EASTERN INTERNATIONAL UNIVERSITY

SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY

PROJECT CSE453

Boundary Value Testing

Student(s)

Bui Hoang Huy – 2031200010

Pham Minh Tuan – 1831200043

Supervisor(s)

Prof. Chandra Debnath Narayan

Binh Duong, 03, 2024

Abstract

In this testing project, we explore the boundary value method, a technique used to test software by examining the values at the boundaries of acceptable input ranges. We investigate how this method can help identify potential errors and vulnerabilities in software systems. By focusing on simple examples and practical applications, we aim to demonstrate the effectiveness of boundary value testing in improving software quality and reliability. This project provides insights into how software developers can leverage boundary value testing to enhance their testing strategies and ensure robustness in their products.

Acknowledgement

We would like to express our sincere gratitude to everyone who contributed to the completion of this testing project. Special thanks to our supervisor for their guidance and support throughout the project. We also extend our appreciation to our classmates and friends for their valuable input and encouragement. Additionally, we acknowledge the resources and materials provided by our institution, which greatly aided our research and analysis.

Table of Contents

Chapter 1: Project Details

1.1 Problem 1

a. Manually Derive the sets S1, S2, S3 for F (X1, X2, X3)

5 Unit:

- S1 = {<X_{1min}, X_{2nom}, X_{3nom}>, <X_{1min+}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1max-}, X_{2nom}, X_{3nom}>, <X_{1max}, X_{2nom}, X_{3nom}>}

- S2 = {<X_{1nom}, X_{2min}, X_{3nom}>, <X_{1nom}, X_{2min+}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2max-}, X_{3nom}>, <X_{1nom}, X_{2max}, X_{3nom}>}

- S3 = {<X_{1nom}, X_{2nom}, X_{3min}>, <X_{1nom}, X_{2nom}, X_{3min+}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3max-}>, <X_{1nom}, X_{2nom}, X_{3max}>}

7 Unit:

- S1 = {<X_{1min-}, X_{2nom}, X_{3nom}>, <X_{1min}, X_{2nom}, X_{3nom}>, <X_{1min+}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1max-}, X_{2nom}, X_{3nom}>, <X_{1max}, X_{2nom}, X_{3nom}>, {<X_{1max+}, X_{2nom}, X_{3nom}>}

- S2 = {<X_{1nom}, X_{2min-}, X_{3nom}>, <X_{1nom}, X_{2min}, X_{3nom}>, <X_{1nom}, X_{2min+}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2max-}, X_{3nom}>, <X_{1nom}, X_{2max}, X_{3nom}>, <X_{1nom}, X_{2max+}, X_{3nom}>}

- S3 = {<X_{1nom}, X_{2nom}, X_{3min-}>, <X_{1nom}, X_{2nom}, X_{3min}>, <X_{1nom}, X_{2nom}, X_{3min+}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3max-}>, <X_{1nom}, X_{2nom}, X_{3max}>, <X_{1nom}, X_{2nom}, X_{3max+}>}

b. How many total elements are presented in each set?

5 Unit: S1 = 5; S2 = 5; S3 = 5

7 Unit: S1 = 7, S2 = 7; S3 = 7

c. Manually Compute S1 U S2 U S3.

5 Unit: S1 U S2 U S3 = {<X_{1min}, X_{2nom}, X_{3nom}>, <X_{1min+}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1max-}, X_{2nom}, X_{3nom}>, <X_{1max}, X_{2nom}, X_{3nom}>} U { <X_{1nom}, X_{2min}, X_{3nom}>, <X_{1nom}, X_{2min+}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2max-}, X_{3nom}>, <X_{1nom}, X_{2max}, X_{3nom}> } U { <X_{1nom}, X_{2nom}, X_{3min}>, <X_{1nom}, X_{2nom}, X_{3min+}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3max-}>,<X_{1nom}, X_{2nom}, X_{3max}>} = {<X_{1min}, X_{2nom}, X_{3nom}>, <X_{1min+}, X_{2nom}, X_{3nom}>, <X_{1max-}, X_{2nom}, X_{3nom}>, <X_{1max}, X_{2nom}, X_{3nom}>,

<X_{1nom}, X_{2min}, X_{3nom}>, <X_{1nom}, X_{2min+}, X_{3nom}>, <X_{1nom}, X_{2max-}, X_{3nom}>, <X_{1nom}, X_{2max}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3min}>, <X_{1nom}, X_{2nom}, X_{3min+}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3max-}>,<X_{1nom}, X_{2nom}, X_{3max}>}

