
Runs: 11/11   Errors: 0   Failures: 0

```
mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.132 s)
    fullRateLengthChange (0.032 s)
    baseFullRateTest (0.020 s)
    fullRateWeightChange (0.018 s)
    typeRate (0.003 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.013 s)
    fullRateWidthChange (0.013 s)
    fullRateTypeChange (0.016 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.009 s)
    getDimensionSubRateTest (0.004 s)
```

Failure Trace

Test 12: weightOutOfRange

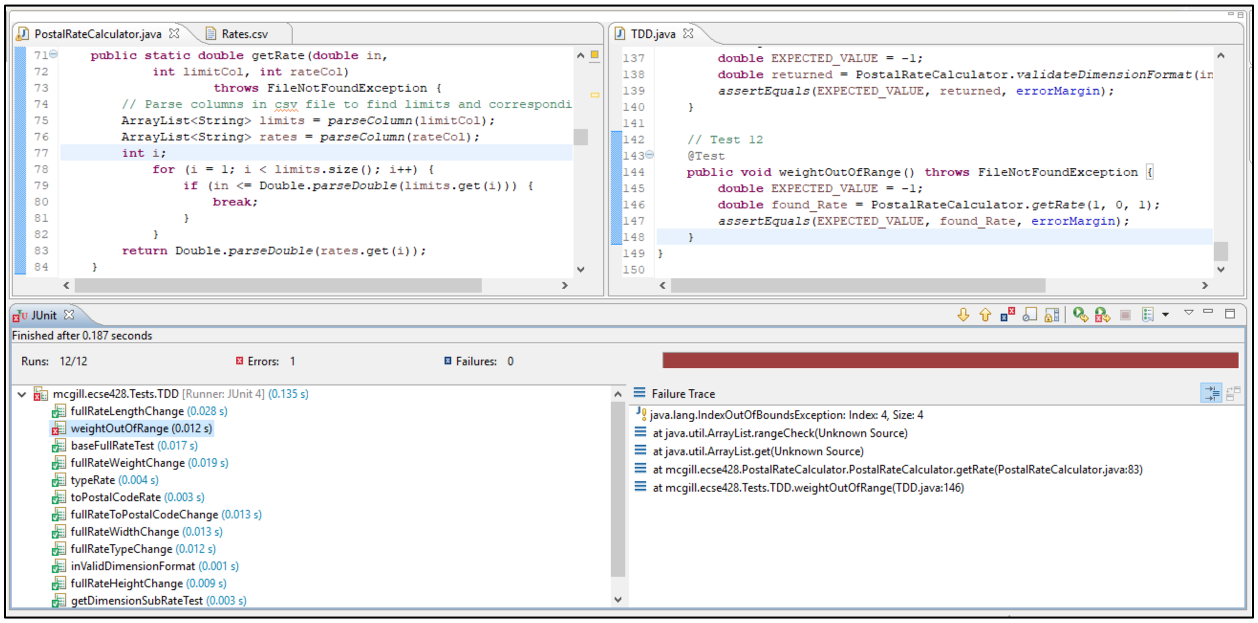
Purpose:

- Verify that the value entered for the ***weight dimension*** is out of range.

Inputs and Expected Output:

- An out of range double (1), and the columns of the weight dimension ranges, and rates.
- The expected output is a double of value -1 indicating an out of range value.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

```
    public static double getRate(double in,
            int limitCol, int rateCol)
                throws FileNotFoundException {
        // Parse columns in csv file to find limits and correspondi
        ArrayList<String> limits = parseColumn(limitCol);
        ArrayList<String> rates = parseColumn(rateCol);
        int i;
        for (i = 1; i < limits.size(); i++) {
            if (in <= Double.parseDouble(limits.get(i))) {
                break;
            }
        }
        return Double.parseDouble(rates.get(i));
    }
```

TDD.java

```
        double EXPECTED_VALUE = -1;
        double returned = PostalRateCalculator.validateDimensionFormat(in
        assertEquals(EXPECTED_VALUE, returned, errorMargin);
    }

    // Test 12
    @Test
    public void weightOutOfRange() throws FileNotFoundException {
        double EXPECTED_VALUE = -1;
        double found_Rate = PostalRateCalculator.getRate(1, 0, 1);
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);
    }
}
```

JUnit

Finished after 0.187 seconds

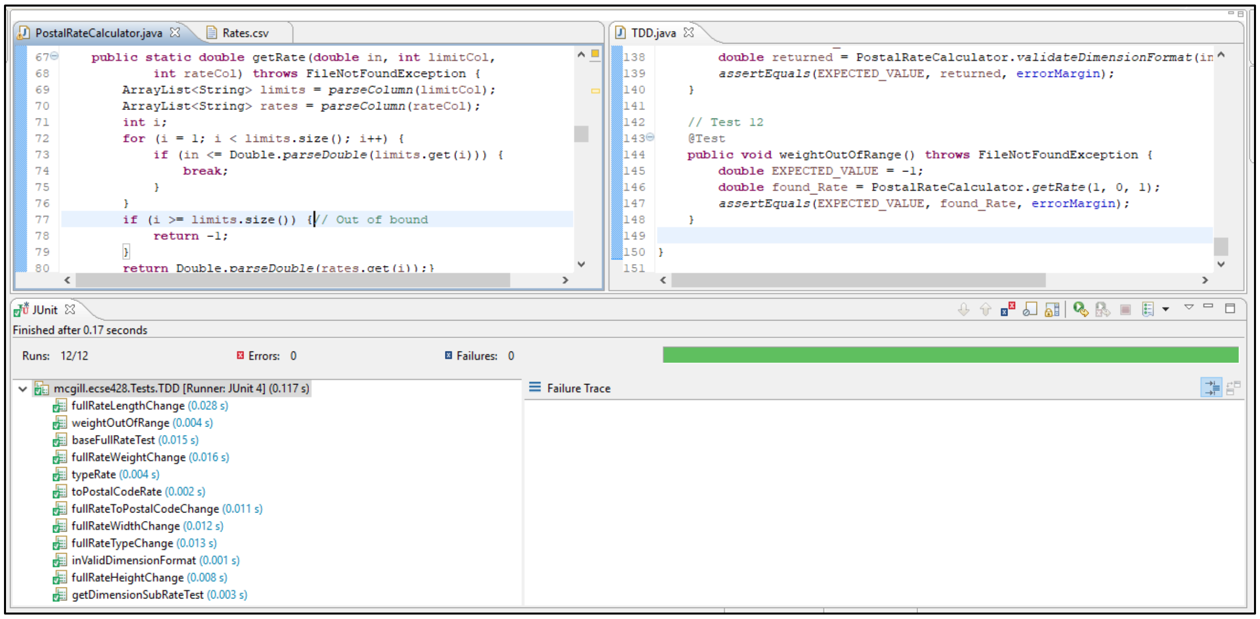
Runs: 12/12 Errors: 1 Failures: 0

```
mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.135 s)
    fullRateLengthChange (0.028 s)
    weightOutOfRange (0.012 s)
    baseFullRateTest (0.017 s)
    fullRateWeightChange (0.019 s)
    typeRate (0.004 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.013 s)
    fullRateWidthChange (0.013 s)
    fullRateTypeChange (0.012 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.009 s)
    getDimensionSubRateTest (0.003 s)
```

