

XÂY DỰNG BÀI TOÁN VÀ CA KIỂM THỬ DỰA TRÊN PHƯƠNG PHÁP KIỂM THỬ LỚP TƯƠNG ĐƯƠNG & KIỂM THỬ BẢNG QUYẾT ĐỊNH

Họ tên: Nguyễn Thanh Huyền

MSSV: 18020666

Môn học: Kiểm thử và đảm bảo chất lượng phần mềm

1. Bài toán

Mô tả bài toán

Nhân dịp kỷ niệm 10 năm hoạt động (2011-2021), một hãng thời trang đã mở một chương trình khuyến mãi nhằm mục đích tri ân khách hàng. Hoạt động khuyến mãi này chỉ được áp dụng trên 1000 người mua đầu tiên với số lượng sản phẩm được mua tối đa 100 sản phẩm trên 1 người, đặc biệt trong số 1000 người đó, nếu là khách hàng lâu năm của hãng sẽ được hưởng thêm một mức ưu đãi bổ sung.

Trong ngày bắt đầu thực hiện chương trình, khách hàng sẽ được phát phiếu đánh số thứ tự tương ứng với thứ tự đến cửa hàng, có tổng cộng 1000 phiếu sẽ được phát ra. Sau khi mua hàng, dựa vào số trên mã phiếu, số lượng sản phẩm mua và thời gian là thành viên của hãng, phần mềm sẽ tính toán mức khuyến mãi cho khách hàng.

- Mức khuyến mãi được áp dụng trên mã phiếu và số lượng sản phẩm

<div>Mã phiếu</div> <div>Số Lượng</div>	[1;50]	(50;100]	(100;1000]
[1;20)	25%	10%	0%
[20;100]		25%	15%

- Bên cạnh đó, khách hàng nếu có năm thành viên trong khoảng [2011;2018] sẽ được cộng thêm 25% vào mức khuyến mãi trên.*

(Ngoài khoảng các giá trị trên thì kết quả đầu ra sẽ là không hợp lệ)

Đầu vào:

Gồm bộ 3 giá trị (Mã phiếu; Số lượng; Năm thành viên)

- Mã phiếu(X): là một số nguyên dương ứng với số thứ tự khách hàng đến mua, phiếu được đánh số bắt đầu từ 1 và tổng số phiếu được phát ra là 1000 phiếu.
Miền giá trị: X thuộc \mathbb{Z} , X thuộc [1;1000]
- Số lượng (Y): là một số nguyên dương ứng với số lượng sản phẩm khách hàng mua trong ngày hôm đó, tối thiểu 1 sản phẩm và tối đa 100 sản phẩm.
Miền giá trị: Y thuộc \mathbb{Z} , Y thuộc [1;100]
- Năm thành viên(Z): là một số nguyên dương ứng với năm mà khách hàng đăng ký thành viên, khách hàng khi mua hàng đều sẽ được đăng ký thành viên, nên số này luôn xác định và có giá trị trong khoảng từ năm thành lập công ty cho đến thời điểm khuyến mãi.
Miền giá trị: Z thuộc \mathbb{Z} , Z thuộc [2011;2021]

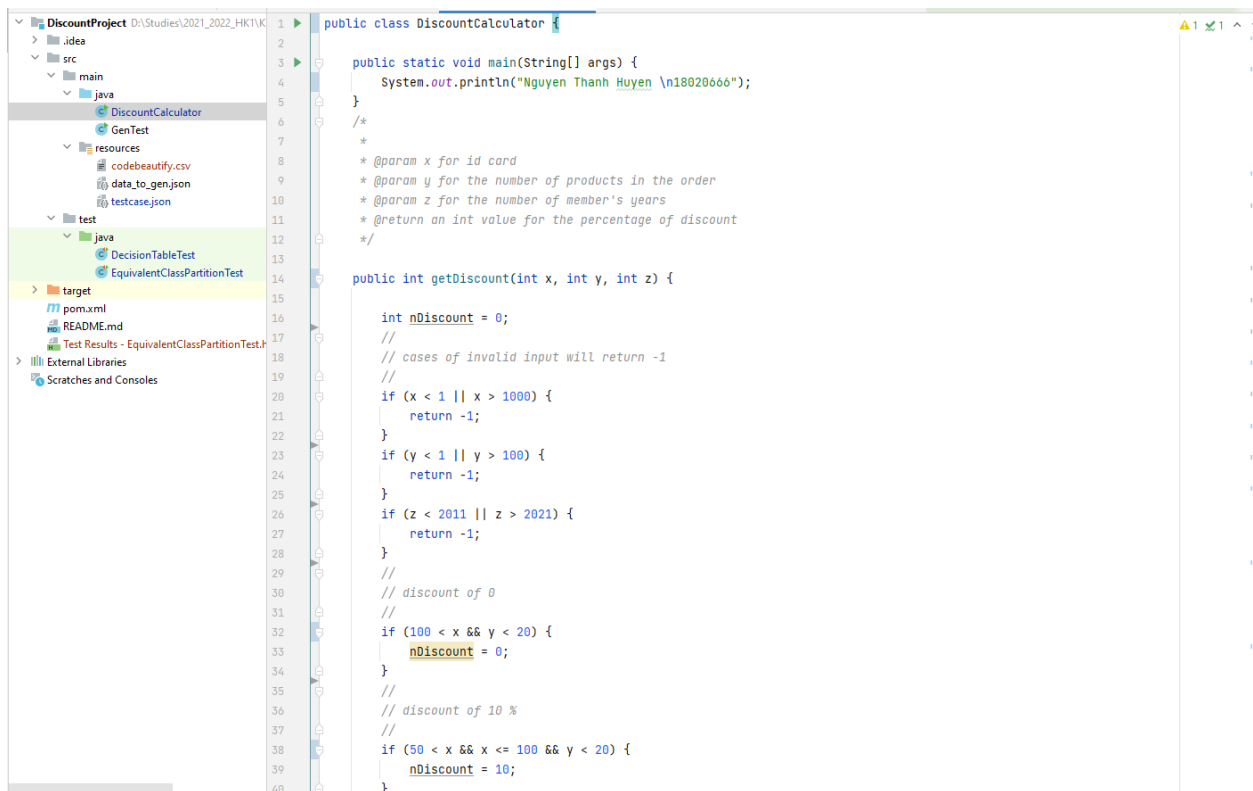
Đầu ra:

Mức khuyến mãi được áp dụng cho đơn hàng đó của khách hàng, có giá trị từ 0 – 50 (đơn vị %) nếu hợp lệ và trả về -1 nếu không hợp lệ.

2. Mã nguồn chương trình:

Mã nguồn cần kiểm thử là hàm tính toán phần trăm khuyến mãi, được lưu trong src/main/java/DiscountCalculator.java với hàm

```
DiscountCalculator.getDiscount(int x, int y, int z)
```



```
1 public class DiscountCalculator {
2
3     public static void main(String[] args) {
4         System.out.println("Nguyen Thanh Huyen \n18020660");
5     }
6
7     /*
8      * @param x for id card
9      * @param y for the number of products in the order
10     * @param z for the number of member's years
11     * @return an int value for the percentage of discount
12     */
13
14     public int getDiscount(int x, int y, int z) {
15
16         int nDiscount = 0;
17         //
18         // cases of invalid input will return -1
19         //
20         if (x < 1 || x > 1000) {
21             return -1;
22         }
23         if (y < 1 || y > 100) {
24             return -1;
25         }
26         if (z < 2011 || z > 2021) {
27             return -1;
28         }
29         //
30         // discount of 0
31         //
32         if (100 < x && y < 20) {
33             nDiscount = 0;
34         }
35         //
36         // discount of 10 %
37         //
38         if (50 < x && x <= 100 && y < 20) {
39             nDiscount = 10;
40         }
41     }
42 }
```



```
33         nDiscount = 0;
34     }
35     //
36     // discount of 10 %
37     //
38     if (50 < x && x <= 100 && y < 20) {
39         nDiscount = 10;
40     }
41     //
42     // discount of 15%
43     //
44     if (100 < x && 20 <= y) {
45         nDiscount = 15;
46     }
47
48     //
49     // discount of 25%
50     //
51     if ((x <= 50) || (x <= 100 && 20 <= y)) {
52         nDiscount = 25;
53     }
54
55     //
56     // bonus 25% discount for gold member
57     //
58
59     if (z <= 2018) {
60         nDiscount += 25;
61     }
62
63     return nDiscount;
64 }
65 }
```

3. Kiểm thử lớp tương đương:

Phân tích

Ứng với đầu vào, ta phân hoạch miền giá trị lần lượt thành các miền:

$$\begin{aligned} X1 &= [1; 50] \\ X2 &= (50; 100] \\ X3 &= (100; 1000] \\ X4 &= (-\infty; 1) \cup (1000; +\infty) \end{aligned}$$

$$\begin{aligned} Y1 &= [1; 20) \\ Y2 &= [20; 100] \\ Y3 &= (-\infty; 1) \cup (100; +\infty) \end{aligned}$$

$$\begin{aligned} Z1 &= [2011; 2018] \\ Z2 &= [2019; 2021] \\ Z3 &= (-\infty; 2011) \cup (2021; +\infty) \end{aligned}$$

Ứng với mỗi miền, chọn một giá trị đại diện:

X1: 25; X2: 75; X3: 500; X4: 1500

Y1: 9; Y2: 66; Y3: -8

Z1: 2015; Z2: 2020; Z3: 2023

Các ca kiểm thử

Sử dụng kiểm thử lớp tương đương mạnh, do đó ta có 36 ca kiểm thử

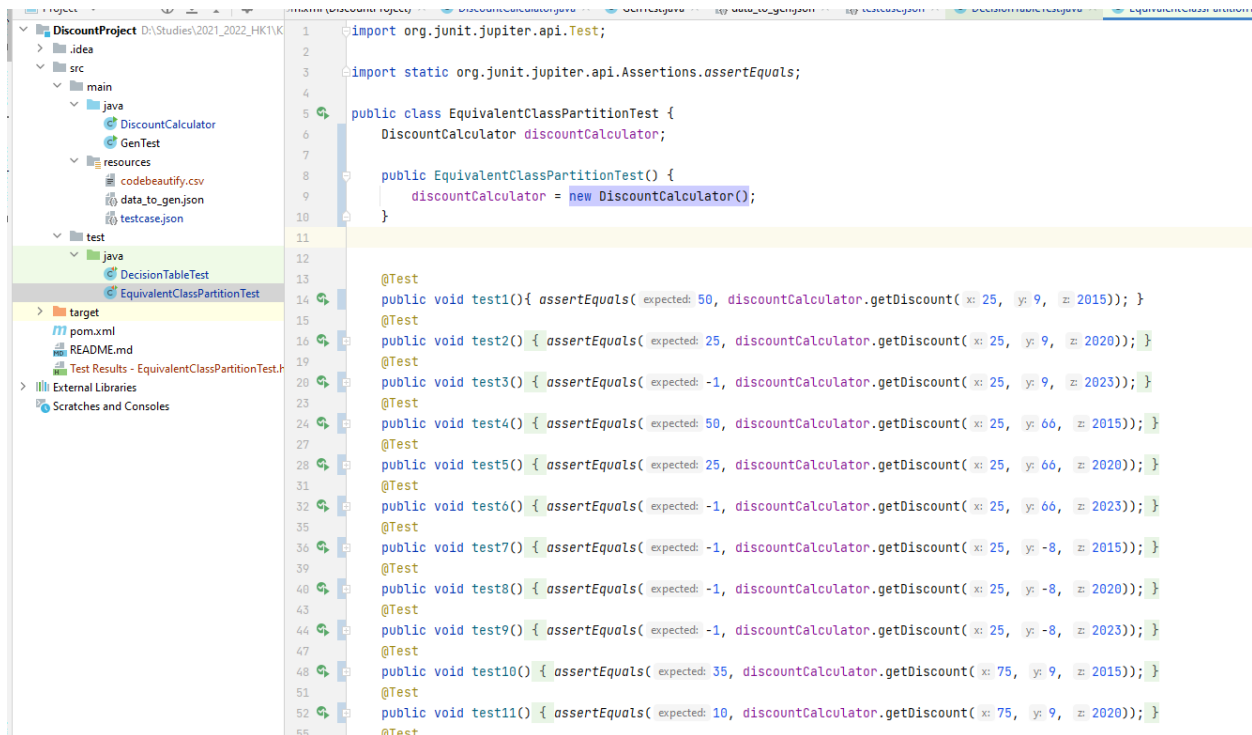
STT	Đầu vào			Đầu ra mong đợi (EO)	Đầu ra thực (AO)
	Mã phiếu	Số lượng	Năm		
1	25	9	2015	50	50
2	25	9	2020	25	25
3	25	9	2023	-1	-1
4	25	66	2015	50	50
5	25	66	2020	25	25

6	25	66	2023	-1	-1
7	25	-8	2015	-1	-1
8	25	-8	2020	-1	-1
9	25	-8	2023	-1	-1
10	75	9	2015	35	35
11	75	9	2020	10	10
12	75	9	2023	-1	-1
13	75	66	2015	50	50
14	75	66	2020	25	25
15	75	66	2023	-1	-1
16	75	-8	2015	-1	-1
17	75	-8	2020	-1	-1
18	75	-8	2023	-1	-1
19	500	9	2015	25	25
20	500	9	2020	0	0
21	500	9	2023	-1	-1
22	500	66	2015	40	40
23	500	66	2020	15	15
24	500	66	2023	-1	-1
25	500	-8	2015	-1	-1
26	500	-8	2020	-1	-1
27	500	-8	2023	-1	-1

28	1500	9	2015	-1	-1
29	1500	9	2020	-1	-1
30	1500	9	2023	-1	-1
31	1500	66	2015	-1	-1
32	1500	66	2020	-1	-1
33	1500	66	2023	-1	-1
34	1500	-8	2015	-1	-1
35	1500	-8	2020	-1	-1
36	1500	-8	2023	-1	-1

Thực thi kiểm thử

Thực hiện kiểm thử sử dụng Junit, kết quả trả về thành công ở 36 testcase đã chuẩn bị ở trên. Kết quả kiểm thử được lưu trong file Test Results – EquivalentClassPartition.html trong project (trên github).



The screenshot shows an IDE with the project structure on the left and the source code of the `EquivalentClassPartitionTest` class in the center. The project structure includes a `src` directory with `main` and `test` packages. The `test` package contains `DecisionTableTest` and `EquivalentClassPartitionTest`. The source code of `EquivalentClassPartitionTest` is as follows:

```

1  import org.junit.jupiter.api.Test;
2
3  import static org.junit.jupiter.api.Assertions.assertEquals;
4
5  public class EquivalentClassPartitionTest {
6      DiscountCalculator discountCalculator;
7
8      public EquivalentClassPartitionTest() {
9          discountCalculator = new DiscountCalculator();
10     }
11
12
13     @Test
14     public void test1(){ assertEquals( expected: 50, discountCalculator.getDiscount( x: 25, y: 9, z: 2015)); }
15
16     @Test
17     public void test2(){ assertEquals( expected: 25, discountCalculator.getDiscount( x: 25, y: 9, z: 2020)); }
18
19     @Test
20     public void test3(){ assertEquals( expected: -1, discountCalculator.getDiscount( x: 25, y: 9, z: 2023)); }
21
22     @Test
23     public void test4(){ assertEquals( expected: 50, discountCalculator.getDiscount( x: 25, y: 66, z: 2015)); }
24
25     @Test
26     public void test5(){ assertEquals( expected: 25, discountCalculator.getDiscount( x: 25, y: 66, z: 2020)); }
27
28     @Test
29     public void test6(){ assertEquals( expected: -1, discountCalculator.getDiscount( x: 25, y: 66, z: 2023)); }
30
31     @Test
32     public void test7(){ assertEquals( expected: -1, discountCalculator.getDiscount( x: 25, y: -8, z: 2015)); }
33
34     @Test
35     public void test8(){ assertEquals( expected: -1, discountCalculator.getDiscount( x: 25, y: -8, z: 2020)); }
36
37     @Test
38     public void test9(){ assertEquals( expected: -1, discountCalculator.getDiscount( x: 25, y: -8, z: 2023)); }
39
40     @Test
41     public void test10(){ assertEquals( expected: 35, discountCalculator.getDiscount( x: 75, y: 9, z: 2015)); }
42
43     @Test
44     public void test11(){ assertEquals( expected: 10, discountCalculator.getDiscount( x: 75, y: 9, z: 2020)); }
45
46     @Test
47