7 Unit: S1 U S2 U S3 = {<X_{1min-}, X_{2nom}, X_{3nom}>, <X_{1min}, X_{2nom}, X_{3nom}>, <X_{1min+}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1max-}, X_{2nom}, X_{3nom}>, <X_{1max}, X_{2nom}, X_{3nom}>, {<X_{1max+}, X_{2nom}, X_{3nom}>} U { <X_{1nom}, X_{2min-}, X_{3nom}>, <X_{1nom}, X_{2min}, X_{3nom}>, <X_{1nom}, X_{2min+}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2max-}, X_{3nom}>, <X_{1nom}, X_{2max}, X_{3nom}>, <X_{1nom}, X_{2max+}, X_{3nom}> } U {<X_{1nom}, X_{2nom}, X_{3min-}>, <X_{1nom}, X_{2nom}, X_{3min}>, <X_{1nom}, X_{2nom}, X_{3min+}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3max-}>,<X_{1nom}, X_{2nom}, X_{3max}>, <X_{1nom}, X_{2nom}, X_{3max+}>} = {<X_{1min-}, X_{2nom}, X_{3nom}>, <X_{1min}, X_{2nom}, X_{3nom}>, <X_{1min+}, X_{2nom}, X_{3nom}>, <X_{1max-}, X_{2nom}, X_{3nom}>, <X_{1max}, X_{2nom}, X_{3nom}>, <X_{1max+}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2min-}, X_{3nom}>, <X_{1nom}, X_{2min}, X_{3nom}>, <X_{1nom}, X_{2min+}, X_{3nom}>, <X_{1nom}, X_{2max-}, X_{3nom}>, <X_{1nom}, X_{2max}, X_{3nom}>, <X_{1nom}, X_{2max+}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3min-}>, <X_{1nom}, X_{2nom}, X_{3min}>, <X_{1nom}, X_{2nom}, X_{3min+}>, <X_{1nom}, X_{2nom}, X_{3nom}>, <X_{1nom}, X_{2nom}, X_{3max-}>,<X_{1nom}, X_{2nom}, X_{3max}>, <X_{1nom}, X_{2nom}, X_{3max+}>}

d. How many **total** elements are presented in the Union?