Failure Trace

```
java.lang.IndexOutOfBoundsException: Index: 4, Size: 4
at java.util.ArrayList.rangeCheck(Unknown Source)
at java.util.ArrayList.get(Unknown Source)
at mcgill.ecse428.PostalRateCalculator.PostalRateCalculator.getRate(PostalRateCalculator.java:83)
at mcgill.ecse428.Tests.TDD.weightOutOfRange(TDD.java:146)
```

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
    public static double getRate(double in, int limitCol,
            int rateCol) throws FileNotFoundException {
        ArrayList<String> limits = parseColumn(limitCol);
        ArrayList<String> rates = parseColumn(rateCol);
        int i;
        for (i = 1; i < limits.size(); i++) {
            if (in <= Double.parseDouble(limits.get(i))) {
                break;
            }
        }
        if (i >= limits.size()) {// Out of bound
            return -1;
        }
        return Double.parseDouble(rates.get(i));}
```

TDD.java

```
        double returned = PostalRateCalculator.validateDimensionFormat(in
        assertEquals(EXPECTED_VALUE, returned, errorMargin);
    }

    // Test 12
    @Test
    public void weightOutOfRange() throws FileNotFoundException {
        double EXPECTED_VALUE = -1;
        double found_Rate = PostalRateCalculator.getRate(1, 0, 1);
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);
    }

}
```

JUnit

Finished after 0.17 seconds

Runs: 12/12 Errors: 0 Failures: 0

```
mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.117 s)
    fullRateLengthChange (0.028 s)
    weightOutOfRange (0.004 s)
    baseFullRateTest (0.015 s)
    fullRateWeightChange (0.016 s)
    typeRate (0.004 s)
    toPostalCodeRate (0.002 s)
    fullRateToPostalCodeChange (0.011 s)
    fullRateWidthChange (0.012 s)
    fullRateTypeChange (0.013 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.008 s)
    getDimensionSubRateTest (0.003 s)
```

Failure Trace

Test 13: heightOutOfRange

Purpose:

- Verify that the value entered for the ***height dimension*** is out of range. The ranges are found csv file.

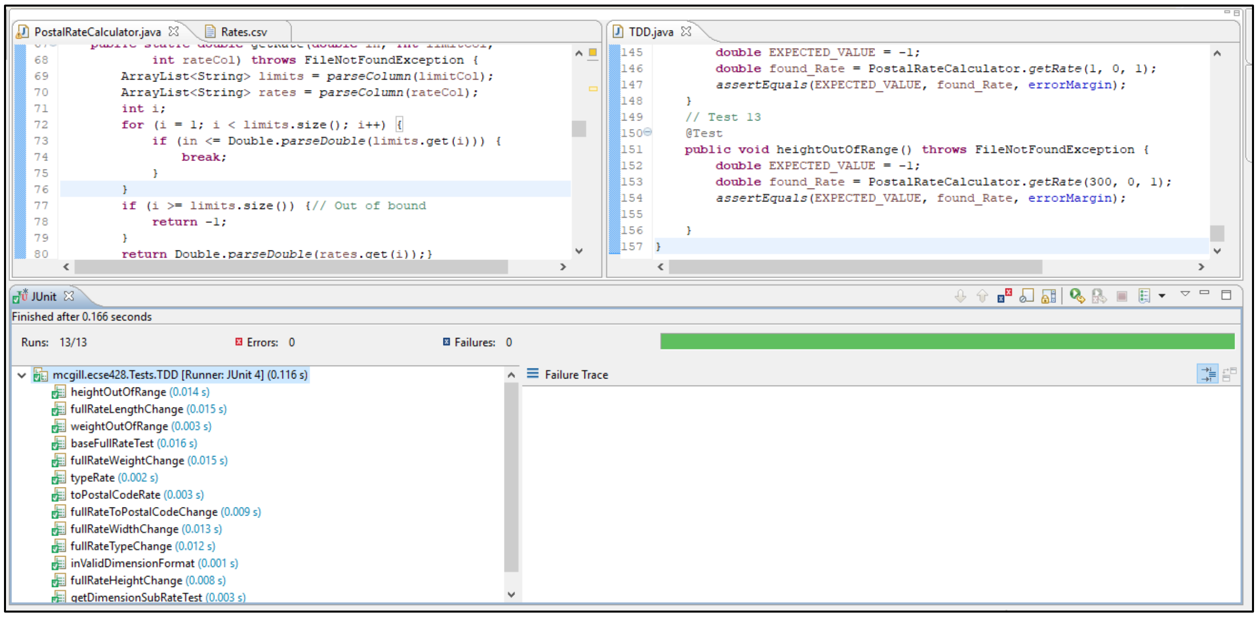
Inputs and Expected Output:

- An out of range double (300), and the columns of the height dimension ranges, and rates.
- The expected output is a double of value -1 indicating an out of range.

Failing Screenshot:

This test passes as it relies on previously tested methods, with no need for any additional logic.

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
            int rateCol) throws FileNotFoundException {
        ArrayList<String> limits = parseColumn(limitCol);
        ArrayList<String> rates = parseColumn(rateCol);
        int i;
        for (i = 1; i < limits.size(); i++) {
            if (in <= Double.parseDouble(limits.get(i))) {
                break;
            }
        }
        if (i >= limits.size()) {// Out of bound
            return -1;
        }
        return Double.parseDouble(rates.get(i));}
```

TDD.java

```
            double EXPECTED_VALUE = -1;
            double found_Rate = PostalRateCalculator.getRate(1, 0, 1);
            assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);
        }
        // Test 13
        @Test
        public void heightOutOfRange() throws FileNotFoundException {
            double EXPECTED_VALUE = -1;
            double found_Rate = PostalRateCalculator.getRate(300, 0, 1);
            assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);

        }
}
```

JUnit

```
Finished after 0.166 seconds
Runs:  13/13        Errors:  0        Failures:  0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.116 s)
    heightOutOfRange (0.014 s)
    fullRateLengthChange (0.015 s)
    weightOutOfRange (0.003 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.015 s)
    typeRate (0.002 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.009 s)
    fullRateWidthChange (0.013 s)
    fullRateTypeChange (0.012 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.008 s)
    getDimensionSubRateTest (0.003 s)
```

Failure Trace

Test 14: lengthOutOfRange

Purpose:

- Verify that the value entered for the ***length dimension*** is out of range. The ranges are found the csv file.

