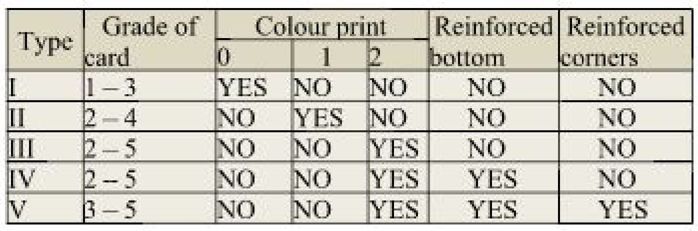
ADPROG Assessment

Description

We have designed an ordering system for FlexBox. The users can input their box details to see the price of the box. As they add more boxes, a receipt is printed on the screen which includes the overall total. The users enter in the size (width X length X height), the grade (between 1 to 5), and the quantity(1 to 100). Besides these essentials, they can also choose to have extra features such as colour (0 to 2), reinforced bottom, reinforced corner or sealable top. These extras increase the price of the box depending on base price. All boxes can have sealable top but the other extras are box specific. This table demonstrates what requirements the boxes have to fulfil for the company to make them:



Type	Grade of card	Colour print			Reinforced bottom	Reinforced corners
		0	1	2		
I	1 – 3	YES	NO	NO	NO	NO
II	2 – 4	NO	YES	NO	NO	NO
III	2 – 5	NO	NO	YES	NO	NO
IV	2 – 5	NO	NO	YES	YES	NO
V	3 – 5	NO	NO	YES	YES	YES

Assumptions

1. We take width/length/height inputs in metres as details about this weren't specified. We chose metres instead of centimetres because the brief gave us the price of the box with metres squared. Whichever measurement we took, the price was the same regardless.

2. We weren't given the minimum and maximum width/length/height of the box that FlexBox makes so we had to make an assumption. We decided to go with 0.5m to 10m. We used validation on the inputs to make sure the size didn't go over this or under this limit.

3. We set the maximum quantity per box order as a 100. The customers can order multiple times so they can get over this limit manually but they can only order 100 of a box at once. This is validated in the input form as this is a way to avoid accidental purchases (e.g. users entering 500 instead of 50).

4. We were instructed to take in the options for 0, 1 or 2 colour(s). We never actually take any input to choose what colour(s) they want. We assume that part is done separately and is not important to the program as the price will be the same regardless.

5. We assume that the FlexBox was their own support system implemented in their company. This program does not have an in built "help" menu.

6. The program doesn't lead onto another billing system and this interface does not have to do any form of transaction between the user and FlexBoxes bank accounts. This program is solely for checking and quoting the price to the customer.

Limitations

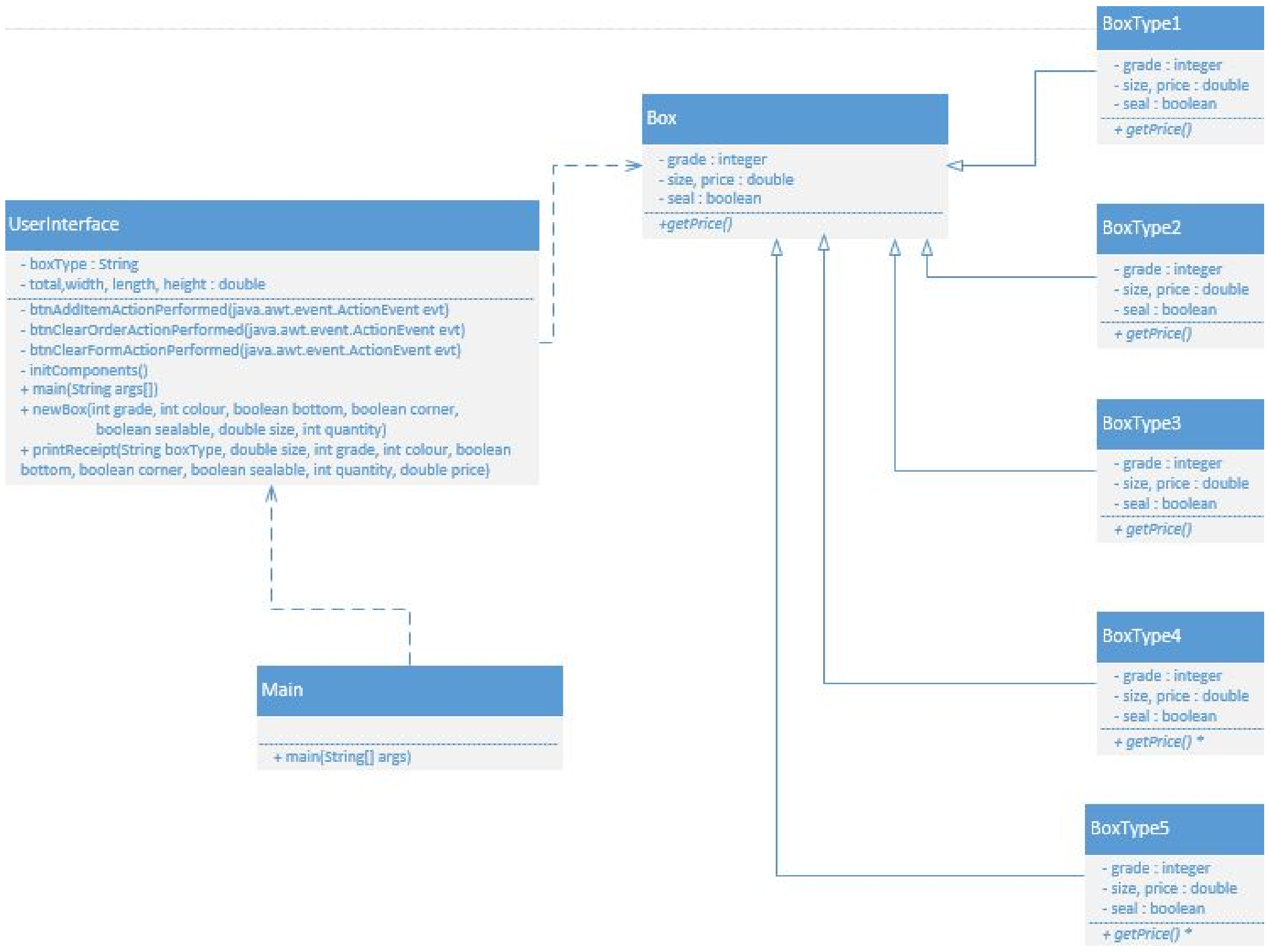
1. We are relatively new to developing UI in Java so we relied on SWING and NetBeans' drag and drop features as well as various online resources to learn this quickly.

2. We did not make use of a version control system, such as github, so we were limited to one person programming at one time. However we did use google drive to store the program as it allows us to restore to a previous version.

3. We had some technical issues downloading work from Google Drive and some group members did not have the appropriate software for developing the design diagrams (Microsoft Visio).