5 unit: Total elements = 15

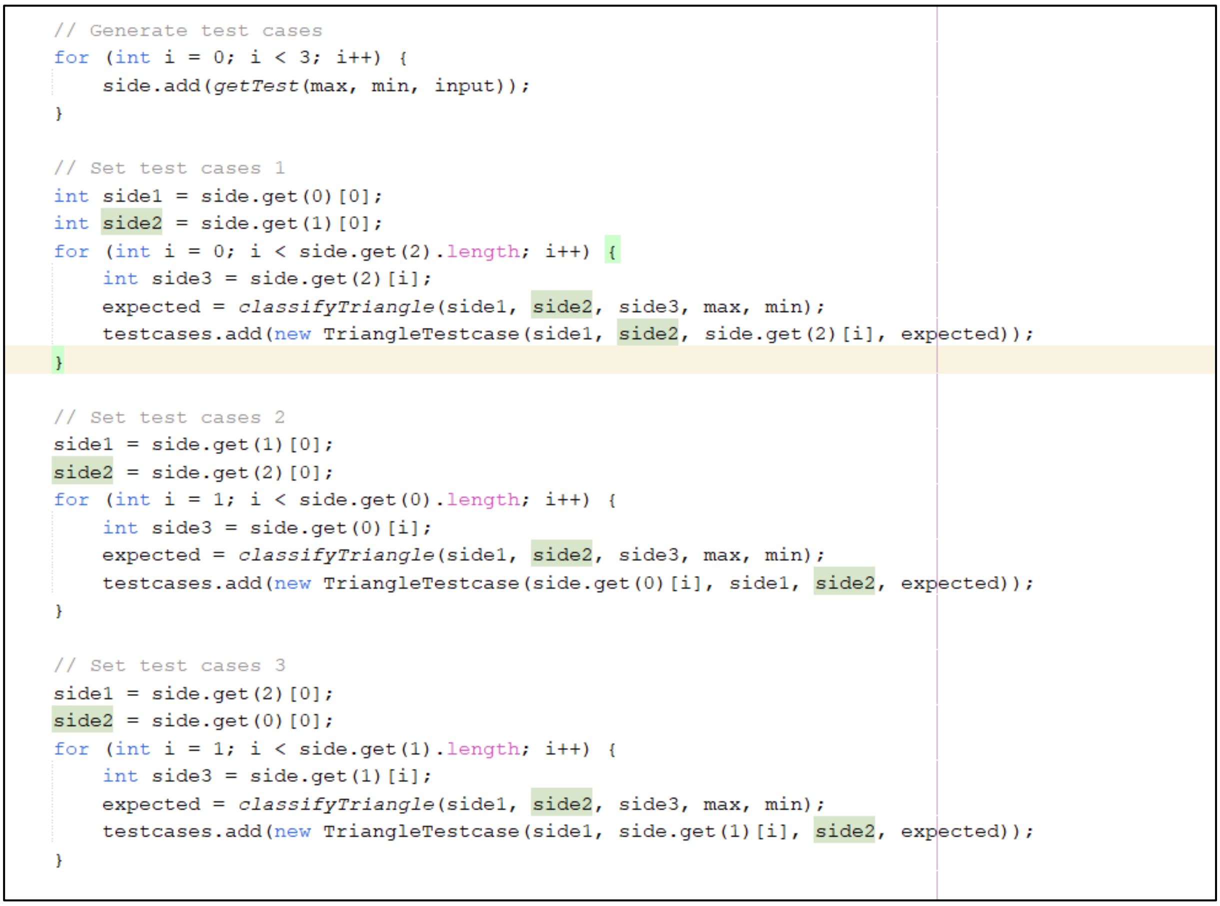
7 unit: Total elements = 21

e. How many **unique** elements are presented in the Union?