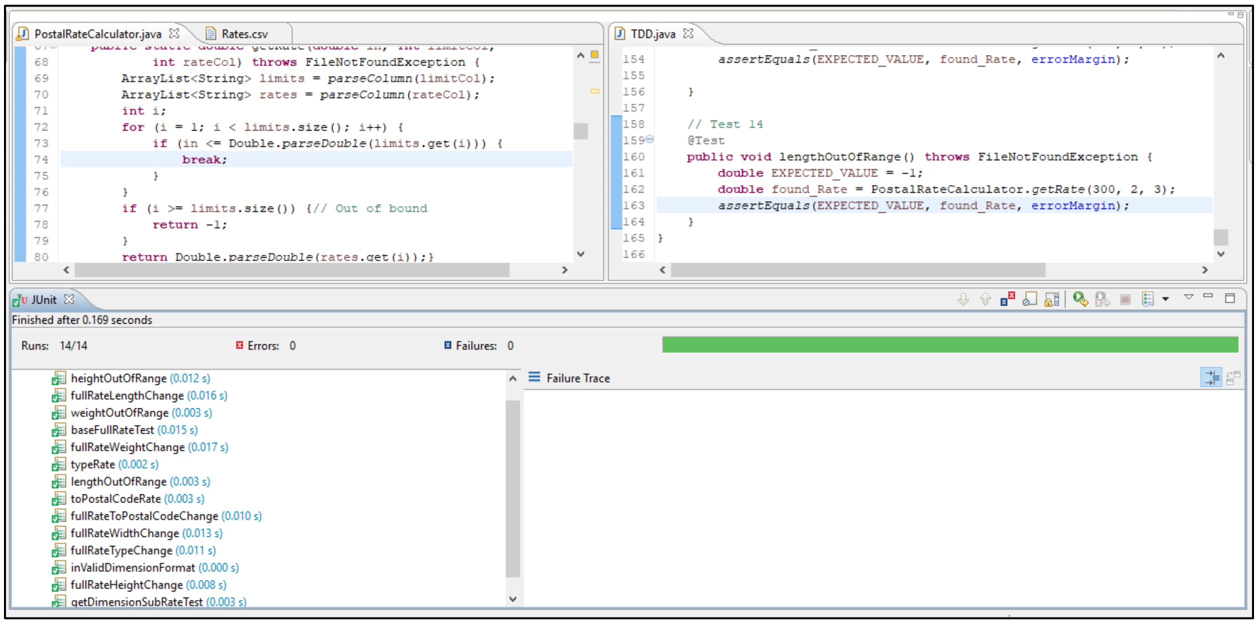
Inputs and Expected Output:

- An out of range double (300), and the columns of the length dimension ranges, and rates.
- The expected output is a double of value -1 indicating an error.

Failing Screenshot:

This test passes as it relies on previously tested methods, with no need for any additional logic.

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```java
            int rateCol) throws FileNotFoundException {
        ArrayList<String> limits = parseColumn(limitCol);
        ArrayList<String> rates = parseColumn(rateCol);
        int i;
        for (i = 1; i < limits.size(); i++) {
            if (in <= Double.parseDouble(limits.get(i))) {
                break;
            }
        }
        if (i >= limits.size()) {// Out of bound
            return -1;
        }
        return Double.parseDouble(rates.get(i));}
```

TDD.java

```java
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);

    }

    // Test 14
    @Test
    public void lengthOutOfRange() throws FileNotFoundException {
        double EXPECTED_VALUE = -1;
        double found_Rate = PostalRateCalculator.getRate(300, 2, 3);
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);
    }
}
```

JUnit

```
Finished after 0.169 seconds
Runs: 14/14        Errors: 0        Failures: 0

heightOutOfRange (0.012 s)
fullRateLengthChange (0.016 s)
weightOutOfRange (0.003 s)
baseFullRateTest (0.015 s)
fullRateWeightChange (0.017 s)
typeRate (0.002 s)
lengthOutOfRange (0.003 s)
toPostalCodeRate (0.003 s)
fullRateToPostalCodeChange (0.010 s)
fullRateWidthChange (0.013 s)
fullRateTypeChange (0.011 s)
inValidDimensionFormat (0.000 s)
fullRateHeightChange (0.008 s)
getDimensionSubRateTest (0.003 s)
```

Test 15: widthOutOfRange

Purpose:

- Verify that the value entered for the ***width dimension*** is out of range. The ranges are found in the csv file.

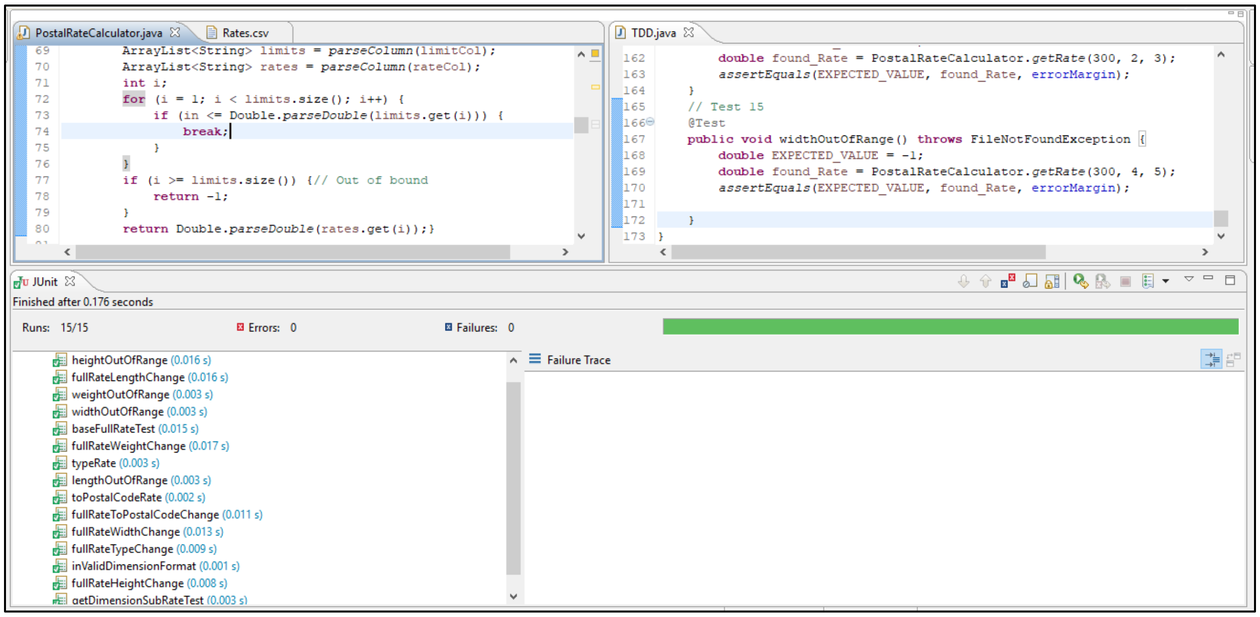
Inputs and Expected Output:

- An out of range double (300), and the expected output is a double of value -1.

Failing Screenshot:

This test does not fail, since the previous logic gives the desired output.

Passing Screenshot:



```java
        ArrayList<String> limits = parseColumn(limitCol);
        ArrayList<String> rates = parseColumn(rateCol);
        int i;
        for (i = 1; i < limits.size(); i++) {
            if (in <= Double.parseDouble(limits.get(i))) {
                break;
            }
        }
        if (i >= limits.size()) {// Out of bound
            return -1;
        }
        return Double.parseDouble(rates.get(i));}
```

```java
        double found_Rate = PostalRateCalculator.getRate(300, 2, 3);
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);
    }
    // Test 15
    @Test
    public void widthOutOfRange() throws FileNotFoundException {
        double EXPECTED_VALUE = -1;
        double found_Rate = PostalRateCalculator.getRate(300, 4, 5);
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);

    }
}
```

```
Finished after 0.176 seconds
Runs: 15/15    Errors: 0    Failures: 0

heightOutOfRange (0.016 s)
fullRateLengthChange (0.016 s)
weightOutOfRange (0.003 s)
widthOutOfRange (0.003 s)
baseFullRateTest (0.015 s)
fullRateWeightChange (0.017 s)
typeRate (0.003 s)
lengthOutOfRange (0.003 s)
toPostalCodeRate (0.002 s)
fullRateToPostalCodeChange (0.011 s)
fullRateWidthChange (0.013 s)
fullRateTypeChange (0.009 s)
inValidDimensionFormat (0.001 s)
fullRateHeightChange (0.008 s)
getDimensionSubRateTest (0.003 s)
```

Test 16: invalidType

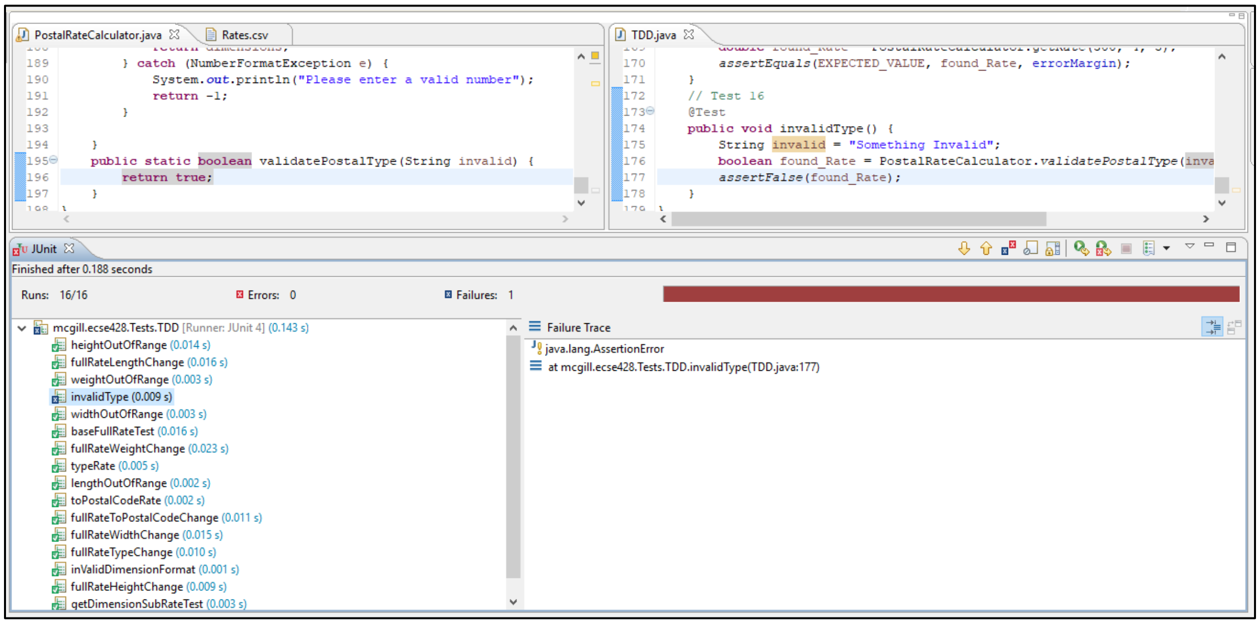
Purpose:

- Verify the user input for the type of postage. Ensuring that the user selects a type of postage from those that are accepted.

Inputs and Expected Output:

- An invalid string, and the expected output is false.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

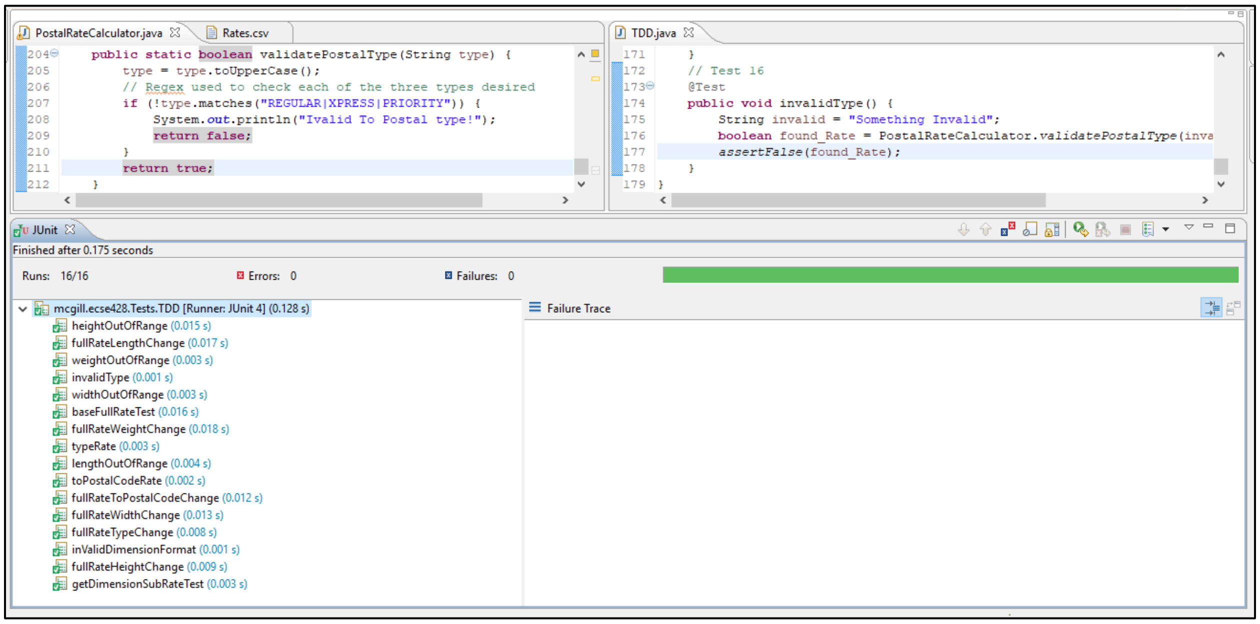
```java
        } catch (NumberFormatException e) {
            System.out.println("Please enter a valid number");
            return -1;
        }