Diagrams

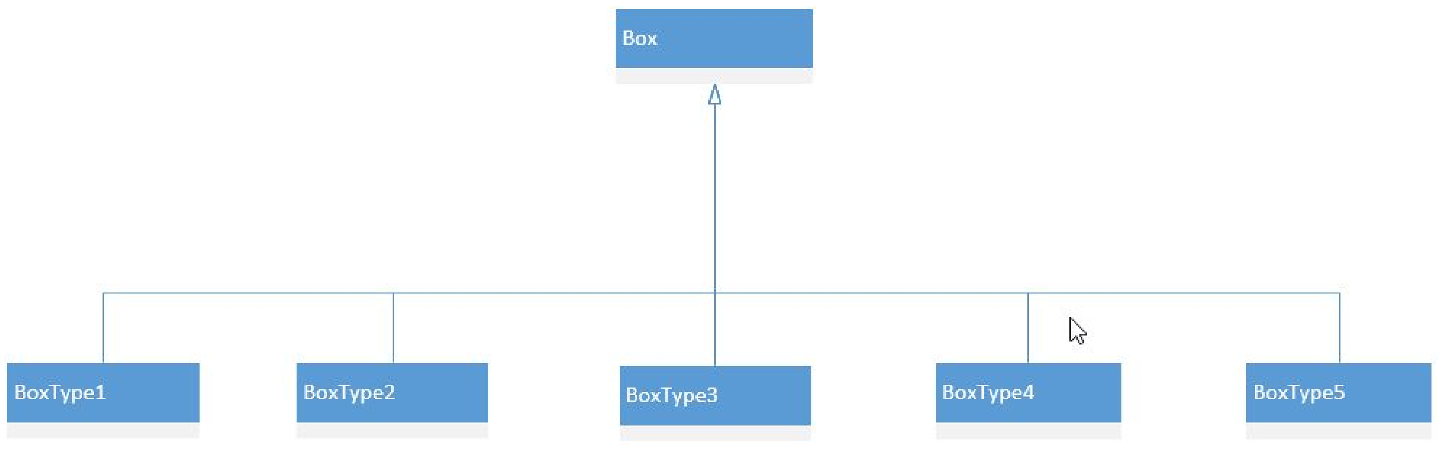
Class



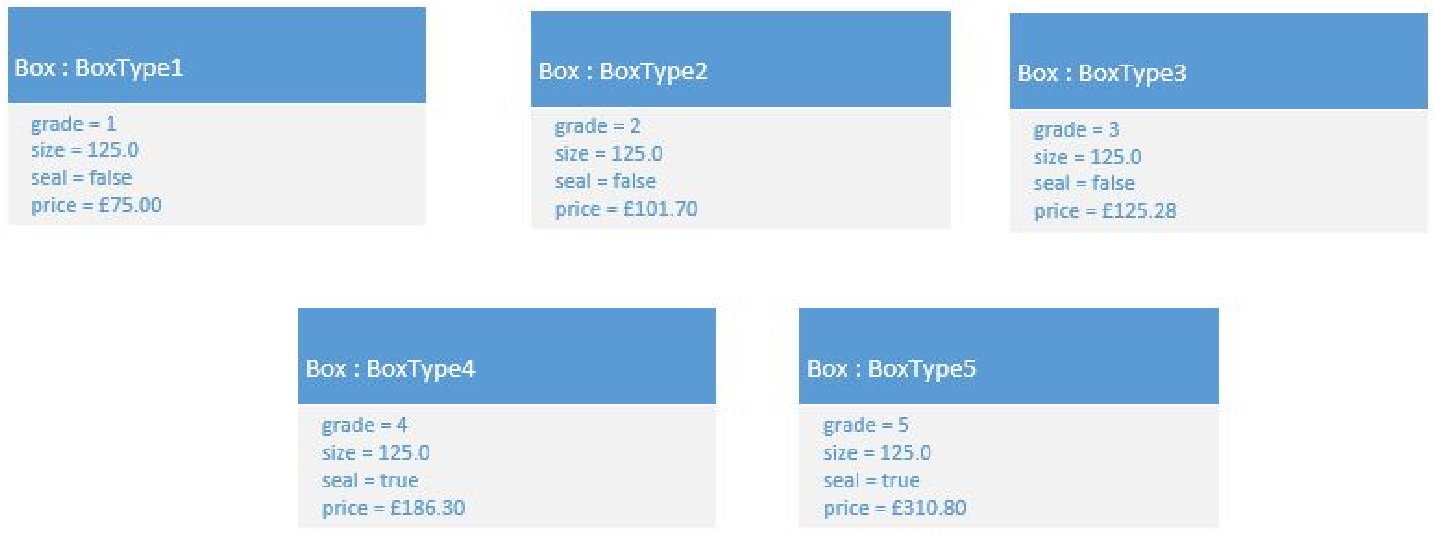
getPrice() is an abstract method

Box class is an abstract class which contains getPrice(). This abstract method gets the price of the specific type box depending on what extras it has.

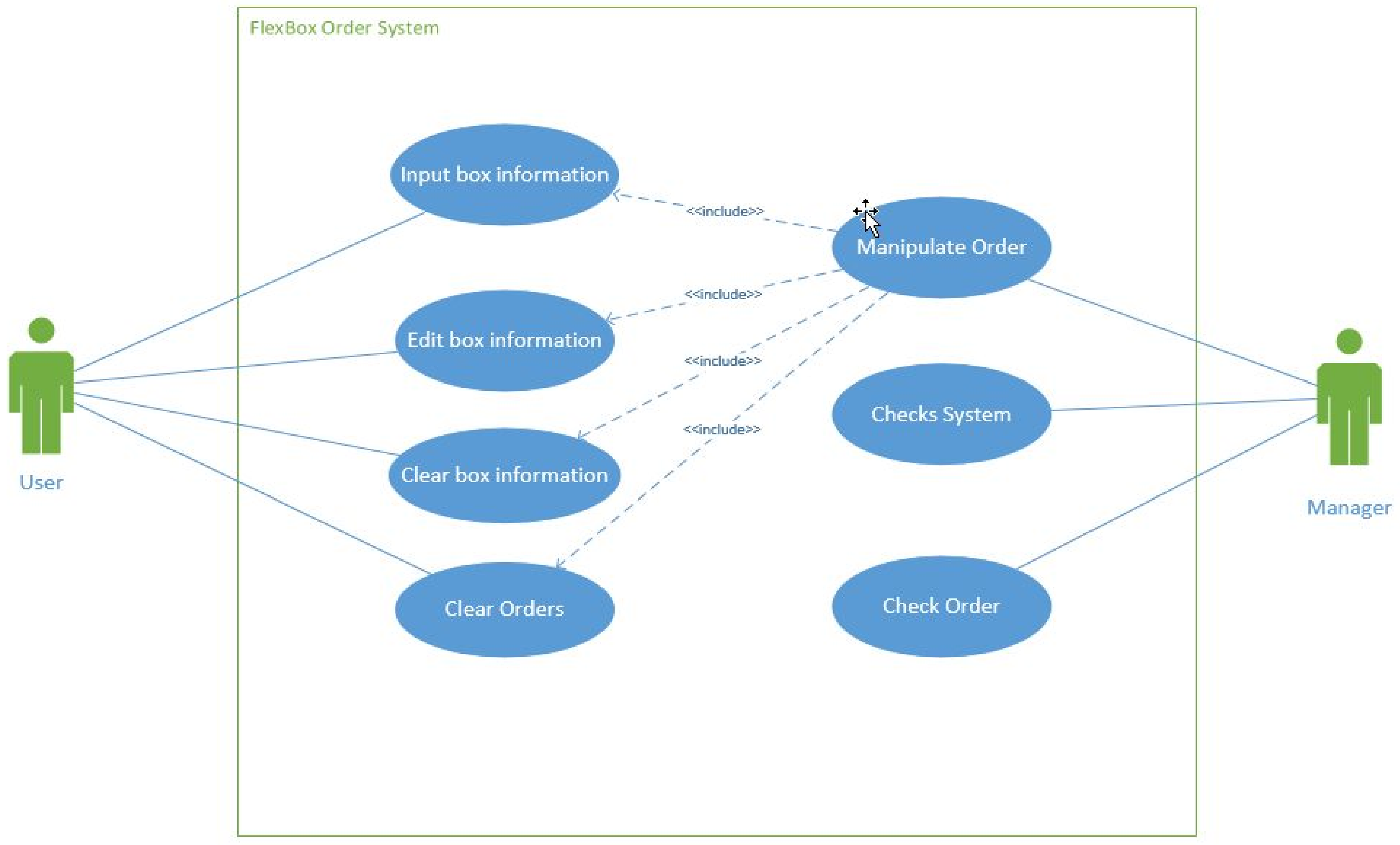
Hierarchy



Instance



Use Case



The user can input the details of the box choosing if he wants to add or clear the order. The user can't manipulate the order the same way as the manager. The manager checks the system for any order that has been sent and can manipulate the order if it is needed for a customer.

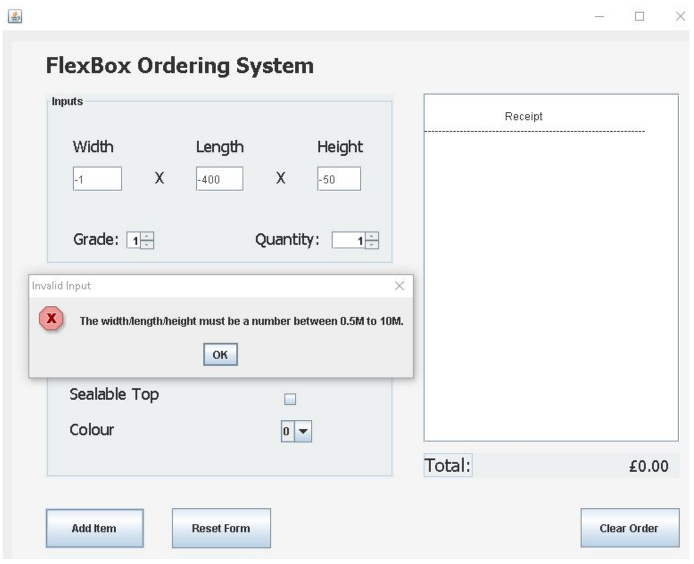
Testing

No.	Purpose	Test Value	Expected Result	Actual Result
1	Width/Length/Height shouldn't accept negative values	-1 -400 -50	A warning message should appear. "The width/length/height must be a number between 0.5M to 10M."	As expected. Evidence Screenshot below.
2	Width/Length/Height shouldn't accept values below 0.5m	0.3 0.2 0.1	A warning message should appear. "The width/length/height must be a number between 0.5M to 10M."	As expected. Evidence Screenshot below.
3	Width/Length/Height shouldn't accept values above 10m	11 12 500	A warning message should appear. ""The size must be a positive number."	As expected. Evidence Screenshot below.
4	Width/Length/Height shouldn't accept alphabets or symbols.	B K **	A warning message should appear. ""The size must be a positive number."	As expected. Evidence Screenshot below.
5	Grade - spinner shouldn't go below 1 or above 5.	<Using Spinner>	The spinner arrows shouldn't do anything when trying go below 1 or above 5.	As expected. Evidence Screenshot below.
6	Making sure total price correctly matches the total of all order prices.	Box 1 price = £0.75 Box 2 price = £1.02 Box 3 price = £1.34 Box 4 price = £1.76 Box 5 price = £2.94	£7.80	Total price = £7.81 Failed. Rounded Incorrectly.
7	The price for extra features are calculated correctly	-0.5 x 0.5 x 0,5 Grade 5 All extras	3.11	As expected. Evidence Screenshot below.
8	Quantity shouldn't go below 1 or over 100.	<Using Spinner>	The spinner arrows shouldn't do anything when trying go below 1 or above 100.	Failed. Lower limit was set to 0 instead of 1.
9	The system should only progress the order if it matches the box types made by FlexBox.	Test for BoxType 1 to 5	A warning message should appear. "Flexbox doesn't produce any boxes that match these requirements. Tip: Boxes with higher grade allow more extra features such as Colours, Reinforced Corners, & Reinforced Bottoms."	As expected. Evidence Screenshot below.
10	Total price is calculated correctly	Total price of test 6 + 7	3.86	As expected. Evidence Screenshot below.
11	Price is calculated correctly with more than 1 quantity	0.5 x 0.5 x 0,5 Grade 1	1.50	As expected. Evidence Screenshot below.