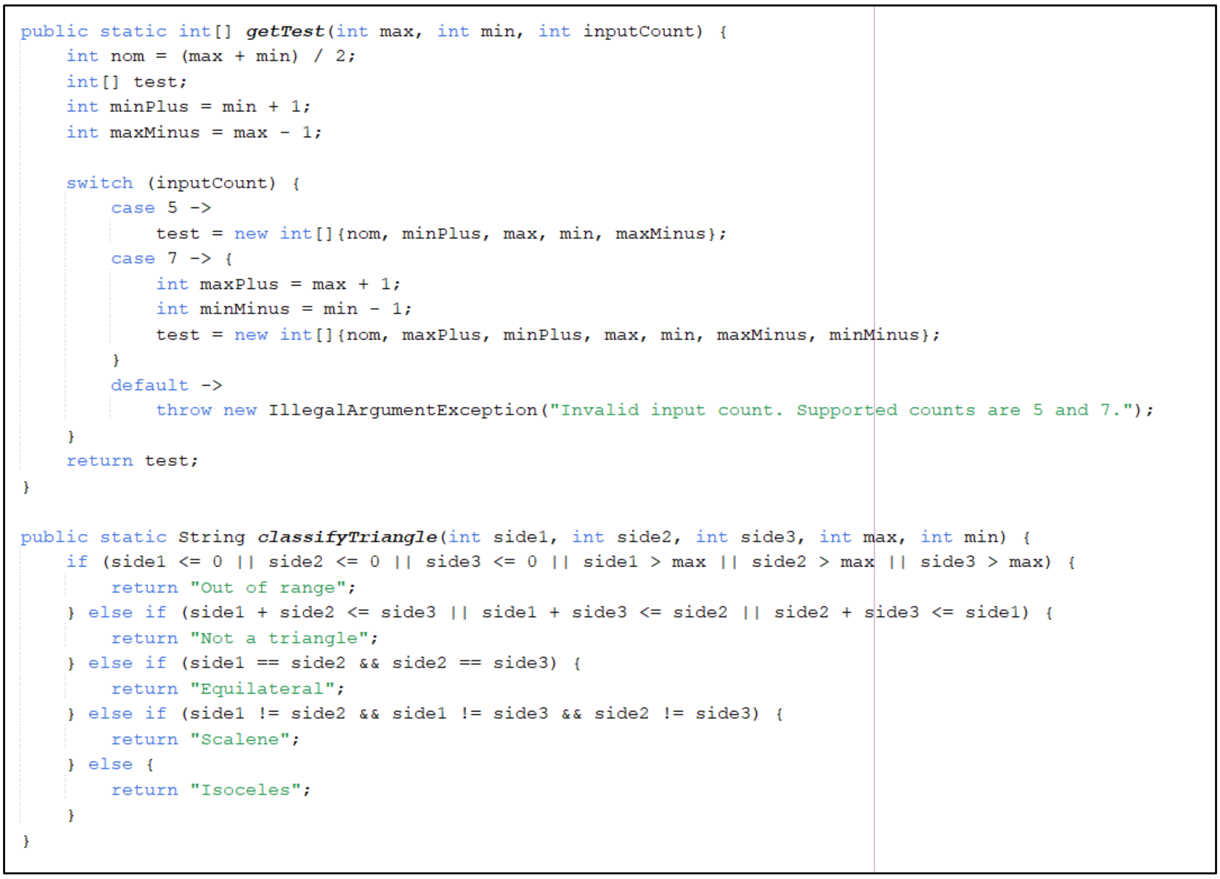
5 unit: Unique elements = 13

7 unit: Unique elements = 19

1.2 Triangle Function



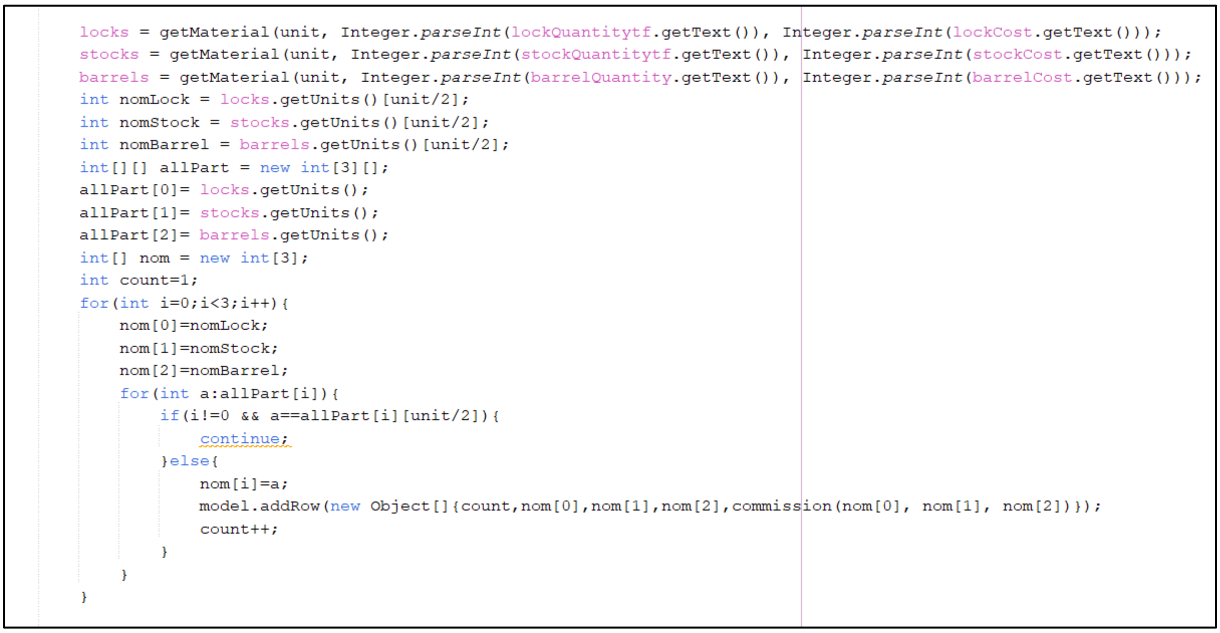
```java
// Generate test cases
for (int i = 0; i < 3; i++) {
    side.add(getTest(max, min, input));
}

// Set test cases 1
int side1 = side.get(0)[0];
int side2 = side.get(1)[0];
for (int i = 0; i < side.get(2).length; i++) {
    int side3 = side.get(2)[i];
    expected = classifyTriangle(side1, side2, side3, max, min);
    testcases.add(new TriangleTestcase(side1, side2, side.get(2)[i], expected));
}

// Set test cases 2
side1 = side.get(1)[0];
side2 = side.get(2)[0];
for (int i = 1; i < side.get(0).length; i++) {
    int side3 = side.get(0)[i];
    expected = classifyTriangle(side1, side2, side3, max, min);
    testcases.add(new TriangleTestcase(side.get(0)[i], side1, side2, expected));
}

// Set test cases 3
side1 = side.get(2)[0];
side2 = side.get(0)[0];
for (int i = 1; i < side.get(1).length; i++) {
    int side3 = side.get(1)[i];
    expected = classifyTriangle(side1, side2, side3, max, min);
    testcases.add(new TriangleTestcase(side1, side.get(1)[i], side2, expected));
}
```