    }
    public static boolean validatePostalType(String invalid) {
        return true;
    }
```

TDD.java

```java
        assertEquals(EXPECTED_VALUE, found_Rate, errorMargin);
    }
    // Test 16
    @Test
    public void invalidType() {
        String invalid = "Something Invalid";
        boolean found_Rate = PostalRateCalculator.validatePostalType(inva
        assertFalse(found_Rate);
    }
```

```
Finished after 0.188 seconds
Runs: 16/16        Errors: 0        Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.143 s)
    heightOutOfRange (0.014 s)
    fullRateLengthChange (0.016 s)
    weightOutOfRange (0.003 s)
    invalidType (0.009 s)
    widthOutOfRange (0.003 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.023 s)
    typeRate (0.005 s)
    lengthOutOfRange (0.002 s)
    toPostalCodeRate (0.002 s)
    fullRateToPostalCodeChange (0.011 s)
    fullRateWidthChange (0.015 s)
    fullRateTypeChange (0.010 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.009 s)
    getDimensionSubRateTest (0.003 s)

Failure Trace
java.lang.AssertionError
at mcgill.ecse428.Tests.TDD.invalidType(TDD.java:177)
```

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```java
    public static boolean validatePostalType(String type) {
        type = type.toUpperCase();
        // Regex used to check each of the three types desired
        if (!type.matches("REGULAR|XPRESS|PRIORITY")) {
            System.out.println("Ivalid To Postal type!");
            return false;
        }
        return true;
    }
```

TDD.java

```java
    }
    // Test 16
    @Test
    public void invalidType() {
        String invalid = "Something Invalid";
        boolean found_Rate = PostalRateCalculator.validatePostalType(inva
        assertFalse(found_Rate);
    }
}
```

```
Finished after 0.175 seconds
Runs: 16/16        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.128 s)
    heightOutOfRange (0.015 s)
    fullRateLengthChange (0.017 s)
    weightOutOfRange (0.003 s)
    invalidType (0.001 s)
    widthOutOfRange (0.003 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.018 s)
    typeRate (0.003 s)
    lengthOutOfRange (0.004 s)
    toPostalCodeRate (0.002 s)
    fullRateToPostalCodeChange (0.012 s)
    fullRateWidthChange (0.013 s)
    fullRateTypeChange (0.008 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.009 s)
    getDimensionSubRateTest (0.003 s)

Failure Trace
```

Test 17: invalidFromPostalCode

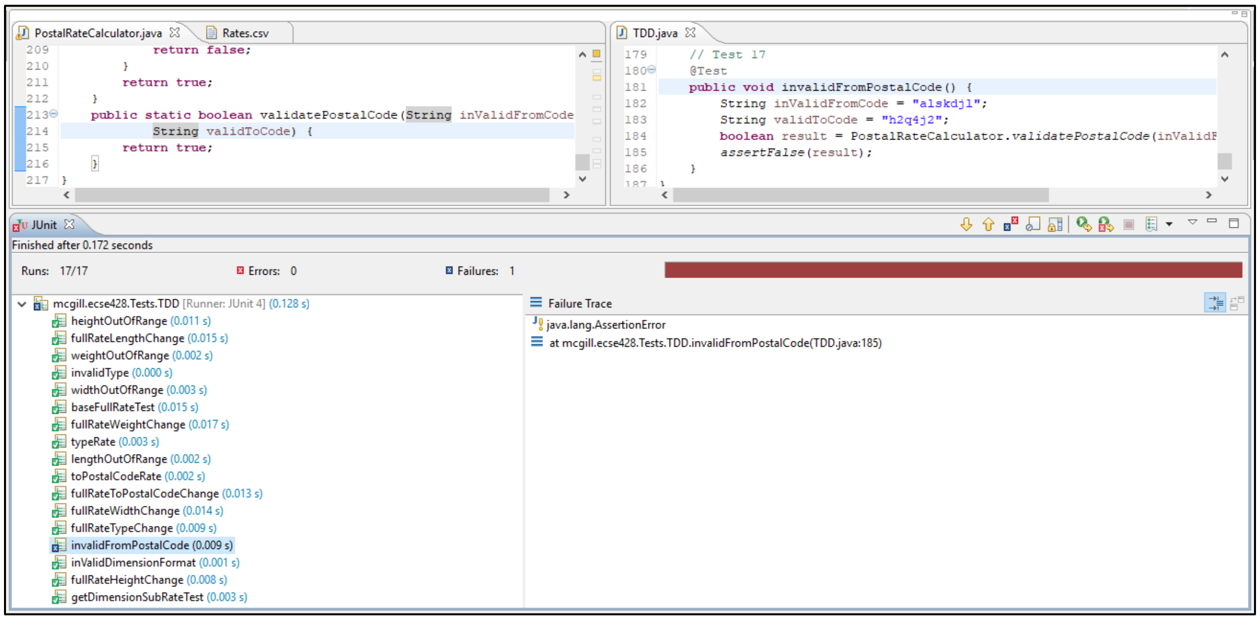
Purpose:

- Verify that the origin postal code is of valid format (H1A 1A1), ***<u>starting with H</u>*** indicating that the package is sent from Montreal.

Inputs and Expected Output:

- An invalid Origin postal code format.
- The expected output is a false Boolean.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

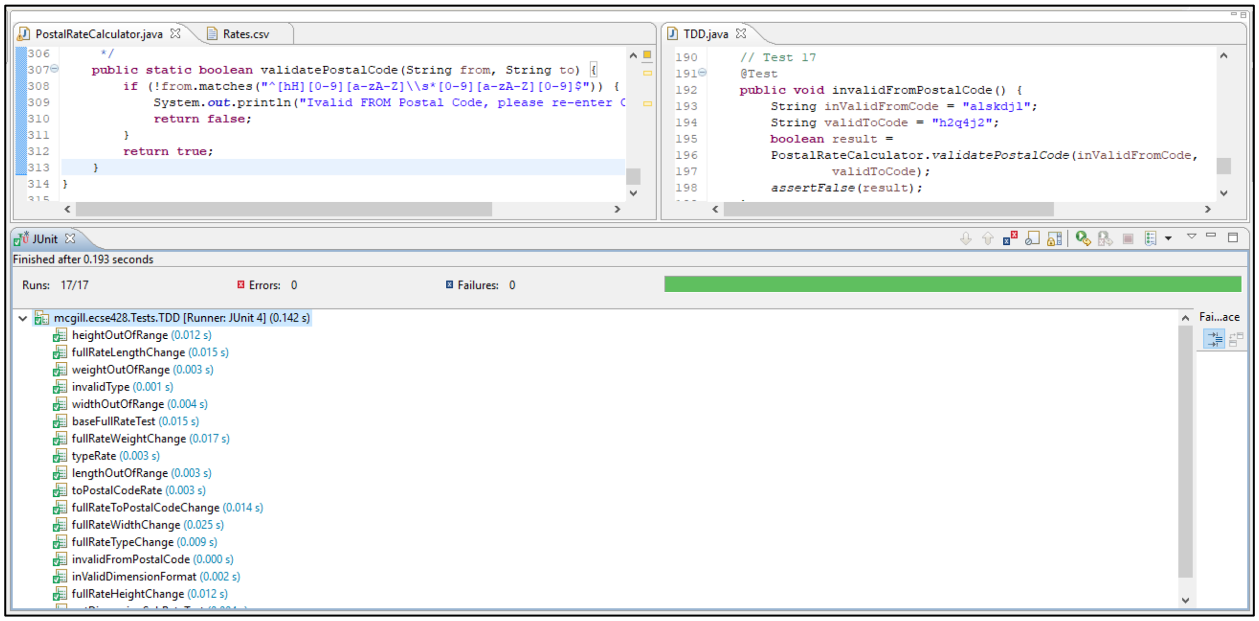
```
                return false;
            }
            return true;
        }
        public static boolean validatePostalCode(String inValidFromCode
                String validToCode) {
            return true;
        }
}
```