```

EquivalentClassPartitionTest: 36 total, 36 passed

76 ms

	Collapse	Expand
test26()	passed	2 ms
test27()	passed	2 ms
test28()	passed	1 ms
test29()	passed	1 ms
test30()		passed
test31()	passed	1 ms
test32()	passed	1 ms
test33()	passed	1 ms
test34()	passed	1 ms
test35()	passed	2 ms
test36()		passed
test1()	passed	1 ms
test2()	passed	1 ms
test3()	passed	1 ms
test4()	passed	1 ms
test5()	passed	1 ms
test6()		passed
test7()	passed	1 ms
test8()	passed	1 ms
test9()	passed	1 ms

4. Kiểm thử bảng quyết định:

Phân tích

Bảng quyết định được xây dựng với các điều kiện và hành động như hình:


	Condition/Action	1	2	3	4	5	6	7	8	9	10	11
	Mã phiếu <= 0	T	T	T	T	T	T	T	T	T	T	T
	1<= Mã Phiếu <= 50	-	-	-	-	-	-	-	-	-	-	-
	51<= Mã phiếu <=100	-	-	-	-	-	-	-	-	-	-	-
	101<= Mã phiếu <= 1000	-	-	-	-	-	-	-	-	-	-	-
	Mã phiếu >= 1001	-	-	-	-	-	-	-	-	-	-	-
	Số lượng <= 0	T	T	T	T	F	F	F	F	F	F	F
	1<= Số lượng <= 19	-	-	-	-	T	T	T	T	F	F	F
	20<= Số lượng <=100	-	-	-	-	-	-	-	-	T	T	T
	Số lượng >= 101	-	-	-	-	-	-	-	-	-	-	-
	Năm thành viên <= 2010	T	F	F	F	T	F	F	F	T	F	F
	2011 <= Năm thành viên <= 2018	-	T	F	F	-	T	F	F	-	T	F
	2019 <= Năm thành viên <= 2021	-	-	T	F	-	-	T	F	-	-	T
Condition	Năm thành viên >= 2022	-	-	-	T	-	-	-	T	-	-	-
	Dữ liệu đầu vào không hợp lệ	x	x	x	x	x	x	x	x	x	x	x
	Không áp mã khuyến mãi											
	Mã giảm giá 10%											
	Mã giảm giá 15%											
	Mã giảm giá 25%											
Hành động	Mã giảm giá bổ sung + 25%											

Sau khi xây dựng, bảng quyết định có tổng cộng 80 rules – 80 cột. Mỗi cột ứng với một bộ testcase.

Để dễ dàng chuyển đổi thông tin các bộ testcase này sang dạng JSON (phục vụ cho việc tự động sinh các ca kiểm thử), em chuyển các cột về 80 hàng, mỗi hàng ứng với một cột Rule trên Decision Table.

1	T	-	-	-	-	T	-	-	-	T	-	-	-	x
2	T	-	-	-	-	T	-	-	-	F	T	-	-	x
3	T	-	-	-	-	T	-	-	-	F	F	T	-	x
4	T	-	-	-	-	T	-	-	-	F	F	F	T	x
5	T	-	-	-	-	F	T	-	-	T	-	-	-	x
6	T	-	-	-	-	F	T	-	-	F	T	-	-	x
7	T	-	-	-	-	F	T	-	-	F	F	T	-	x
8	T	-	-	-	-	F	T	-	-	F	F	F	T	x
9	T	-	-	-	-	F	F	T	-	T	-	-	-	x
10	T	-	-	-	-	F	F	T	-	F	T	-	-	x
11	T	-	-	-	-	F	F	T	-	F	F	T	-	x
12	T	-	-	-	-	F	F	T	-	F	F	F	T	x
13	T	-	-	-	-	F	F	F	T	T	-	-	-	x
14	T	-	-	-	-	F	F	F	T	F	T	-	-	x
15	T	-	-	-	-	F	F	F	T	F	F	T	-	x
16	T	-	-	-	-	F	F	F	T	F	F	F	T	x
17	F	T	-	-	-	T	-	-	-	T	-	-	-	x
18	F	T	-	-	-	T	-	-	-	F	T	-	-	x
19	F	T	-	-	-	T	-	-	-	F	F	T	-	x
20	F	T	-	-	-	T	-	-	-	F	F	F	T	x
21	F	T	-	-	-	F	T	-	-	T	-	-	-	x
22	F	T	-	-	-	F	T	-	-	F	T	-	-	x
23	F	T	-	-	-	F	T	-	-	F	F	T	-	x
24	F	T	-	-	-	F	T	-	-	F	F	F	T	x
25	F	T	-	-	-	F	F	T	-	T	-	-	-	x
26	F	T	-	-	-	F	F	T	-	F	T	-	-	x
27	F	T	-	-	-	F	F	T	-	F	F	T	-	x
28	F	T	-	-	-	F	F	T	-	F	F	F	T	x
29	F	T	-	-	-	F	F	T	-	T	-	-	-	x