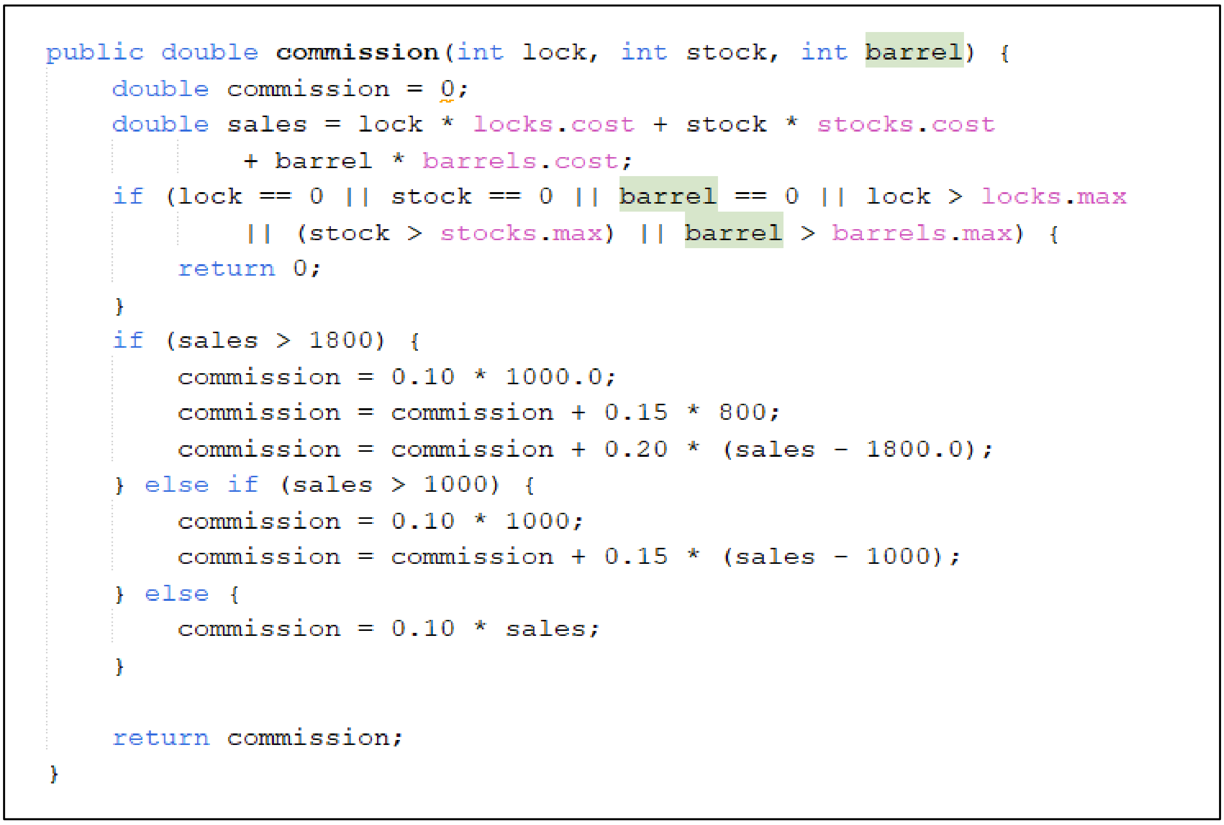
```java
public static int[] getTest(int max, int min, int inputCount) {
    int nom = (max + min) / 2;
    int[] test;
    int minPlus = min + 1;
    int maxMinus = max - 1;

    switch (inputCount) {
        case 5 ->
            test = new int[]{nom, minPlus, max, min, maxMinus};
        case 7 -> {
            int maxPlus = max + 1;
            int minMinus = min - 1;
            test = new int[]{nom, maxPlus, minPlus, max, min, maxMinus, minMinus};
        }
        default ->
            throw new IllegalArgumentException("Invalid input count. Supported counts are 5 and 7.");
    }
    return test;
}

public static String classifyTriangle(int side1, int side2, int side3, int max, int min) {
    if (side1 <= 0 || side2 <= 0 || side3 <= 0 || side1 > max || side2 > max || side3 > max) {
        return "Out of range";
    } else if (side1 + side2 <= side3 || side1 + side3 <= side2 || side2 + side3 <= side1) {
        return "Not a triangle";
    } else if (side1 == side2 && side2 == side3) {
        return "Equilateral";
    } else if (side1 != side2 && side1 != side3 && side2 != side3) {
        return "Scalene";
    } else {
        return "Isoceles";
    }
}
```

1.3 Commission Function

Generate Testcase Method:



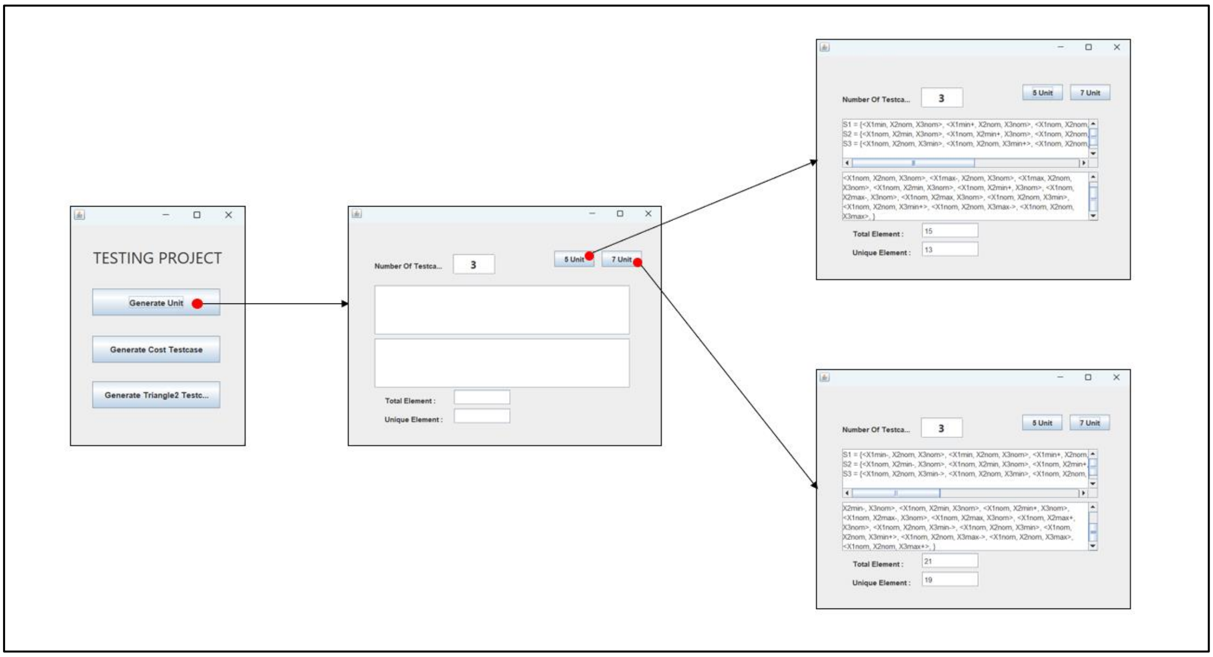
```
locks = getMaterial(unit, Integer.parseInt(lockQuantitytf.getText()), Integer.parseInt(lockCost.getText()));
stocks = getMaterial(unit, Integer.parseInt(stockQuantitytf.getText()), Integer.parseInt(stockCost.getText()));
barrels = getMaterial(unit, Integer.parseInt(barrelQuantity.getText()), Integer.parseInt(barrelCost.getText()));
int nomLock = locks.getUnits()[unit/2];
int nomStock = stocks.getUnits()[unit/2];
int nomBarrel = barrels.getUnits()[unit/2];
int[][] allPart = new int[3][];
allPart[0]= locks.getUnits();
allPart[1]= stocks.getUnits();
allPart[2]= barrels.getUnits();
int[] nom = new int[3];
int count=1;
for(int i=0;i<3;i++){
    nom[0]=nomLock;
    nom[1]=nomStock;
    nom[2]=nomBarrel;
    for(int a:allPart[i]){
        if(i!=0 && a==allPart[i][unit/2]){
            continue;
        }else{
            nom[i]=a;
            model.addRow(new Object[]{count,nom[0],nom[1],nom[2],commission(nom[0], nom[1], nom[2])});
            count++;
        }
    }
}
```



```
public double commission(int lock, int stock, int barrel) {
    double commission = 0;
    double sales = lock * locks.cost + stock * stocks.cost
            + barrel * barrels.cost;
    if (lock == 0 || stock == 0 || barrel == 0 || lock > locks.max
            || (stock > stocks.max) || barrel > barrels.max) {
        return 0;
    }
    if (sales > 1800) {
        commission = 0.10 * 1000.0;
        commission = commission + 0.15 * 800;
        commission = commission + 0.20 * (sales - 1800.0);
    } else if (sales > 1000) {
        commission = 0.10 * 1000;
        commission = commission + 0.15 * (sales - 1000);
    } else {
        commission = 0.10 * sales;
    }

    return commission;
}
```

Chapter 2: Demonstration

2.1 Problem 1 Implementation



Number Of Testca... 3

Total Element :

Unique Element :

Number Of Testca... 3

S1 = {<X1min, X2nom, X3nom>, <X1min+, X2nom, X3nom>, <X1nom, X2nom,
S2 = {<X1nom, X2min, X3nom>, <X1nom, X2min+, X3nom>, <X1nom, X2nom,
S3 = {<X1nom, X2nom, X3min>, <X1nom, X2nom, X3min+>, <X1nom, X2nom,

<X1nom, X2nom, X3nom>, <X1max-, X2nom, X3nom>, <X1max, X2nom, X3nom>, <X1nom, X2min, X3nom>, <X1nom, X2min+, X3nom>, <X1nom, X2max-, X3nom>, <X1nom, X2max, X3nom>, <X1nom, X2nom, X3min>, <X1nom, X2nom, X3min+>, <X1nom, X2nom, X3max->, <X1nom, X2nom, X3max>, }