TDD.java

```
    // Test 17
    @Test
    public void invalidFromPostalCode() {
        String inValidFromCode = "alskdjl";
        String validToCode = "h2q4j2";
        boolean result = PostalRateCalculator.validatePostalCode(inValidF
        assertFalse(result);
    }
}
```

```
JUnit
Finished after 0.172 seconds
Runs: 17/17        Errors: 0        Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.128 s)
    heightOutOfRange (0.011 s)
    fullRateLengthChange (0.015 s)
    weightOutOfRange (0.002 s)
    invalidType (0.000 s)
    widthOutOfRange (0.003 s)
    baseFullRateTest (0.015 s)
    fullRateWeightChange (0.017 s)
    typeRate (0.003 s)
    lengthOutOfRange (0.002 s)
    toPostalCodeRate (0.002 s)
    fullRateToPostalCodeChange (0.013 s)
    fullRateWidthChange (0.014 s)
    fullRateTypeChange (0.009 s)
    invalidFromPostalCode (0.009 s)
    inValidDimensionFormat (0.001 s)
    fullRateHeightChange (0.008 s)
    getDimensionSubRateTest (0.003 s)

Failure Trace
java.lang.AssertionError
at mcgill.ecse428.Tests.TDD.invalidFromPostalCode(TDD.java:185)
```

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
     */
    public static boolean validatePostalCode(String from, String to) {
        if (!from.matches("^[hH][0-9][a-zA-Z]\\s*[0-9][a-zA-Z][0-9]$")) {
            System.out.println("Ivalid FROM Postal Code, please re-enter C
            return false;
        }
        return true;
    }
}
```

TDD.java

```
    // Test 17
    @Test
    public void invalidFromPostalCode() {
        String inValidFromCode = "alskdjl";
        String validToCode = "h2q4j2";
        boolean result =
        PostalRateCalculator.validatePostalCode(inValidFromCode,
                validToCode);
        assertFalse(result);
```

```
JUnit
Finished after 0.193 seconds
Runs: 17/17        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.142 s)
    heightOutOfRange (0.012 s)
    fullRateLengthChange (0.015 s)
    weightOutOfRange (0.003 s)
    invalidType (0.001 s)
    widthOutOfRange (0.004 s)
    baseFullRateTest (0.015 s)
    fullRateWeightChange (0.017 s)
    typeRate (0.003 s)
    lengthOutOfRange (0.003 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.014 s)
    fullRateWidthChange (0.025 s)
    fullRateTypeChange (0.009 s)
    invalidFromPostalCode (0.000 s)
    inValidDimensionFormat (0.002 s)
    fullRateHeightChange (0.012 s)

Fai...ace
```

Test 18: invalidToPostalCode

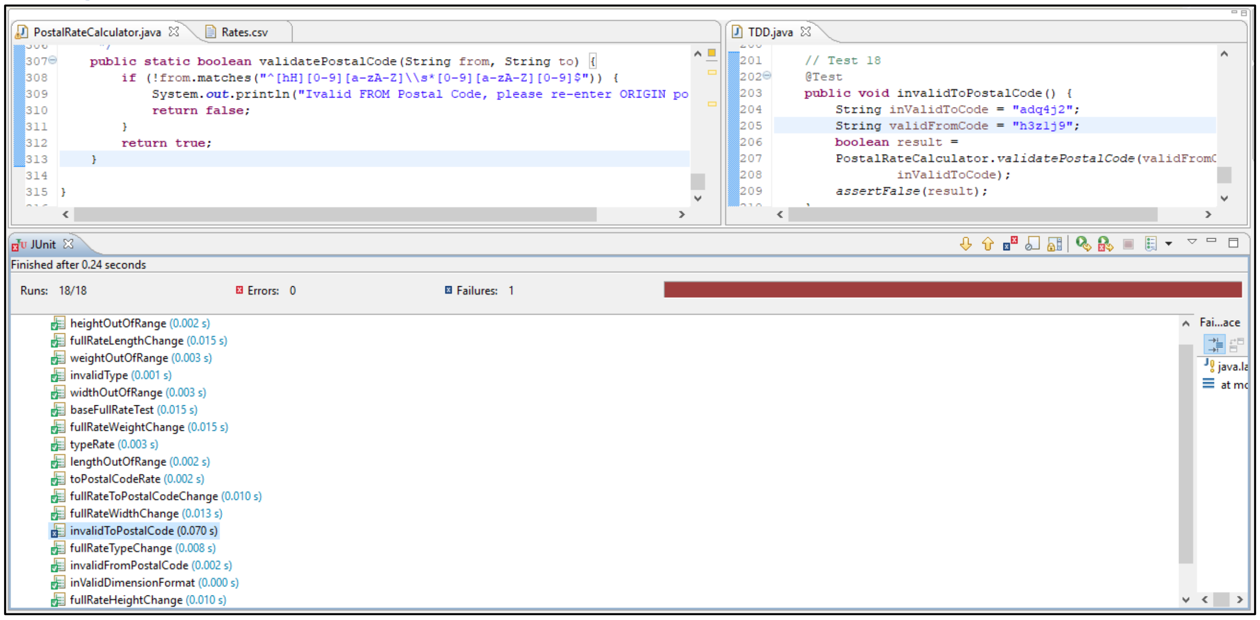
Purpose:

- Verify that the destination postal code is of valid format (A1A 1A1).

Inputs and Expected Output:

- An invalid destination postal code format.
- The expected output is a false Boolean.

Failing Screenshot:



PostalRateCalculator.java | Rates.csv

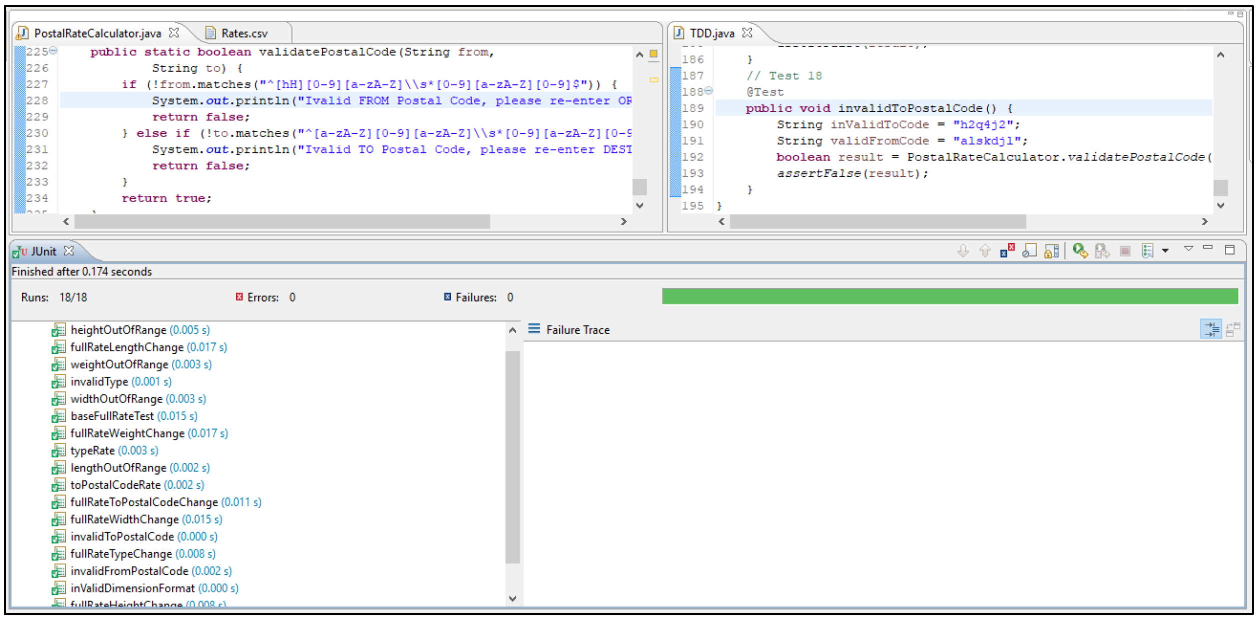
```
    public static boolean validatePostalCode(String from, String to) {
        if (!from.matches("^[hH][0-9][a-zA-Z]\\s*[0-9][a-zA-Z][0-9]$")) {
            System.out.println("Invalid FROM Postal Code, please re-enter ORIGIN po
            return false;
        }
        return true;
    }