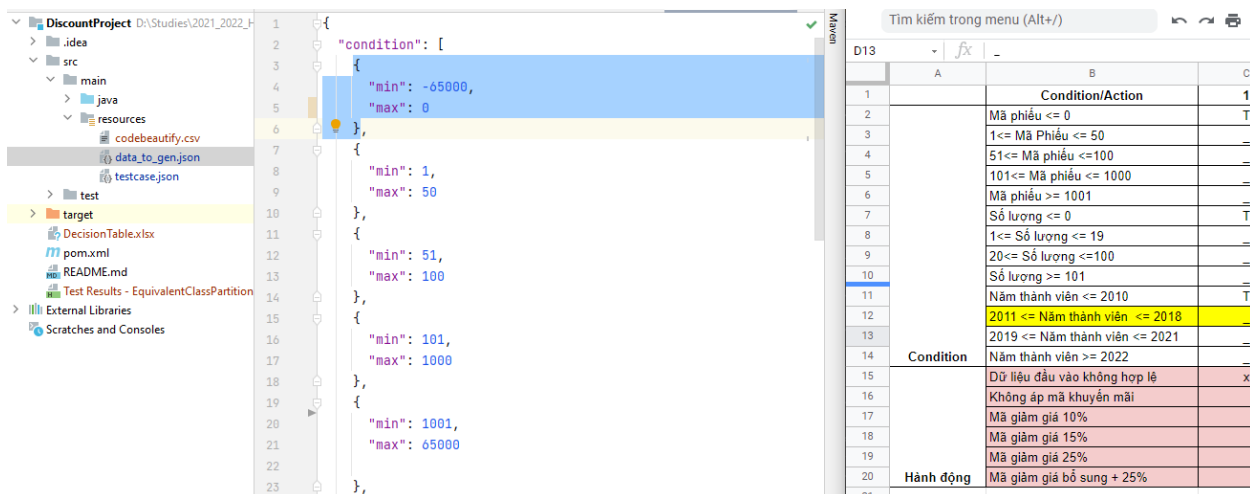
Bảng quyết định được lưu file DecisonTable.xlsx trên đường dẫn git tới mã nguồn chương trình.

	huyenlovemath add Decision Table	9678534 now	9 commits
idea	fix gentest	21 hours ago	
src	done decision table test	21 hours ago	
target	done decision table test	21 hours ago	
DecisionTable.xlsx	add Decision Table	now	
README.md	create readme	20 hours ago	
pom.xml	update run test in DecisionTableTest	21 hours ago	

Xây dựng tự động các ca kiểm thử

Từ bảng quyết định, xây dựng một file json (src/main/resources/data_to_gen.json) chứa đối tượng thông tin về bảng có cấu trúc như sau:

- “condition”: ứng với một mảng lưu các cặp giá trị (min;max) cho điều kiện kiểm thử. Ở đây, có 13 điều kiện, thì mảng này lưu 13 cặp (min;max). Với các điều kiện chỉ có min, thì em giới hạn max là 65000; các điều kiện chỉ có max, thì em giới hạn min là -65000. Các con số này có thể được thay đổi bằng việc sửa trong file data_to_gen.json này



The screenshot shows an IDE with a project named 'DiscountProject'. The file explorer on the left shows the project structure, including 'src/main/resources/data_to_gen.json'. The main editor displays the content of this JSON file, which defines conditions for a decision table. The conditions are defined as an array of objects, each with 'min' and 'max' values. The right side of the screenshot shows a spreadsheet titled 'DecisionTable.xlsx' with columns A, B, and C. The spreadsheet contains a decision table with 13 conditions and their corresponding actions.

Condition/Action	1
Mã phiếu <= 0	T
1<= Mã Phiếu <= 50	-
51<= Mã phiếu <=100	-
101<= Mã phiếu <= 1000	-
Mã phiếu >= 1001	-
Số lượng <= 0	T
1<= Số lượng <= 19	-
20<= Số lượng <=100	-
Số lượng >= 101	-
Năm thành viên <= 2010	T
2011 <= Năm thành viên <= 2018	-
2019 <= Năm thành viên <= 2021	-
Năm thành viên >= 2022	-
Dữ liệu đầu vào không hợp lệ	x
Không áp mã khuyến mãi	-
Mã giảm giá 10%	-
Mã giảm giá 15%	-
Mã giảm giá 25%	-
Hành động	Mã giảm giá bổ sung + 25%

- “action”: là mảng lưu giá trị phần trăm giảm giá ứng với từng action

- Hàm `GetRandomData(int min, int max)`: lấy ngẫu nhiên một giá trị trong đoạn từ min đến max

```
public static int GetRandomData(int min, int max)
{
    int upperbound = max - min + 1;
    Random rand = new Random();

    // nextInt(upperbound):
    // generates random numbers in the range 0 to upperbound-1.
    return min + rand.nextInt(upperbound);
}
```

- Hàm `GetJsonData(String json_file_path)`: đọc dữ liệu từ file `data_to_gen.json` đã được chuẩn bị từ bước trên.

```
} public static JSONObject GetJsonData(String json_file_path) throws IOException {
    FileInputStream json_file_in;

    json_file_in = new FileInputStream(json_file_path);
    String json_data = IOUtils.toString(json_file_in, encoding: "UTF-8");

    return new JSONObject(json_data);
}
```

- Hàm `GenTestcases()`: sinh các bộ testcase, mỗi testcase gồm 3 giá trị truyền vào hàm cần kiểm thử và một giá trị Expected Output tương ứng.

```
public static void GenTestCases() throws IOException {
    JSONObject data_object;
    JSONArray condition_array, action_array, rule_array, rule_set, testcase, result;
    int n_conditions, n_actions, n_tests;
    int min, max, expected_value;

    //preparation
    data_object = GetJsonData(data_file_path);
    condition_array = data_object.getJSONArray( key: "condition");
    action_array = data_object.getJSONArray( key: "action");
    rule_array = data_object.getJSONArray( key: "rule");
    n_conditions = condition_array.length();
    n_actions = action_array.length();
    n_tests = rule_array.length();

    // array to store all testcases
    result = new JSONArray();

    //loop for all set of value in rule_array
    for (int i_test = 0; i_test < n_tests; i_test++)
    {
```