Total Element : 15

Unique Element : 13

Number Of Testca... 3

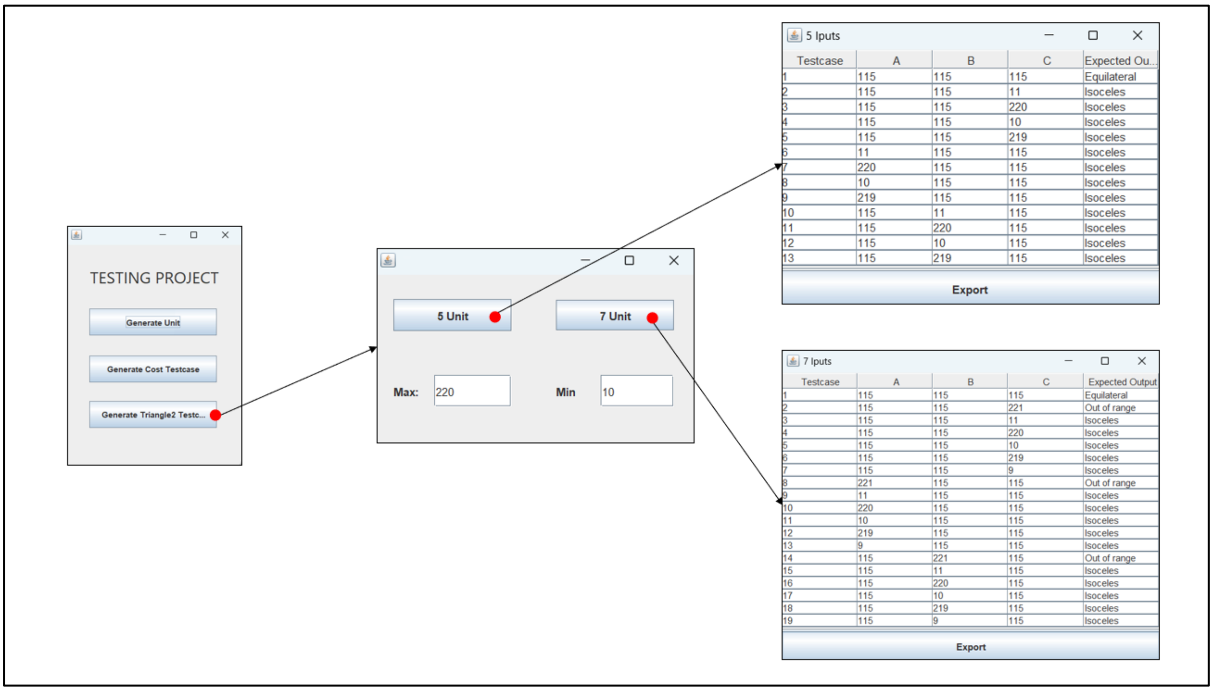
S1 = {<X1min-, X2nom, X3nom>, <X1min, X2nom, X3nom>, <X1min+, X2nom,
S2 = {<X1nom, X2min-, X3nom>, <X1nom, X2min, X3nom>, <X1nom, X2min+,
S3 = {<X1nom, X2nom, X3min->, <X1nom, X2nom, X3min>, <X1nom, X2nom,

X2min-, X3nom>, <X1nom, X2min, X3nom>, <X1nom, X2min+, X3nom>, <X1nom, X2max-, X3nom>, <X1nom, X2max, X3nom>, <X1nom, X2max+, X3nom>, <X1nom, X2nom, X3min->, <X1nom, X2nom, X3min>, <X1nom, X2nom, X3min+>, <X1nom, X2nom, X3max->, <X1nom, X2nom, X3max>, <X1nom, X2nom, X3max+>, }