}
```

TDD.java

```
    // Test 18
    @Test
    public void invalidToPostalCode() {
        String inValidToCode = "adq4j2";
        String validFromCode = "h3z1j9";
        boolean result =
        PostalRateCalculator.validatePostalCode(validFromC
                inValidToCode);
        assertFalse(result);
```

```
Finished after 0.24 seconds
Runs: 18/18    Errors: 0    Failures: 1

heightOutOfRange (0.002 s)
fullRateLengthChange (0.015 s)
weightOutOfRange (0.003 s)
invalidType (0.001 s)
widthOutOfRange (0.003 s)
baseFullRateTest (0.015 s)
fullRateWeightChange (0.015 s)
typeRate (0.003 s)
lengthOutOfRange (0.002 s)
toPostalCodeRate (0.002 s)
fullRateToPostalCodeChange (0.010 s)
fullRateWidthChange (0.013 s)
invalidToPostalCode (0.070 s)
fullRateTypeChange (0.008 s)
invalidFromPostalCode (0.002 s)
inValidDimensionFormat (0.000 s)
fullRateHeightChange (0.010 s)
```

Passing Screenshot:



PostalRateCalculator.java | Rates.csv

```
    public static boolean validatePostalCode(String from,
            String to) {
        if (!from.matches("^[hH][0-9][a-zA-Z]\\s*[0-9][a-zA-Z][0-9]$")) {
            System.out.println("Invalid FROM Postal Code, please re-enter OF
            return false;
        } else if (!to.matches("^[a-zA-Z][0-9][a-zA-Z]\\s*[0-9][a-zA-Z][0-9
            System.out.println("Invalid TO Postal Code, please re-enter DEST
            return false;
        }
        return true;
```

TDD.java

```
    }
    // Test 18
    @Test
    public void invalidToPostalCode() {
        String inValidToCode = "h2q4j2";
        String validFromCode = "alskdjl";
        boolean result = PostalRateCalculator.validatePostalCode(
        assertFalse(result);
    }
}
```