		No extras Quantity 2		

Test Screenshots

Test 1



FlexBox Ordering System

Inputs

Width: -1 X Length: -400 X Height: -50

Grade: 1 Quantity: 1

Invalid Input

The width/length/height must be a number between 0.5M to 10M.

OK

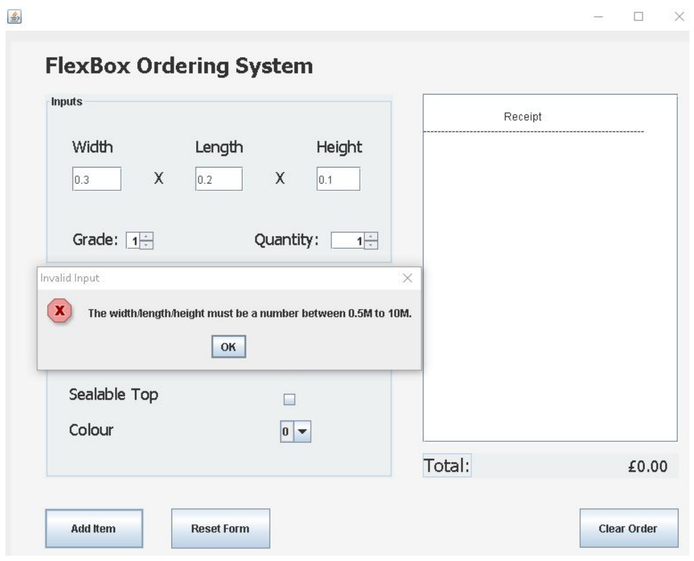
Sealable Top

Colour: 0

Receipt

Total: £0.00

Test 2



FlexBox Ordering System

Inputs

Width: 0.3 X Length: 0.2 X Height: 0.1

Grade: 1 Quantity: 1

Invalid Input

The width/length/height must be a number between 0.5M to 10M.

OK

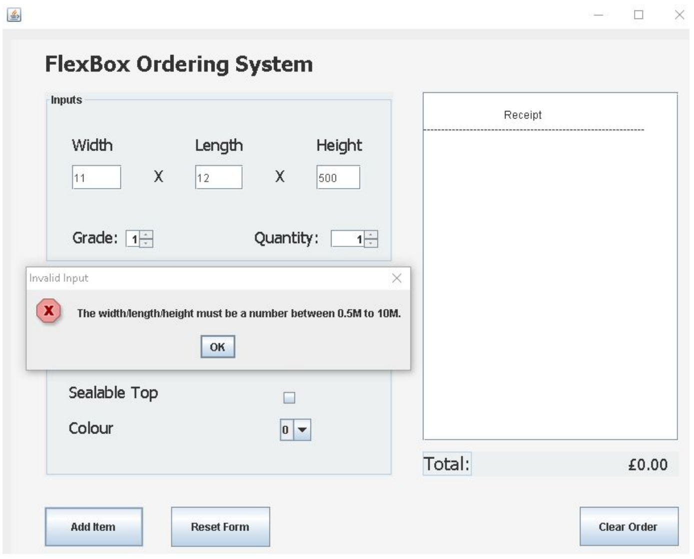
Sealable Top

Colour: 0

Receipt

Total: £0.00

Test 3



FlexBox Ordering System

Inputs

Width: 11 X Length: 12 X Height: 500

Grade: 1 Quantity: 1

Invalid Input

The width/length/height must be a number between 0.5M to 10M.

OK

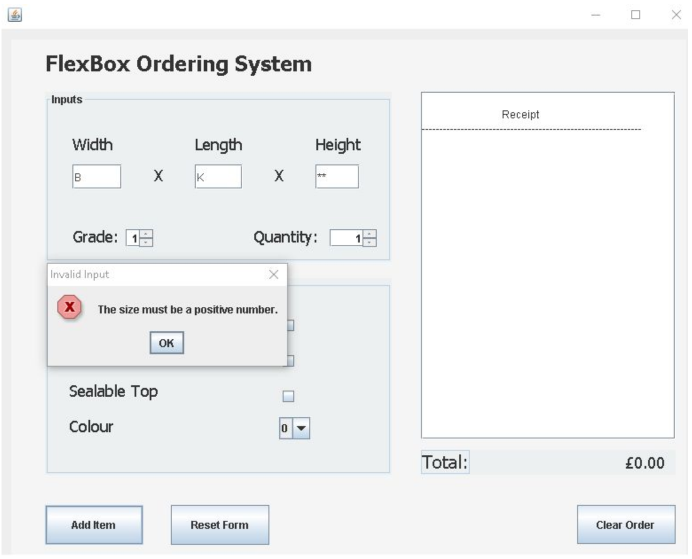
Sealable Top

Colour: 0

Receipt

Total: £0.00

Test 4



FlexBox Ordering System

Inputs

Width: B X Length: K X Height: **

Grade: 1 Quantity: 1

Invalid Input

The size must be a positive number.

OK

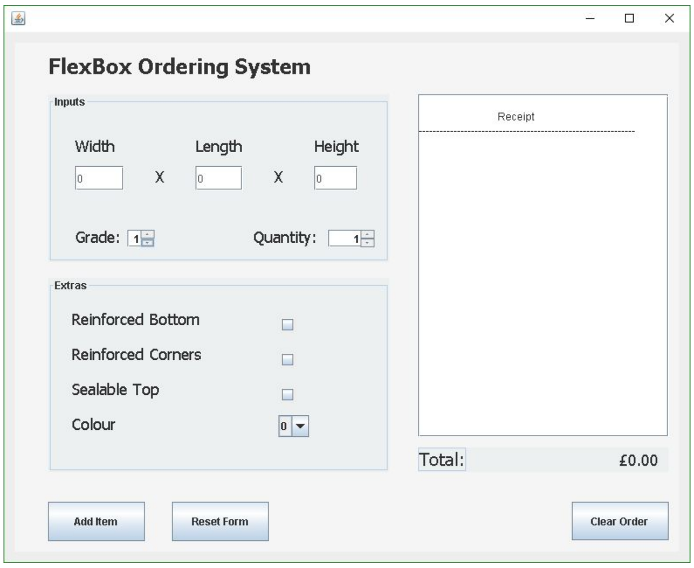
Sealable Top

Colour: 0

Receipt

Total: £0.00

Test 5



FlexBox Ordering System

Inputs

Width 0 X Length 0 X Height 0

Grade: 1 Quantity: 1

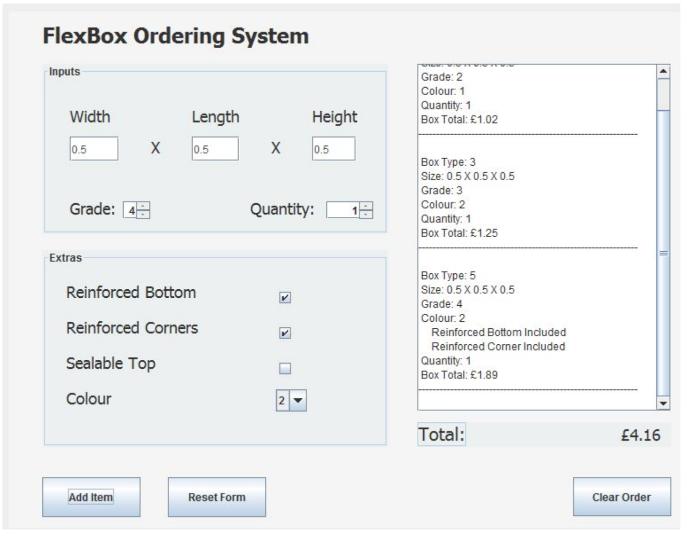
Extras

Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 0

```
Receipt
-----------------------------------------
```

Total: £0.00

Test 6 Failed (details below)



FlexBox Ordering System

Inputs

Width 0.5 X Length 0.5 X Height 0.5

Grade: 4 Quantity: 1

Extras

Reinforced Bottom ✓
Reinforced Corners ✓
Sealable Top ☐
Colour 2

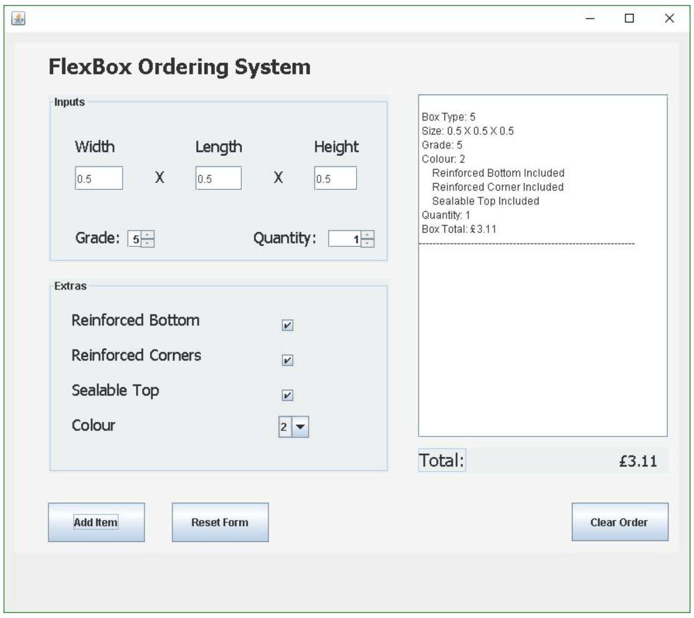
```
Grade: 2
Colour: 1
Quantity: 1
Box Total: £1.02
-----------------------------------------

Box Type: 3
Size: 0.5 X 0.5 X 0.5
Grade: 3
Colour: 2
Quantity: 1
Box Total: £1.25
-----------------------------------------

Box Type: 5
Size: 0.5 X 0.5 X 0.5
Grade: 4
Colour: 2
   Reinforced Bottom Included
   Reinforced Corner Included
Quantity: 1
Box Total: £1.89
-----------------------------------------
```