Total Element : 21

Unique Element : 19

2.2 Triangle Function



Max: 220

Min 10

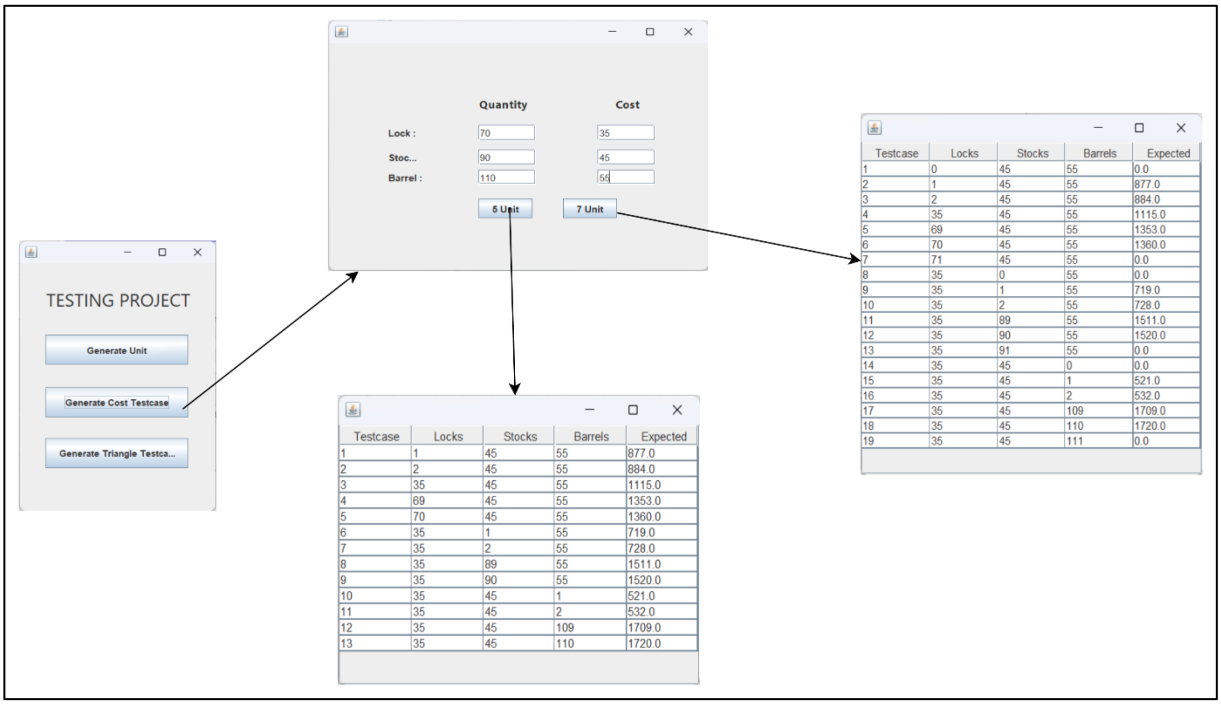
5 Iputs

Testcase	A	B	C	Expected Ou...
1	115	115	115	Equilateral
2	115	115	11	Isoceles
3	115	115	220	Isoceles
4	115	115	10	Isoceles
5	115	115	219	Isoceles
6	11	115	115	Isoceles
7	220	115	115	Isoceles
8	10	115	115	Isoceles
9	219	115	115	Isoceles
10	115	11	115	Isoceles
11	115	220	115	Isoceles
12	115	10	115	Isoceles
13	115	219	115	Isoceles

7 Iputs

Testcase	A	B	C	Expected Output
1	115	115	115	Equilateral
2	115	115	221	Out of range
3	115	115	11	Isoceles
4	115	115	220	Isoceles
5	115	115	10	Isoceles
6	115	115	219	Isoceles
7	115	115	9	Isoceles
8	221	115	115	Out of range
9	11	115	115	Isoceles
10	220	115	115	Isoceles
11	10	115	115	Isoceles
12	219	115	115	Isoceles
13	9	115	115	Isoceles
14	115	221	115	Out of range
15	115	11	115	Isoceles
16	115	220	115	Isoceles
17	115	10	115	Isoceles
18	115	219	115	Isoceles
19	115	9	115	Isoceles

2.3 Commission Function



	Quantity	Cost
Lock :	70	35
Stoc...	90	45
Barrel :	110	55

Testcase	Locks	Stocks	Barrels	Expected
1	0	45	55	0.0
2	1	45	55	877.0
3	2	45	55	884.0
4	35	45	55	1115.0
5	69	45	55	1353.0
6	70	45	55	1360.0
7	71	45	55	0.0
8	35	0	55	0.0
9	35	1	55	719.0
10	35	2	55	728.0
11	35	89	55	1511.0
12	35	90	55	1520.0
13	35	91	55	0.0
14	35	45	0	0.0
15	35	45	1	521.0
16	35	45	2	532.0
17	35	45	109	1709.0
18	35	45	110	1720.0
19	35	45	111	0.0

Testcase	Locks	Stocks	Barrels	Expected
1	1	45	55	877.0
2	2	45	55	884.0
3	35	45	55	1115.0
4	69	45	55	1353.0
5	70	45	55	1360.0
6	35	1	55	719.0
7	35	2	55	728.0
8	35	89	55	1511.0
9	35	90	55	1520.0
10	35	45	1	521.0
11	35	45	2	532.0
12	35	45	109	1709.0
13	35	45	110	1720.0

Preferences

[1]. Hambling, B. Software testing: an istqb-bcs certified tester foundation guide. BCS, The Chartered Institute for IT.

[2]. Paul C. Jorgensen, Software Testing: A Craftsman's Approach. CRC Press (Taylor and Francis Group), Boca Raton, New York.