- `WriteTestcaseToFile(JSONArray testcases)`: lưu các bộ testcase vừa sinh ra file `src/main/resources/testcase.json`.

```

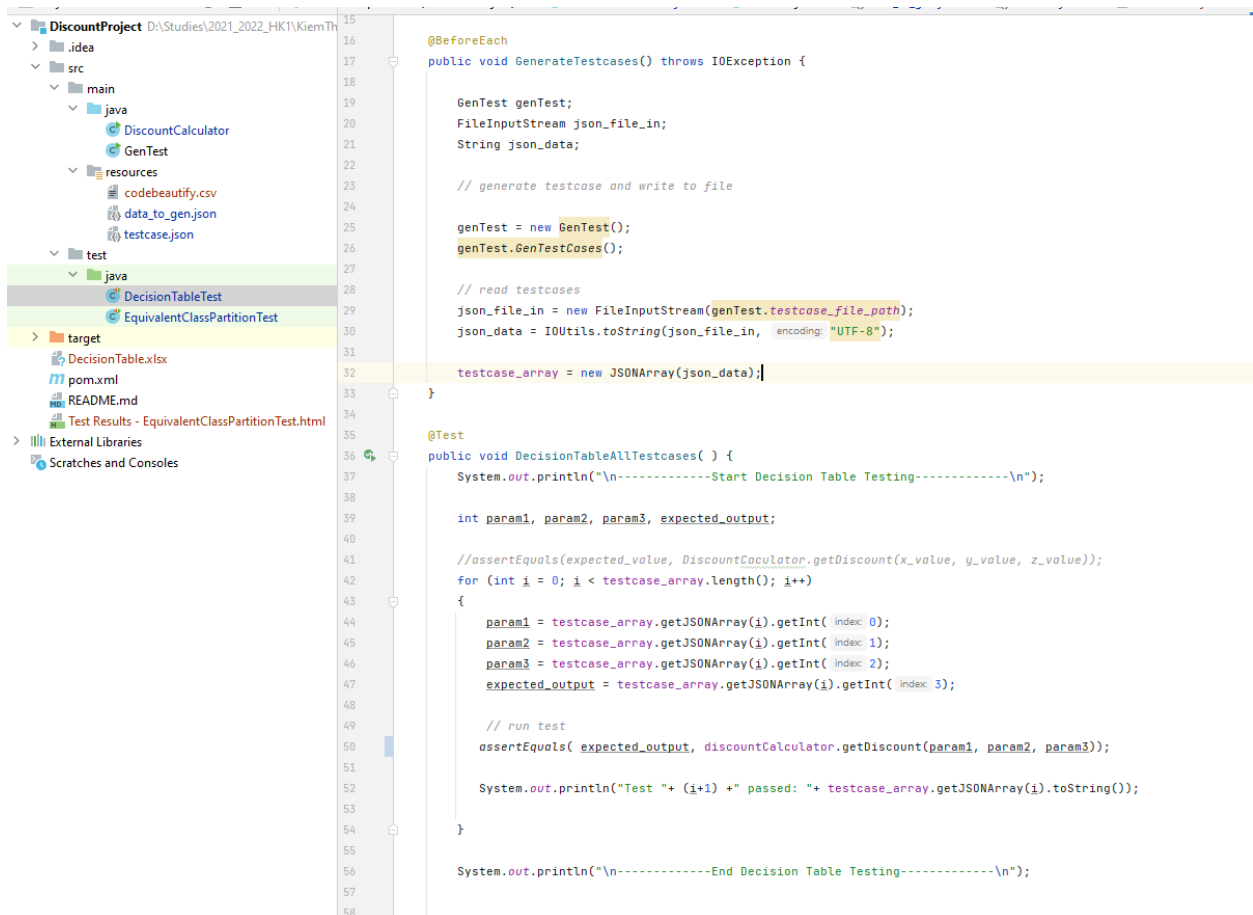
    public static void WriteTestCaseToFile(JSONArray testcases) throws IOException {

        FileWriter fileWriter = new FileWriter(testcase_file_path);

        fileWriter.write(testcases.toString());
        fileWriter.close();
    }
}

```

Tiến hành viết file kiểm thử src/test/java/DecisionTableTest.java thực hiện gọi đến các hàm trong GenTest.java để sinh các bộ giá trị truyền vào hàm discountCalculator.getDiscount() để kiểm thử.



The screenshot shows an IDE with a project named 'DiscountProject'. The project structure on the left includes 'src/main/java' with 'DiscountCalculator' and 'GenTest', and 'src/test/java' with 'DecisionTableTest' and 'EquivalentClassPartitionTest'. The main editor displays the code for 'DecisionTableTest.java'.

```

15
16
17 @BeforeEach
18 public void GenerateTestcases() throws IOException {
19
20     GenTest genTest;
21     FileInputStream json_file_in;
22     String json_data;
23
24     // generate testcase and write to file
25
26     genTest = new GenTest();
27     genTest.GenTestCases();
28
29     // read testcases
30     json_file_in = new FileInputStream(genTest.testcase_file_path);
31     json_data = IOUtils.toString(json_file_in, encoding: "UTF-8");
32
33     testcase_array = new JSONArray(json_data);
34 }
35
36 @Test
37 public void DecisionTableAllTestcases() {
38     System.out.println("\n-----Start Decision Table Testing-----\n");
39
40     int param1, param2, param3, expected_output;
41
42     //assertEquals(expected_value, DiscountCalculator.getDiscount(x_value, y_value, z_value));
43     for (int i = 0; i < testcase_array.length(); i++)
44     {
45         param1 = testcase_array.getJSONArray(i).getInt( index: 0);
46         param2 = testcase_array.getJSONArray(i).getInt( index: 1);
47         param3 = testcase_array.getJSONArray(i).getInt( index: 2);
48         expected_output = testcase_array.getJSONArray(i).getInt( index: 3);
49
50         // run test
51         assertEquals( expected_output, discountCalculator.getDiscount(param1, param2, param3));
52
53         System.out.println("Test " + (i+1) + " passed: " + testcase_array.getJSONArray(i).toString());
54     }
55
56     System.out.println("\n-----End Decision Table Testing-----\n");
57
58