Total: £4.16

Test 7



FlexBox Ordering System

Inputs

Width 0.5 X Length 0.5 X Height 0.5

Grade: 5 Quantity: 1

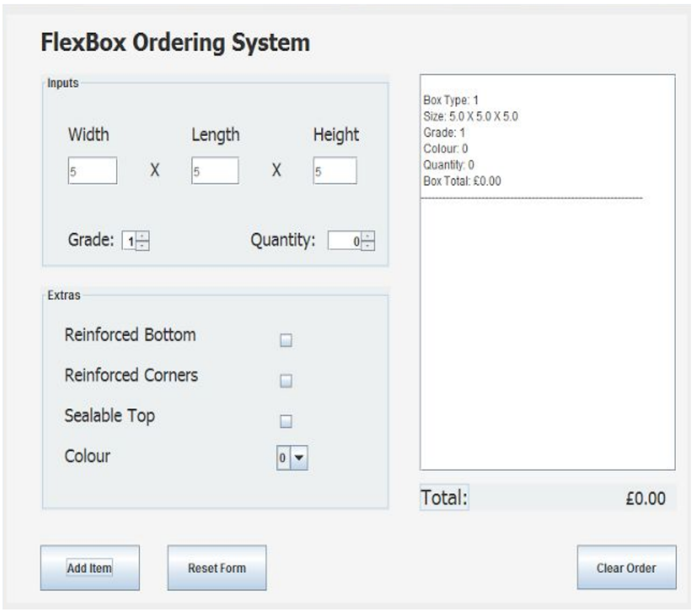
Extras

Reinforced Bottom ✓
Reinforced Corners ✓
Sealable Top ✓
Colour 2

```
Box Type: 5
Size: 0.5 X 0.5 X 0.5
Grade: 5
Colour: 2
   Reinforced Bottom Included
   Reinforced Corner Included
   Sealable Top Included
Quantity: 1
Box Total: £3.11
-----------------------------------------
```

Total: £3.11

Test 8 Failed (details below)



FlexBox Ordering System

Inputs

Width 5 X Length 5 X Height 5

Grade: 1 Quantity: 0

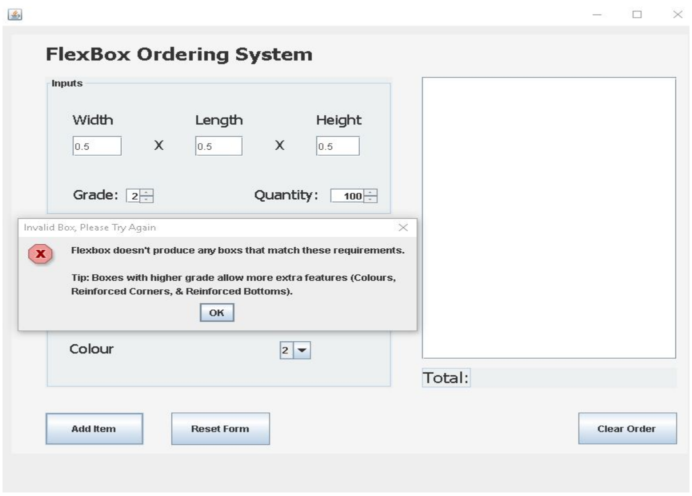
Extras

Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 0

```
Box Type: 1
Size: 5.0 X 5.0 X 5.0
Grade: 1
Colour: 0
Quantity: 0
Box Total: £0.00
-----------------------------------------
```

Total: £0.00

Test 9



FlexBox Ordering System

Inputs

Width 0.5 X Length 0.5 X Height 0.5

Grade: 2 Quantity: 100

Invalid Box, Please Try Again

Flexbox doesn't produce any boxs that match these requirements.

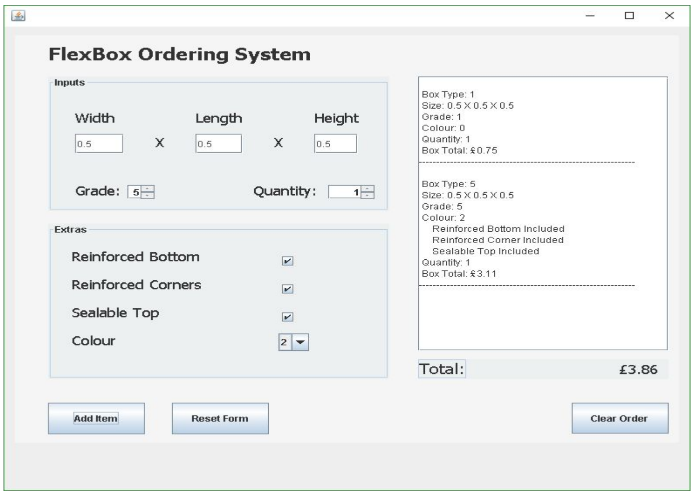
Tip: Boxes with higher grade allow more extra features (Colours, Reinforced Corners, & Reinforced Bottoms).

OK

Colour 2

Total:

Test 10



FlexBox Ordering System

Inputs

Width 0.5 X Length 0.5 X Height 0.5

Grade: 5 Quantity: 1

Extras

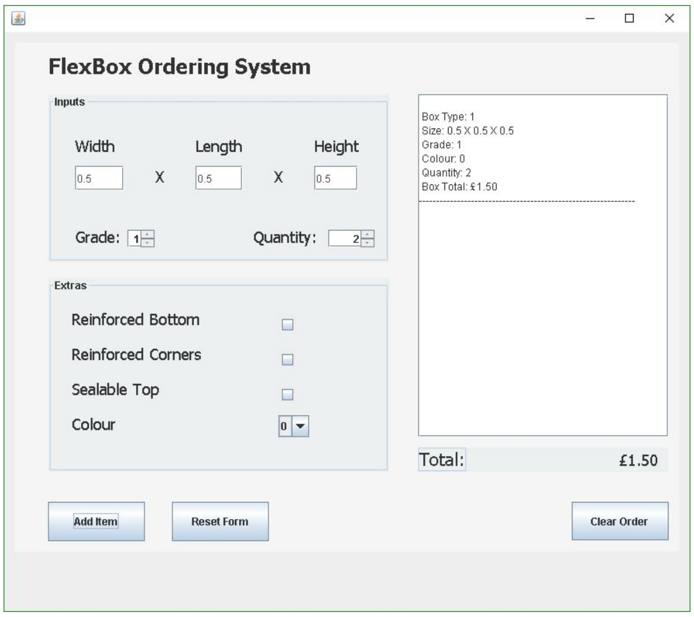
Reinforced Bottom ✓
Reinforced Corners ✓
Sealable Top ✓
Colour 2

```
Box Type: 1
Size: 0.5 X 0.5 X 0.5
Grade: 1
Colour: 0
Quantity: 1
Box Total: £0.75
-----------------------------------------

Box Type: 5
Size: 0.5 X 0.5 X 0.5
Grade: 5
Colour: 2
   Reinforced Bottom Included
   Reinforced Corner Included
   Sealable Top Included
Quantity: 1
Box Total: £3.11
-----------------------------------------
```

Total: £3.86

Test 11



```
Box Type: 1
Size: 0.5 X 0.5 X 0.5
Grade: 1
Colour: 0
Quantity: 2
Box Total: £1.50
------------------------------------------------
```

Total: £1.50

Test Improvements

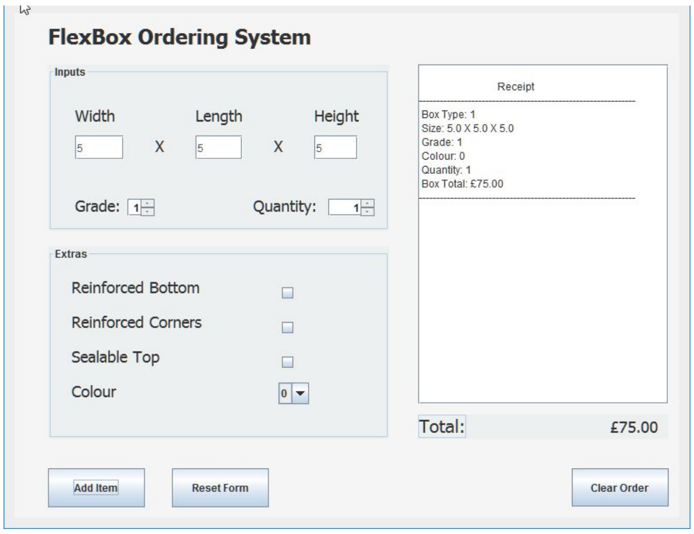
Test 6
We used java.text.DecimalFormat to round the number to 2 decimal points. This gave incorrect values for the total in some cases as it chopped off the numbers after the second digits. To fix this, we used “(Math.round(total*100))/100;" to first round correctly and format it to 2 digits. After the rounding, the test was successful and we got the correct number (£7.81) instead of (£7.80).

Test 8
We had accidently set the quantity limit to anywhere between 0-100. 0 is too low so and would output a useless order that would not add anything for the user. We have since changed the quantity to a limit between 1-100. After making the change, the test was successful as the user couldn’t go below 1 quantity.