```
Finished after 0.174 seconds
Runs: 18/18    Errors: 0    Failures: 0

heightOutOfRange (0.005 s)
fullRateLengthChange (0.017 s)
weightOutOfRange (0.003 s)
invalidType (0.001 s)
widthOutOfRange (0.003 s)
baseFullRateTest (0.015 s)
fullRateWeightChange (0.017 s)
typeRate (0.003 s)
lengthOutOfRange (0.002 s)
toPostalCodeRate (0.002 s)
fullRateToPostalCodeChange (0.011 s)
fullRateWidthChange (0.015 s)
invalidToPostalCode (0.000 s)
fullRateTypeChange (0.008 s)
invalidFromPostalCode (0.002 s)
inValidDimensionFormat (0.000 s)
```

Test 19: negativeDimensions

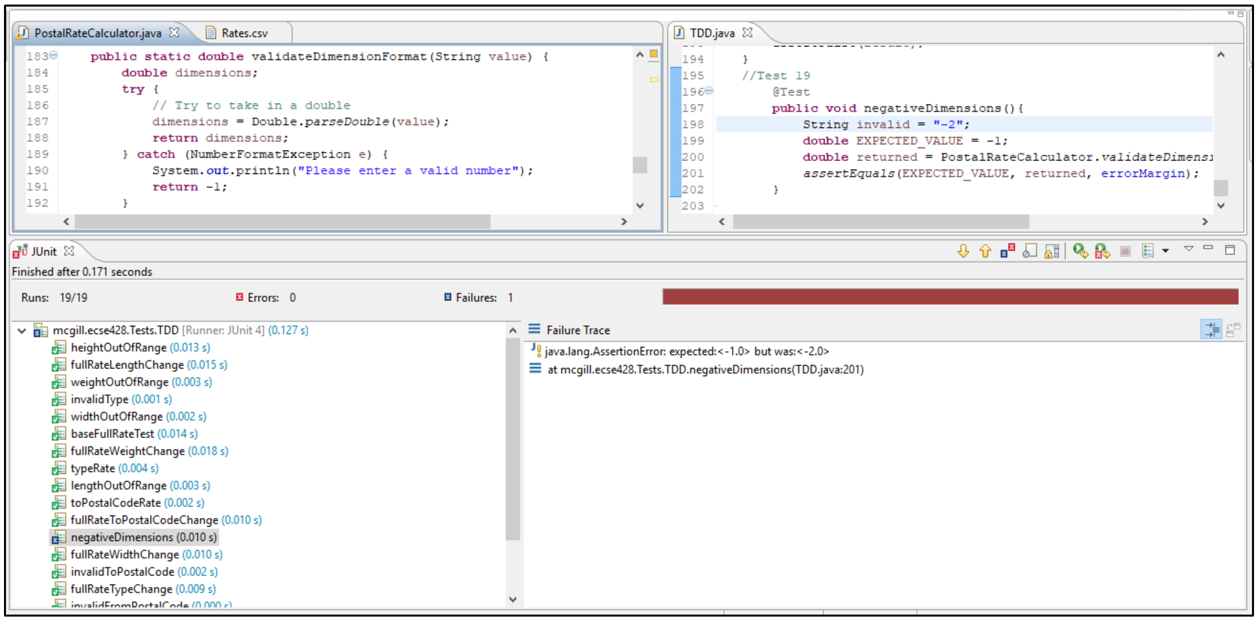
Purpose:

- Verify negative attributes (weight, height, length and width).

Inputs and Expected Output:

- A negative value.
- The expected output is value of -1 indicating an error.

Failing Screenshot:



PostalRateCalculator.java

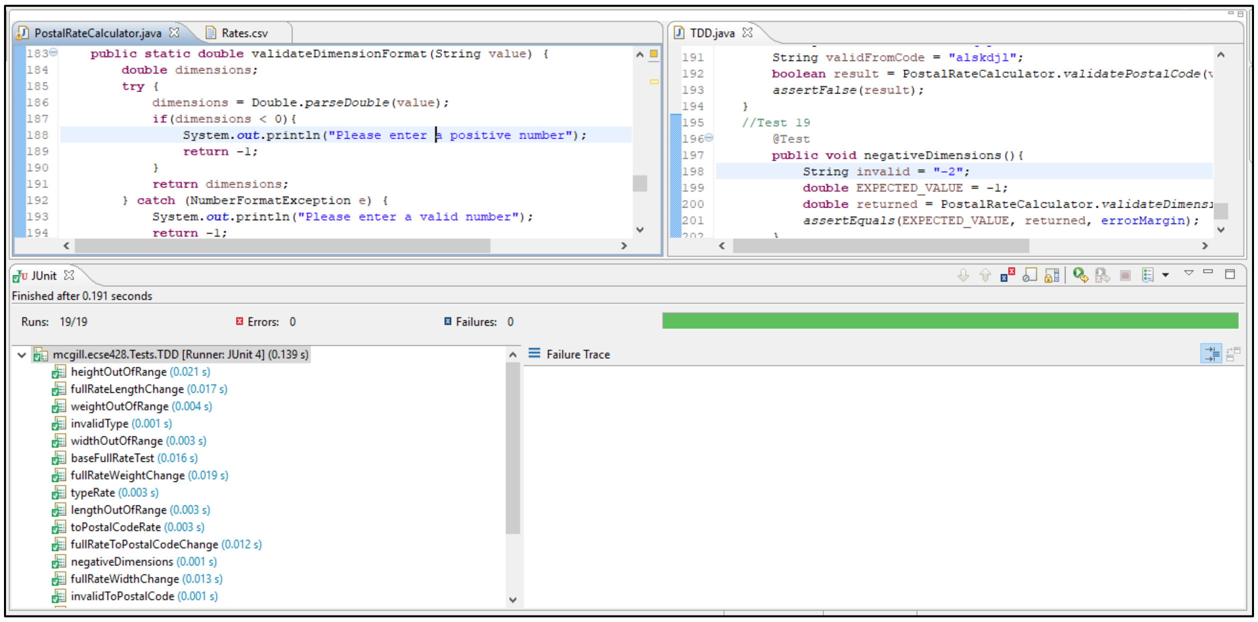
```java
    public static double validateDimensionFormat(String value) {
        double dimensions;
        try {
            // Try to take in a double
            dimensions = Double.parseDouble(value);
            return dimensions;
        } catch (NumberFormatException e) {
            System.out.println("Please enter a valid number");
            return -1;
        }
```

TDD.java

```java
    }
    //Test 19
        @Test
        public void negativeDimensions(){
            String invalid = "-2";
            double EXPECTED_VALUE = -1;
            double returned = PostalRateCalculator.validateDimensi
            assertEquals(EXPECTED_VALUE, returned, errorMargin);
        }
```

```
Finished after 0.171 seconds
Runs: 19/19        Errors: 0        Failures: 1

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.127 s)
    heightOutOfRange (0.013 s)
    fullRateLengthChange (0.015 s)
    weightOutOfRange (0.003 s)
    invalidType (0.001 s)
    widthOutOfRange (0.002 s)
    baseFullRateTest (0.014 s)
    fullRateWeightChange (0.018 s)
    typeRate (0.004 s)
    lengthOutOfRange (0.003 s)
    toPostalCodeRate (0.002 s)
    fullRateToPostalCodeChange (0.010 s)
    negativeDimensions (0.010 s)
    fullRateWidthChange (0.010 s)
    invalidToPostalCode (0.002 s)
    fullRateTypeChange (0.009 s)

Failure Trace
java.lang.AssertionError: expected:<-1.0> but was:<-2.0>
at mcgill.ecse428.Tests.TDD.negativeDimensions(TDD.java:201)
```

Passing Screenshot:



PostalRateCalculator.java

```java
    public static double validateDimensionFormat(String value) {
        double dimensions;
        try {
            dimensions = Double.parseDouble(value);
            if(dimensions < 0){
                System.out.println("Please enter a positive number");
                return -1;
            }
            return dimensions;
        } catch (NumberFormatException e) {
            System.out.println("Please enter a valid number");
            return -1;
```

TDD.java

```java
        String validFromCode = "alskdjl";
        boolean result = PostalRateCalculator.validatePostalCode(v
        assertFalse(result);
    }
    //Test 19
        @Test
        public void negativeDimensions(){
            String invalid = "-2";
            double EXPECTED_VALUE = -1;
            double returned = PostalRateCalculator.validateDimensi
            assertEquals(EXPECTED_VALUE, returned, errorMargin);
```

```
Finished after 0.191 seconds
Runs: 19/19        Errors: 0        Failures: 0

mcgill.ecse428.Tests.TDD [Runner: JUnit 4] (0.139 s)
    heightOutOfRange (0.021 s)
    fullRateLengthChange (0.017 s)
    weightOutOfRange (0.004 s)
    invalidType (0.001 s)
    widthOutOfRange (0.003 s)
    baseFullRateTest (0.016 s)
    fullRateWeightChange (0.019 s)
    typeRate (0.003 s)
    lengthOutOfRange (0.003 s)
    toPostalCodeRate (0.003 s)
    fullRateToPostalCodeChange (0.012 s)
    negativeDimensions (0.001 s)
    fullRateWidthChange (0.013 s)
    invalidToPostalCode (0.001 s)

Failure Trace
```