```

Sử dụng junit để kiểm tra giá trị đầu ra có đúng với đầu ra mong muốn đã xây dựng hay không. Chạy file kiểm thử trên, được kết quả thành công ở 80 bộ testcase

```
✓ Tests passed: 1 of 1 test - 193 ms
Test Results 193 ms
  ✓ DecisionTableTest 193 ms
    ✓ DecisionTableAllTestcases() 193 ms
-----Start Decision Table Testing-----
Test 1 passed: [-1801,-30185,-64050,-1]
Test 2 passed: [-54810,-19031,2017,-1]
Test 3 passed: [-20118,-50757,2020,-1]
Test 4 passed: [-42400,-61074,61386,-1]
Test 5 passed: [-56248,15,-14212,-1]
Test 6 passed: [-11356,16,2012,-1]
Test 7 passed: [-7532,13,2019,-1]
Test 8 passed: [-53030,12,27165,-1]
Test 9 passed: [-55588,80,-59195,-1]
Test 10 passed: [-3430,35,2013,-1]
Test 11 passed: [-54685,48,2019,-1]
Test 12 passed: [-48421,36,30231,-1]
Test 13 passed: [-59348,24883,-15794,-1]
Test 14 passed: [-54627,35399,2014,-1]
Test 15 passed: [-18055,22315,2020,-1]
Test 16 passed: [-45056,30014,22467,-1]
Test 17 passed: [50,-3494,-5042,-1]
Test 18 passed: [34,-2913,2017,-1]
Test 19 passed: [29,-12183,2019,-1]
Test 20 passed: [24,-56801,17119,-1]
Test 21 passed: [13,1,-23286,-1]
Test 22 passed: [40,5,2017,50]
Test 23 passed: [50,14,2020,25]
Test 24 passed: [39,8,6103,-1]
Test 25 passed: [28,60,-26642,-1]
Test 26 passed: [3,73,2012,50]
Test 27 passed: [42,82,2020,25]
Test 28 passed: [43,97,29061,-1]
Test 29 passed: [21,41532,-12564,-1]
Test 30 passed: [20,59001,2016,-1]
Test 31 passed: [33,4357,2018,-1]
```

```
✓ Tests passed: 1 of 1 test - 193 ms
Test Results 193 ms
  ✓ DecisionTableTest 193 ms
    ✓ DecisionTableAllTestcases() 193 ms
Test 54 passed: [108,7,2011,25]
Test 55 passed: [764,1,2020,0]
Test 56 passed: [664,1,2947,-1]
Test 57 passed: [586,88,-2145,-1]
Test 58 passed: [593,96,2018,40]
Test 59 passed: [902,21,2021,15]
Test 60 passed: [971,34,16310,-1]
Test 61 passed: [662,30259,-27815,-1]
Test 62 passed: [560,38045,2015,-1]
Test 63 passed: [502,28447,2021,-1]
Test 64 passed: [521,5550,44535,-1]
Test 65 passed: [48695,-43648,-19230,-1]
Test 66 passed: [59461,-15604,2012,-1]
Test 67 passed: [52636,-41084,2020,-1]
Test 68 passed: [19061,-64931,52313,-1]
Test 69 passed: [25520,4,-8097,-1]
Test 70 passed: [6268,13,2013,-1]
Test 71 passed: [21831,14,2020,-1]
Test 72 passed: [24907,7,5562,-1]
Test 73 passed: [36795,76,-52122,-1]
Test 74 passed: [61956,77,2013,-1]
Test 75 passed: [61696,56,2020,-1]
Test 76 passed: [28309,53,43985,-1]
Test 77 passed: [55373,22219,-60172,-1]
Test 78 passed: [27240,56004,2015,-1]
Test 79 passed: [61085,8467,2019,-1]
Test 80 passed: [28691,24430,3052,-1]
-----End Decision Table Testing-----
Process finished with exit code 0
```

Chuyển file testcase.json sang định dạng csv (src/main/java/testcases_table.csv) để dễ dàng lấy được thông tin các ca kiểm thử, phục vụ cho việc viết báo cáo:

STT	Mã phiếu	Số lượng	Năm thành viên	Expected Output
1	-1801	-30185	-64050	-1
2	-54810	-19031	2017	-1
3	-20118	-50757	2020	-1
4	-42400	-61074	61386	-1
5	-56248	15	-14212	-1
6	-11356	16	2012	-1
7	-7532	13	2019	-1
8	-53030	12	27165	-1
9	-55588	80	-59195	-1
10	-3430	35	2013	-1
11	-54685	48	2019	-1
12	-48421	36	30231	-1
13	-59348	24883	-15794	-1
14	-54627	35399	2014	-1
15	-18055	22315	2020	-1
16	-45056	30014	22467	-1
17	50	-3494	-5042	-1
18	34	-2913	2017	-1
19	29	-12183	2019	-1
20	24	-56801	17119	-1
21	13	1	-23286	-1
22	40	5	2017	50
23	50	14	2020	25
24	39	8	6103	-1
25	28	60	-26642	-1
26	3	73	2012	50
27	42	82	2020	25
28	43	97	29061	-1
29	21	41532	-12564	-1
30	20	59001	2016	-1
31	33	63547	2019	-1
32	3	61476	57882	-1
33	81	-35425	-43239	-1
34	86	-730	2017	-1
35	68	-18722	2021	-1
36	76	-23032	41974	-1
37	97	10	-22011	-1
38	89	11	2011	35
39	61	11	2019	10
40	95	8	31473	-1

41	93	46	-26545	-1
42	87	70	2018	50
43	56	76	2021	25
44	62	27	36290	-1
45	53	64921	-10394	-1
46	63	13705	2017	-1
47	60	56731	2021	-1
48	57	51839	19073	-1
49	314	-33238	-49753	-1
50	331	-64851	2014	-1
51	448	-6010	2019	-1
52	482	-16421	51497	-1
53	276	15	-7354	-1
54	108	7	2011	25
55	764	1	2020	0
56	664	1	2947	-1
57	586	88	-2145	-1
58	593	96	2018	40
59	902	21	2021	15
60	971	34	16310	-1
61	662	30259	-27815	-1
62	560	38045	2015	-1
63	502	20447	2021	-1
64	521	5550	44535	-1
65	48695	-43648	-19230	-1
66	59461	-15604	2012	-1
67	52636	-41084	2020	-1
68	19061	-64931	52313	-1
69	25520	4	-8097	-1
70	6268	13	2013	-1
71	21831	14	2020	-1
72	24907	7	5562	-1
73	36795	76	-52122	-1
74	61956	77	2013	-1
75	61696	56	2020	-1
76	28309	53	43985	-1
77	55373	22219	-60172	-1
78	27240	56004	2015	-1
79	61085	8467	2019	-1
80	28691	24430	3052	-1

Thực thi kiểm thử

Mỗi lần thực thi DecisionTableTest.java, sẽ có một tập các ca kiểm thử được tự động sinh ra, kết quả cho lần chạy ứng với bảng các ca kiểm thử trên được lưu trong file Test Results – DecisionTableTest.html trong Project.

DecisionTableTest: 1 total, 1 passed193 ms

Collapse | Expand

-----Start Decision Table Testing-----
Test 1 passed: [-1801,-30185,-64050,-1]
Test 2 passed: [-54810,-19031,2017,-1]
Test 3 passed: [-20118,-50757,2020,-1]
Test 4 passed: [-42400,-81074,81388,-1]
Test 5 passed: [-58248,15,-14212,-1]
Test 6 passed: [-11358,16,2012,-1]
Test 7 passed: [-7532,13,2019,-1]
Test 8 passed: [-53030,12,27165,-1]
Test 9 passed: [-55588,80,-59195,-1]
Test 10 passed: [-3430,35,2013,-1]
Test 11 passed: [-54885,48,2019,-1]
Test 12 passed: [-48421,36,30231,-1]
Test 13 passed: [-59348,24883,-15794,-1]
Test 14 passed: [-54827,35399,2014,-1]
Test 15 passed: [-18055,22315,2020,-1]
Test 16 passed: [-48058,30014,22467,-1]
Test 17 passed: [50,-3494,-5042,-1]
Test 18 passed: [34,-2913,2017,-1]
Test 19 passed: [29,-12183,2019,-1]
Test 20 passed: [24,-56801,17119,-1]
Test 21 passed: [13,1,-23286,-1]
Test 22 passed: [40,5,2017,50]
Test 23 passed: [50,14,2020,25]
Test 24 passed: [39,8,8103,-1]
Test 25 passed: [28,60,-26642,-1]
Test 26 passed: [3,73,2012,50]
Test 27 passed: [42,82,2020,25]
Test 28 passed: [43,97,29081,-1]
Test 29 passed: [21,41532,-12584,-1]
Test 30 passed: [20,59001,2016,-1]
Test 31 passed: [33,83547,2019,-1]
Test 32 passed: [3,81478,57882,-1]
Test 33 passed: [81,-35425,-43239,-1]
Test 34 passed: [86,-730,2017,-1]
Test 35 passed: [68,-18722,2021,-1]
Test 36 passed: [76,-23032,41974,-1]
Test 37 passed: [97,10,-22011,-1]
Test 38 passed: [89,11,2011,35]
Test 39 passed: [81,11,2019,10]
Test 40 passed: [95,8,31473,-1]
Test 41 passed: [93,48,-26545,-1]
Test 42 passed: [87,70,2018,50]
Test 43 passed: [56,76,2021,25]
Test 44 passed: [82,27,36290,-1]
Test 45 passed: [53,64921,-10394,-1]
Test 46 passed: [83,13705,2017,-1]
Test 47 passed: [80,56731,2021,-1]
Test 48 passed: [57,51839,19073,-1]
Test 49 passed: [314,-33238,-49753,-1]
Test 50 passed: [331,-64851,2014,-1]
Test 51 passed: [448,-6010,2019,-1]
Test 52 passed: [482,-16421,51497,-1]
Test 53 passed: [278,15,-7354,-1]
Test 54 passed: [108,7,2011,25]
Test 55 passed: [764,1,2020,0]
Test 56 passed: [664,1,2947,-1]