Sample Input / Output

BoxType1 input and output:

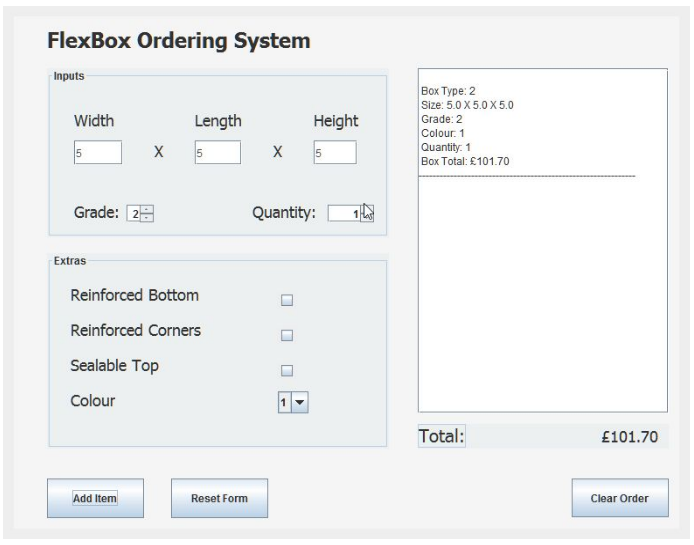
BoxType2 input and output:



Width 5 X Length 5 X Height 5
Grade: 1 Quantity: 1
Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 0

```
Receipt
----------------------------
Box Type: 1
Size: 5.0 X 5.0 X 5.0
Grade: 1
Colour: 0
Quantity: 1
Box Total: £75.00
----------------------------
```

Total: £75.00

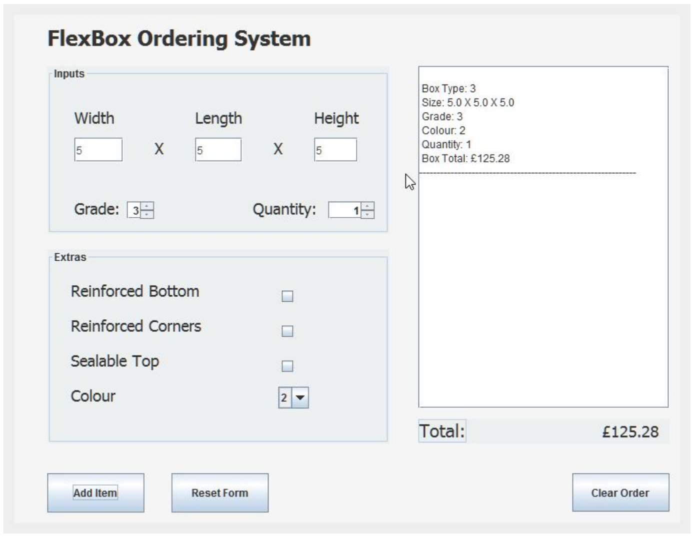


Width 5 X Length 5 X Height 5
Grade: 2 Quantity: 1
Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 1

```
Box Type: 2
Size: 5.0 X 5.0 X 5.0
Grade: 2
Colour: 1
Quantity: 1
Box Total: £101.70
----------------------------
```

Total: £101.70

BoxType3 input and output:

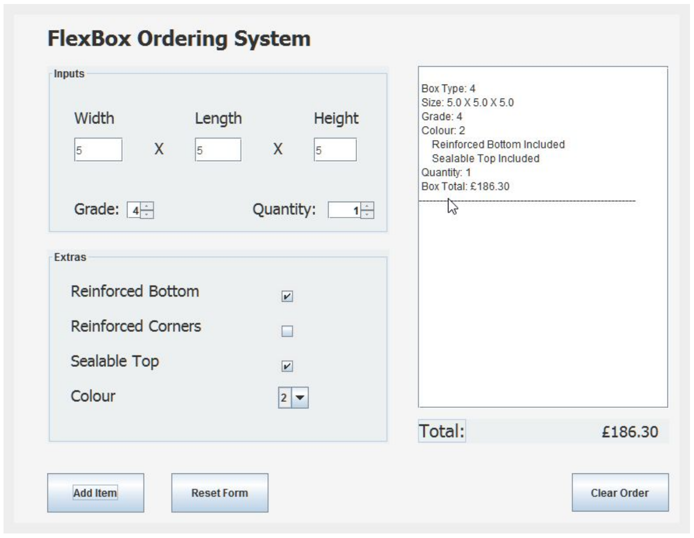


Width 5 X Length 5 X Height 5
Grade: 3 Quantity: 1
Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 2

```
Box Type: 3
Size: 5.0 X 5.0 X 5.0
Grade: 3
Colour: 2
Quantity: 1
Box Total: £125.28
----------------------------
```

Total: £125.28

BoxType4 input and output:

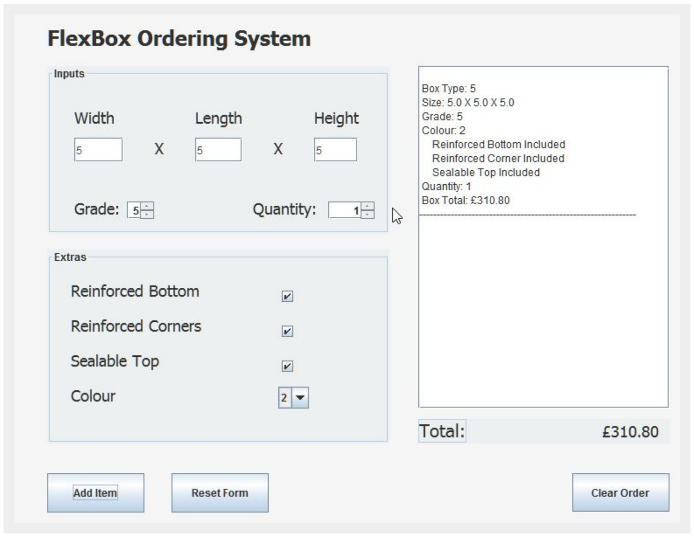


Width 5 X Length 5 X Height 5
Grade: 4 Quantity: 1
Reinforced Bottom ✓
Reinforced Corners ☐
Sealable Top ✓
Colour 2

```
Box Type: 4
Size: 5.0 X 5.0 X 5.0
Grade: 4
Colour: 2
  Reinforced Bottom Included
  Sealable Top Included
Quantity: 1
Box Total: £186.30
----------------------------
```

Total: £186.30

BoxType5 input and output:

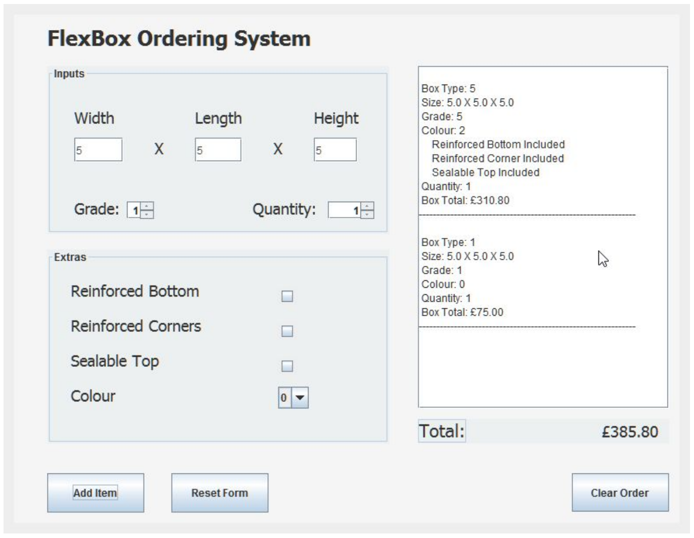


Width 5 X Length 5 X Height 5
Grade: 5 Quantity: 1
Reinforced Bottom ✓
Reinforced Corners ✓
Sealable Top ✓
Colour 2

```
Box Type: 5
Size: 5.0 X 5.0 X 5.0
Grade: 5
Colour: 2
  Reinforced Bottom Included
  Reinforced Corner Included
  Sealable Top Included
Quantity: 1
Box Total: £310.80
----------------------------
```

Total: £310.80

Multiple orders:



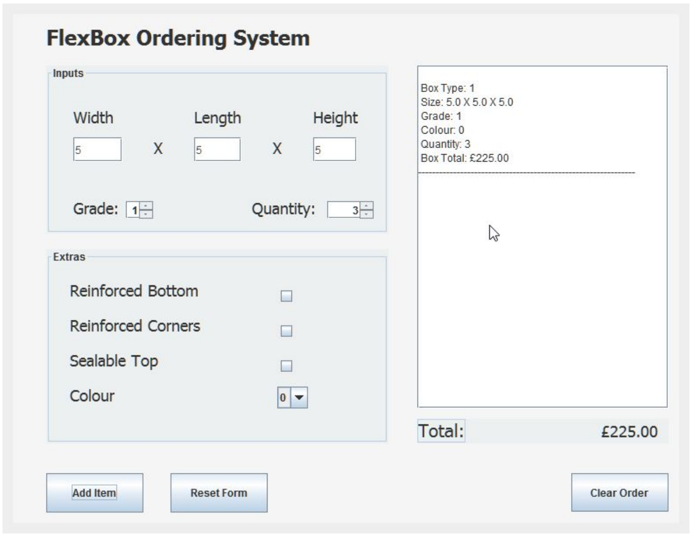
Width 5 X Length 5 X Height 5
Grade: 1 Quantity: 1
Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 0

```
Box Type: 5
Size: 5.0 X 5.0 X 5.0
Grade: 5
Colour: 2
  Reinforced Bottom Included
  Reinforced Corner Included
  Sealable Top Included
Quantity: 1
Box Total: £310.80
----------------------------

Box Type: 1
Size: 5.0 X 5.0 X 5.0
Grade: 1
Colour: 0
Quantity: 1
Box Total: £75.00
----------------------------
```

Total: £385.80

Multiple quantities:



Width 5 X Length 5 X Height 5

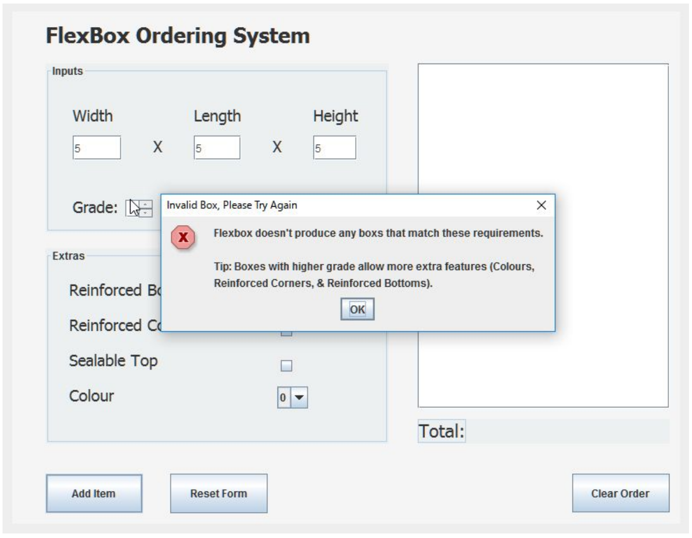
Grade: 1 Quantity: 3

Reinforced Bottom ☐
Reinforced Corners ☐
Sealable Top ☐
Colour 0

```
Box Type: 1
Size: 5.0 X 5.0 X 5.0
Grade: 1
Colour: 0
Quantity: 3
Box Total: £225.00
```

Total: £225.00

Incorrect box inputted:



Width 5 X Length 5 X Height 5

Grade:

Reinforced B
Reinforced C
Sealable Top ☐
Colour 0

Total:

Invalid Box, Please Try Again

Flexbox doesn't produce any boxs that match these requirements.

Tip: Boxes with higher grade allow more extra features (Colours, Reinforced Corners, & Reinforced Bottoms).

ADPROC Cwk - Group Contribution

Complete the Group Members' Contribution to Coursework **below.** This should cover the overall contribution to the coursework of each group member (remember to include your own contribution).

Group Members' Contribution to Coursework

Distribute 100% among all the members of your group (including yourself) to indicate each person's relative contribution.

For example, in a group of three students Alpha, Beta, and Gamma, where all have contributed evenly you would give 33.3% each.

However, if the contributions were significantly uneven, you might mark them as follows – Alpha has done most of the work, so give her/him 50%, Beta and Gamma have completed the rest of the work and between them Beta did 20% and Gamma did 30%.

List your group members **by student number** and their scores below:

1. ______________________________ /100

2. ______________________________ /100

3. ______________________________ /100

TOTAL 100/100

Source code

Main.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
public class Main {
    /**
    Starts program and sets GUI to visible
    */
    public static void main (String[] args) {
        //Creating a GUI window and setting it visible
        UserInterface screen = new UserInterface();
        screen.setVisible(true);
    }

}
```

Box.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
abstract class Box {
    private final int grade; //grade of the box
    private final double size; //size of the box (width * length * height)
    private final boolean seal; // if sealable top

/**
 *
 * @param grade grade of the box
 * @param size size of the box (w*l*h)
 * @param seal sealable top
 */
    public Box(int grade, double size, boolean seal) {
        this.grade = grade;
        this.size = size;
        this.seal = seal;
    }

 /**
  * Abstract method that returns the price of the box
  * @return
  */
    abstract double getPrice();
}
```

BoxType1.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
public class BoxType1 extends Box {

    private final int grade; //grade of the box
    private final double size; //size of the box (width * length * height)
    private double price; // price of the box
    private final boolean seal; // if sealable top

    /**Sub class constructor
     * @param g grade of the box
     * @param s size of the box (w*l*h)
     * @param sl sealable top
     */
    public BoxType1 (int g, double s, boolean sl) {
        super(g, s, sl);
        grade = g;
        size = s;
        seal = sl;
    }

    /**
     * Gets price of the box type1 including specific extras
     * @return
     */
    @Override
    double getPrice(){
      // Price depending on the grade
      switch(grade) {
          case 1:
              price = size * 0.5;
              break;
          case 2:
              price = size * 0.6;
              break;
          case 3:
               price = size * 0.72;
               break;
          }

         //price increase if sealable top
         if (seal) {
             price = price * 1.08;
         }
         return price;
     }

}
```

BoxType2.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
public class BoxType2 extends Box {

    private final int grade; //grade of the box
    private final double size; //size of the box (width * length * height)
    private double price; // price of the box
    private final boolean seal; // if sealable top

    /**Sub class constructor
     * @param g grade of the box
     * @param s size of the box (w*l*h)
     * @param sl sealable top
     */    public BoxType2 (int g, double s, boolean sl) {
        super(g, s, sl);
        grade = g;
        size = s;
        seal = sl;
    }

    /**
     * Gets price of the box type1 including specific extras
     * @return
     */
    @Override
    double getPrice(){
       double tPrice;
       //gets price depending on grade
```

```
        switch(grade) {
            case 2:
                price = size * 0.6;
                break;
            case 3:
                price = size * 0.72;
                break;
            case 4:
                price = size * 0.9;
                break;
         }
        tPrice = price * 1.13; //price increase for 1 colour

        //price increace if sealable top
        if (seal) {
            tPrice = tPrice + price * 0.08;
        }
        return tPrice;
    }

}
```

BoxType3.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
public class BoxType3 extends Box {

    private final int grade; //grade of the box
    private final double size; //size of the box (width * length * height)
    private double price; // price of the box
    private final boolean seal; // if sealable top

    /**Sub class constructor
     * @param g grade of the box
     * @param s size of the box (w*l*h)
     * @param sl sealable top
     */
    public BoxType3 (int g, double s, boolean sl) {
        super(g, s, sl);
        grade = g;
        size = s;
        seal = sl;
    }

    /**
     * Gets price of the box type1 including specific extras
     * @return
     */
    @Override
    double getPrice(){
        double tPrice;
        //gets price depending on grade
        switch(grade) {
            case 2:
                price = size * 0.6;
                break;

            case 3:
                price = size * 0.72;
                break;
            case 4:
                price = size * 0.9;
                break;
            case 5:
                price = size * 1.4;
         }
        tPrice = price * 1.16; //price increase for 2 colours

        //price increase for sealable top
        if (seal) {
            tPrice = tPrice + price * 0.08;
        }
        return tPrice;
    }

}
```

BoxType4.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
public class BoxType4 extends Box {

    private final int grade; //grade of the box
    private final double size; //size of the box (width * length * height)
    private double price; // price of the box
    private final boolean seal; // if sealable top

    /**Sub class constructor
     * @param g grade of the box
     * @param s size of the box (w*l*h)
     * @param sl sealable top
     */
    public BoxType4 (int g, double s, boolean sl) {
        super(g, s, sl);
        grade = g;
        size = s;
        seal = sl;
    }

    /**
     * Gets price of the box type1 including specific extras
     * @return
     */
    @Override
    double getPrice(){
       double tPrice;
       //gets price depending on grade
       switch(grade) {
          case 2:
              price = size * 0.6;
              break;

           case 3:
              price = size * 0.72;
              break;
           case 4:
              price = size * 0.9;
              break;
           case 5:
              price = size * 1.4;
        }
       //price increase for 2 colours and reinforced bottoms
       tPrice = price + (price * 0.16) + (price * 0.14);

       //price increase for sealable top
       if (seal) {
           tPrice = tPrice + price * 0.08;
       }
       return tPrice;    }
```

```
}
```

BoxType5.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
public class BoxType5 extends Box {

    private final int grade; //grade of the box
    private final double size; //size of the box (width * length * height)
    private double price; // price of the box
    private final boolean seal; // if sealable top

    /**Sub class constructor
     * @param g grade of the box
     * @param s size of the box (w*l*h)
     * @param sl sealable top
     */
    public BoxType5 (int g, double s, boolean sl) {
        super(g, s, sl);
        grade = g;
        size = s;
        seal = sl;
    }

    /**
     * Gets price of the box type1 including specific extras
     * @return
     */
    @Override
    double getPrice(){
       double tPrice;
       //gets price depending on
       switch(grade) {
           case 3:
               price = size * 0.72;
               break;
           case 4:
               price = size * 0.9;
               break;
           case 5:
               price = size * 1.4;
               break;
           }
       //price increase for 2 colours and reinforced corners + bottoms
       tPrice = price + (price * 0.16) + (price * 0.14) + (price * 0.1);

       //price increase for sealable top
       if (seal) {
           tPrice = tPrice + price * 0.08;
       }
       return tPrice;
    }

}
```

UserInterface.java

```
/**
 *
 * @author up781587, up769535, up743299
 */
import java.awt.HeadlessException;
import javax.swing.*;
import java.text.DecimalFormat;

public class UserInterface extends javax.swing.JFrame {
private String boxType;
private double total;
private double length;
private double height;
private double width;
    /**
     * Creates new form UserInterface
     */
    public UserInterface() {
        initComponents();
        spinQuantity.setValue(1);
        spinGrade.setValue(1);

    }

    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        panelBackground = new javax.swing.JPanel();
        panelDimensions = new javax.swing.JPanel();
```

```
textHeight = new javax.swing.JTextField();
jLabel2 = new javax.swing.JLabel();
textWidth = new javax.swing.JTextField();
textLength = new javax.swing.JTextField();
jLabel9 = new javax.swing.JLabel();
jLabel13 = new javax.swing.JLabel();
spinGrade = new javax.swing.JSpinner();
jLabel14 = new javax.swing.JLabel();
spinQuantity = new javax.swing.JSpinner();
jLabel1 = new javax.swing.JLabel();
jLabel4 = new javax.swing.JLabel();
jLabel6 = new javax.swing.JLabel();
panelExtras = new javax.swing.JPanel();
jLabel3 = new javax.swing.JLabel();
jLabel10 = new javax.swing.JLabel();
jLabel11 = new javax.swing.JLabel();
jLabel12 = new javax.swing.JLabel();
comboColour = new javax.swing.JComboBox<>();
checkBottom = new javax.swing.JCheckBox();
checkCorner = new javax.swing.JCheckBox();
checkTop = new javax.swing.JCheckBox();
btnAddItem = new javax.swing.JButton();
btnClearOrder = new javax.swing.JButton();
jLabel8 = new javax.swing.JLabel();
btnClearForm = new javax.swing.JButton();
jPanel5 = new javax.swing.JPanel();
labelTotal = new javax.swing.JLabel();
jLabel5 = new javax.swing.JLabel();
jScrollPane1 = new javax.swing.JScrollPane();
panelOrders = new javax.swing.JTextArea();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
setBackground(new java.awt.Color(236, 240, 241));

panelBackground.setBackground(new java.awt.Color(245, 245, 245));
panelBackground.setToolTipText("");

panelDimensions.setBackground(new java.awt.Color(236, 240, 241));
panelDimensions.setBorder(javax.swing.BorderFactory.createTitledBorder("Inputs"));
panelDimensions.setForeground(new java.awt.Color(240, 240, 240));
panelDimensions.setToolTipText("");
panelDimensions.setName("enterDimensions"); // NOI18N

textHeight.setForeground(new java.awt.Color(77, 77, 77));

jLabel2.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
jLabel2.setText("Length");

textWidth.setForeground(new java.awt.Color(77, 77, 77));

textLength.setForeground(new java.awt.Color(77, 77, 77));

jLabel9.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
jLabel9.setText("Width");

jLabel13.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
jLabel13.setText("Grade:");

spinGrade.setModel(new javax.swing.SpinnerNumberModel(1, 1, 5, 1));

jLabel14.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
jLabel14.setText("Quantity:");
```

```
        spinQuantity.setModel(new javax.swing.SpinnerNumberModel(1, 1, 100, 1));

        jLabel1.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel1.setText("Height");

        jLabel4.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel4.setText("    X");
        jLabel4.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);

        jLabel6.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel6.setText("    X");
        jLabel6.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);

        javax.swing.GroupLayout panelDimensionsLayout = new javax.swing.GroupLayout(panelDimensions);
        panelDimensions.setLayout(panelDimensionsLayout);
        panelDimensionsLayout.setHorizontalGroup(
            panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(panelDimensionsLayout.createSequentialGroup()
                .addGap(26, 26, 26)
                .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jLabel9)
                    .addGroup(panelDimensionsLayout.createSequentialGroup()
                        .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                            .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 56, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(textWidth, javax.swing.GroupLayout.PREFERRED_SIZE, 56, javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addGroup(panelDimensionsLayout.createSequentialGroup()
                                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(jLabel4, javax.swing.GroupLayout.PREFERRED_SIZE, 58, javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGroup(panelDimensionsLayout.createSequentialGroup()
                                .addGap(4, 4, 4)
                                .addComponent(spinGrade, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(panelDimensionsLayout.createSequentialGroup()
                        .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addGroup(panelDimensionsLayout.createSequentialGroup()
                                .addComponent(textLength, javax.swing.GroupLayout.PREFERRED_SIZE, 54, javax.swing.GroupLayout.PREFERRED_SIZE)
                                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE, 58, javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addComponent(jLabel2))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel1)
                            .addComponent(textHeight, javax.swing.GroupLayout.PREFERRED_SIZE, 50, javax.swing.GroupLayout.PREFERRED_SIZE)))
                    .addGroup(panelDimensionsLayout.createSequentialGroup()
                        .addGap(152, 152, 152)
                        .addComponent(spinQuantity, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(panelDimensionsLayout.createSequentialGroup()
                        .addGap(66, 66, 66)
                        .addComponent(jLabel14, javax.swing.GroupLayout.PREFERRED_SIZE, 82, javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        panelDimensionsLayout.setVerticalGroup(
            panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(panelDimensionsLayout.createSequentialGroup()
                .addGap(24, 24, 24)
                .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel2, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel9, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel1, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(textWidth, javax.swing.GroupLayout.PREFERRED_SIZE, 28, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(textLength, javax.swing.GroupLayout.PREFERRED_SIZE, 28, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(textHeight, javax.swing.GroupLayout.PREFERRED_SIZE, 28, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel4)
                    .addComponent(jLabel6))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 36, Short.MAX_VALUE)
                .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, panelDimensionsLayout.createSequentialGroup()
                        .addGroup(panelDimensionsLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                            .addComponent(jLabel13, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(spinGrade, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                            .addComponent(jLabel14, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE))
                        .addGap(7, 7, 7))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, panelDimensionsLayout.createSequentialGroup()
                        .addComponent(spinQuantity, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addContainerGap())))
        );

        panelExtras.setBackground(new java.awt.Color(236, 240, 241));
        panelExtras.setBorder(javax.swing.BorderFactory.createTitledBorder("Extras"));

        jLabel3.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel3.setText("Reinforced Bottom");

        jLabel10.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel10.setText("Colour");

        jLabel11.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel11.setText("Sealable Top");

        jLabel12.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        jLabel12.setText("Reinforced Corners");

        comboColour.setModel(new javax.swing.DefaultComboBoxModel<>(new String[] { "0", "1", "2" }));

        javax.swing.GroupLayout panelExtrasLayout = new javax.swing.GroupLayout(panelExtras);
        panelExtras.setLayout(panelExtrasLayout);
        panelExtrasLayout.setHorizontalGroup(
            panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(panelExtrasLayout.createSequentialGroup()
                .addGap(22, 22, 22)
                .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                        .addGroup(panelExtrasLayout.createSequentialGroup()
                            .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                                .addComponent(jLabel11)
                                .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED_SIZE, 131, javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addGap(106, 106, 106))
                        .addComponent(jLabel12, javax.swing.GroupLayout.Alignment.LEADING))
                    .addGroup(panelExtrasLayout.createSequentialGroup()
                        .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 176, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(61, 61, 61)))
                .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(checkBottom)
                    .addComponent(checkCorner)
                    .addComponent(checkTop)
                    .addComponent(comboColour, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, 
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        panelExtrasLayout.setVerticalGroup(
            panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(panelExtrasLayout.createSequentialGroup()
                .addContainerGap()
                .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                    .addGroup(panelExtrasLayout.createSequentialGroup()
                        .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                            .addGroup(panelExtrasLayout.createSequentialGroup()
                                .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
                                    .addComponent(checkBottom)
                                    .addComponent(jLabel3, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE))
                                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                                .addComponent(jLabel12, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE))
                            .addComponent(checkCorner))
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jLabel11, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addComponent(checkTop))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(panelExtrasLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jLabel10, javax.swing.GroupLayout.PREFERRED_SIZE, 34, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(comboColour, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, 
javax.swing.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(33, Short.MAX_VALUE))
        );

        btnAddItem.setText("Add Item");
        btnAddItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnAddItemActionPerformed(evt);
            }
        });

        btnClearOrder.setText("Clear Order");
        btnClearOrder.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnClearOrderActionPerformed(evt);
            }
        });

        jLabel8.setFont(new java.awt.Font("Tahoma", 1, 24)); // NOI18N
        jLabel8.setText("FlexBox Ordering System");

        btnClearForm.setText("Reset Form");
        btnClearForm.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                btnClearFormActionPerformed(evt);
            }
        });

        jPanel5.setBackground(new java.awt.Color(236, 240, 241));

        labelTotal.setFont(new java.awt.Font("Tahoma", 0, 18)); // NOI18N
        labelTotal.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
        labelTotal.setText("£0.00");
        labelTotal.setAlignmentX(1.0F);

        jLabel5.setFont(new java.awt.Font("Tahoma", 0, 20)); // NOI18N
        jLabel5.setText("Total:");
```

```
        jLabel5.setBorder(javax.swing.BorderFactory.createTitledBorder(""));

        javax.swing.GroupLayout jPanel5Layout = new javax.swing.GroupLayout(jPanel5);
        jPanel5.setLayout(jPanel5Layout);
        jPanel5Layout.setHorizontalGroup(
            jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel5Layout.createSequentialGroup()
                .addComponent(jLabel5)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(labelTotal, javax.swing.GroupLayout.PREFERRED_SIZE, 111, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        jPanel5Layout.setVerticalGroup(
            jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, jPanel5Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                .addComponent(labelTotal, javax.swing.GroupLayout.DEFAULT_SIZE, 28, Short.MAX_VALUE)
                .addComponent(jLabel5, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        panelOrders.setEditable(false);
        panelOrders.setColumns(20);
        panelOrders.setRows(5);
        panelOrders.setText("                              \n                              Receipt\n-------------------------------------------------------------");
        jScrollPane1.setViewportView(panelOrders);

        javax.swing.GroupLayout panelBackgroundLayout = new javax.swing.GroupLayout(panelBackground);
        panelBackground.setLayout(panelBackgroundLayout);
        panelBackgroundLayout.setHorizontalGroup(
            panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(panelBackgroundLayout.createSequentialGroup()
                .addGap(38, 38, 38)
                .addGroup(panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(panelBackgroundLayout.createSequentialGroup()
                        .addComponent(jLabel8)
                        .addGap(0, 0, Short.MAX_VALUE))
                    .addGroup(panelBackgroundLayout.createSequentialGroup()
                        .addComponent(btnAddItem, javax.swing.GroupLayout.PREFERRED_SIZE, 111, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(31, 31, 31)
                        .addComponent(btnClearForm, javax.swing.GroupLayout.PREFERRED_SIZE, 111, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                        .addComponent(btnClearOrder, javax.swing.GroupLayout.PREFERRED_SIZE, 111, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addContainerGap())
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, panelBackgroundLayout.createSequentialGroup()
                        .addGroup(panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
                            .addComponent(panelExtras, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                            .addComponent(panelDimensions, javax.swing.GroupLayout.Alignment.LEADING, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                        .addGap(33, 33, 33)
                        .addGroup(panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jScrollPane1, javax.swing.GroupLayout.DEFAULT_SIZE, 287, Short.MAX_VALUE)
                            .addComponent(jPanel5, javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                        .addGap(12, 12, 12))))
        );
        panelBackgroundLayout.setVerticalGroup(
            panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, panelBackgroundLayout.createSequentialGroup()
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jLabel8)
                .addGap(18, 18, 18)
                .addGroup(panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
                    .addGroup(panelBackgroundLayout.createSequentialGroup()
                        .addComponent(panelDimensions, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addGap(18, 18, 18)
                        .addComponent(panelExtras, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, panelBackgroundLayout.createSequentialGroup()
                        .addComponent(jScrollPane1)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
                        .addComponent(jPanel5, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addGap(34, 34, 34)
                .addGroup(panelBackgroundLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(btnAddItem, javax.swing.GroupLayout.PREFERRED_SIZE, 45, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnClearForm, javax.swing.GroupLayout.PREFERRED_SIZE, 45, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(btnClearOrder, javax.swing.GroupLayout.PREFERRED_SIZE, 44, javax.swing.GroupLayout.PREFERRED_SIZE))
                .addGap(268, 268, 268))
        );

        javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(panelBackground, javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(panelBackground, javax.swing.GroupLayout.PREFERRED_SIZE, 582, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        pack();
      }// </editor-fold>
       /**
        * Starts validation and stored inputs of the box
        * @param evt
        */
       private void btnAddItemActionPerformed(java.awt.event.ActionEvent evt) {
           //gets values from fields in form
           double size;
           int grade = (int)spinGrade.getValue();
           int color = comboColour.getSelectedIndex();
           boolean sealable = checkTop.isSelected();
           boolean bottom = checkBottom.isSelected();
           boolean corner = checkCorner.isSelected();
           int quantity = (int)spinQuantity.getValue();

           //validation to avoid number format exception
           try{
                width = Double.parseDouble(textWidth.getText());
                length = Double.parseDouble(textLength.getText());
                height = Double.parseDouble(textHeight.getText());

                //checking if size is within required range 0.5 to 10
                if (width < 0.5 || width > 10 || length < 0.5 || length > 10 || height < 0.5 || height > 10) {
                    JOptionPane.showMessageDialog(null, "The width/length/height must be a number between 0.5M to 
        10M.",
                        "Invalid Input", JOptionPane.ERROR_MESSAGE);
                }

                else {
                    //working out the size of the box
                    size = ((width * length) + (width * height) + (length * height)) * 2;
                    newBox(grade, color, bottom, corner, sealable, size, quantity);
                }

           //return error if exception found
           }catch(NumberFormatException | HeadlessException e){
                JOptionPane.showMessageDialog(null, "The size must be a positive number.",
                        "Invalid Input", JOptionPane.ERROR_MESSAGE);
           }
       }
       /**
        * Creates a new box with the values passed
        * @param grade grade of the box
        * @param colour number of colours
        * @param bottom if reinforced bottoms or not
        * @param corner if reinforced corner or not
        * @param sealable if sealable top or not
        * @param size size of box (w*l*h)
        * @param quantity  how many boxes
        */
       public void newBox(int grade, int colour, boolean bottom, boolean corner,
                        boolean sealable, double size, int quantity){

                double price;

                //checks which type the box is by using if statements to match requirements
                if(colour == 0 && !bottom && !corner && grade >= 1 && grade <= 3){
                        Box box = new BoxType1(grade, size, sealable); //creates box type 1
                        price = box.getPrice(); //Gets price from abstract method
                        boxType = "1";
                        printReceipt(boxType, size, grade, colour, bottom, corner, sealable, quantity, price);
                }else if(colour == 1 && !bottom && !corner && grade >= 2 && grade <= 4){
                        Box box = new BoxType2(grade, size, sealable ); //creates box type 2
                        price = box.getPrice(); //Gets price from abstract method
                        boxType = "2";
                        printReceipt(boxType, size, grade, colour, bottom, corner, sealable, quantity, price);
                }else if(colour == 2 && !bottom && !corner && grade >= 2 && grade <= 5){
                        Box box = new BoxType3(grade, size, sealable); //creates box type 3
                        price = box.getPrice(); //Gets price from abstract method
                        boxType = "3";
                        printReceipt(boxType, size, grade, colour, bottom, corner, sealable, quantity, price);
                }else if(colour == 2 && bottom && !corner && grade >= 2 && grade <= 5){
                        Box box = new BoxType4(grade, size, sealable); //creates box type 4
                        price = box.getPrice(); //Gets price from abstract method
                        boxType = "4";
                        printReceipt(boxType, size, grade, colour, bottom, corner, sealable, quantity, price);
                }else if(colour == 2 && bottom && corner && grade >= 3 && grade <= 5){ //creates box type 5
                        Box box = new BoxType5(grade, size, sealable);
                        price = box.getPrice(); //Gets price from abstract method
                        boxType = "5";
                        printReceipt(boxType, size, grade, colour, bottom, corner, sealable, quantity, price);
                }

                // if the box doesn't match any type of box, an error is given to the user with tips
                else {
                  JOptionPane.showMessageDialog(null, " Flexbox doesn't produce any boxs that match these "
                          + "requirements. \n \n Tip: Boxes with higher grade allow more extra features "
                          + "(Colours, \n Reinforced Corners, & Reinforced Bottoms).", ""
                                  + " Invalid Box, Please Try Again",JOptionPane.ERROR_MESSAGE);
                }
           }

           /**
            * Prints the receipt in the output section
            *
            * @param boxType type of box from 1 to 5
            * @param size size of box
            * @param grade grade of box
            * @param colour how many colours
            * @param bottom if reinforced bottom
            * @param corner if reinforced corner
            * @param sealable if sealable top
            * @param quantity how many boxes
            * @param price price of the box order
            */
           public void printReceipt(String boxType, double size, int grade, int colour,
                    boolean bottom, boolean corner, boolean sealable, int quantity, double price) {
                panelOrders.setText(panelOrders.getText() + "\n Box Type: " + boxType); //For demo and debugging
        purposes
```

```
        panelOrders.setText(panelOrders.getText() + "\n Size: " + width + " X " + length + " X " + height);
        panelOrders.setText(panelOrders.getText() + "\n Grade: " + grade);
        panelOrders.setText(panelOrders.getText() + "\n Colour: " + colour);

        if (bottom) { panelOrders.setText(panelOrders.getText() + "\n     Reinforced Bottom Included"); }

        if (corner) { panelOrders.setText(panelOrders.getText() + "\n     Reinforced Corner Included"); }

        if (sealable) { panelOrders.setText(panelOrders.getText() + "\n     Sealable Top Included"); }

        panelOrders.setText(panelOrders.getText() + "\n Quantity: " + quantity);
        price = price * quantity;
        DecimalFormat dpForm = new DecimalFormat("0.00");//rounds the number to 2 decimal places
        panelOrders.setText(panelOrders.getText() + "\n Box Total: £" + dpForm.format(price) + "\n");
        panelOrders.setText(panelOrders.getText() + "-------------------------------------------------------------\n");

        total = price + total;
        total = (double) (Math.round(total*100))/100;
        labelTotal.setText("£" + (dpForm.format(total)));

    }

    /**
     * Clears the orders and resets the total
     * @param evt
     */
    private void btnClearOrderActionPerformed(java.awt.event.ActionEvent evt) {
        // clears order
        panelOrders.setText("");
        total = 0;
        labelTotal.setText("");
    }

    /** Resets the form by clearing all input fields
     *
     * @param evt
     */
    private void btnClearFormActionPerformed(java.awt.event.ActionEvent evt) {
        //reset entire form
        textWidth.setText("");
        textLength.setText("");
        textHeight.setText("");
        checkBottom.setSelected(false);
        checkCorner.setSelected(false);
        checkTop.setSelected(false);
        spinQuantity.setValue(1);
        spinGrade.setValue(1);
        comboColour.setSelectedIndex(0);
    }

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
         * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        try {
            for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(UserInterface.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new UserInterface().setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JButton btnAddItem;
    private javax.swing.JButton btnClearForm;
    private javax.swing.JButton btnClearOrder;
    private javax.swing.JCheckBox checkBottom;
    private javax.swing.JCheckBox checkCorner;
    private javax.swing.JCheckBox checkTop;
    private javax.swing.JComboBox<String> comboColour;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel13;
    private javax.swing.JLabel jLabel14;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel5;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel labelTotal;
    private javax.swing.JPanel panelBackground;
    private javax.swing.JPanel panelDimensions;
    private javax.swing.JPanel panelExtras;
    private javax.swing.JTextArea panelOrders;
    private javax.swing.JSpinner spinGrade;
    private javax.swing.JSpinner spinQuantity;
    private javax.swing.JTextField textHeight;
    private javax.swing.JTextField textLength;
    private javax.swing.JTextField textWidth;
    // End of variables declaration
}
```