

```

1  /*****
2  *
3  * File: CoffeeOrders.java
4  *
5  * Author: Joshua Wiley
6  *
7  * Description: Finds all calculations for a coffee shop
8  *
9  * Date: 4-24-15
10 *
11 * Comments: It is cheaper to send 1 medium box then 2 small boxes i.e. 8 bags
12 *           ordered would be shipped in 1 medium box not 2 small boxes.
13 *
14 *****/
15
16 public class CoffeeOrders
17 {
18     /** Class Constants */
19
20     private static final int LARGE_BOX = 20;
21     private static final int MEDIUM_BOX = 10;
22     private static final int SMALL_BOX = 5;
23
24     public static final double LARGE_BOX_PRICE = 1.80;
25     public static final double MEDIUM_BOX_PRICE = 1.00;
26     public static final double SMALL_BOX_PRICE = 0.60;
27     private static final int MAX_QUANTITY = 1500;
28
29     private static final double PRICE_PER_BAG = 5.50;
30
31     /** Class Methods */
32
33     public static double calculatePurchasePrice( int bagsOrdered )
34     {
35         /** local Variables */
36
37         double purchasePrice = 0.0;
38
39         /** Calculates the purchase price before shipping */
40
41         purchasePrice = PRICE_PER_BAG * bagsOrdered;
42
43         return purchasePrice;
44     }
45
46     public static double totalPrice( int bagsOrdered, int large, int medium, int small )
47     {
48         /** local Variables */
49
50         double totalPrice = 0.0;
51
52         /** Calculates the total purchase price with shipping */
53
54         totalPrice += calculatePurchasePrice( bagsOrdered );
55         totalPrice += boxSizeCost( large, LARGE_BOX_PRICE );
56         totalPrice += boxSizeCost( medium, MEDIUM_BOX_PRICE );
57         totalPrice += boxSizeCost( small, SMALL_BOX_PRICE );
58
59         return totalPrice;
60     }
61
62     public static int largeBoxesNeeded( int bagsToBox )
63     {
64         /** local Variables */
65
66         int boxes = 0;
67
68         boxes = bagsToBox / LARGE_BOX;
69
70         return boxes;

```

```

71 }
72
73 public static int mediumBoxesNeeded( int bagsToBox )
74 {
75     /** local Variables */
76
77     int boxes = 0;
78
79     boxes = ( bagsToBox % LARGE_BOX ) / MEDIUM_BOX;
80
81     if ( ( bagsToBox % LARGE_BOX ) % MEDIUM_BOX > SMALL_BOX )
82         boxes ++; //It is cheaper to send 1 medium then 2 small
83
84     return boxes;
85 }
86
87 public static int smallBoxesNeeded( int bagsToBox )
88 {
89     /** local Variables */
90
91     int boxes = 0;
92
93     boxes = ( ( ( bagsToBox % LARGE_BOX ) % MEDIUM_BOX ) );
94     if ( boxes > SMALL_BOX )
95         boxes = 0; //It is cheaper to send 1 medium then 2 small
96     else if ( boxes > 0 )
97         boxes = 1;
98     else
99         boxes = 0;
100
101     return boxes;
102 }
103
104 public static double boxSizeCost( int numberOfBoxes, double boxPrice )
105 {
106     /** local Variables */
107
108     double price = 0;
109
110     /** Calculates shipping */
111
112     price = numberOfBoxes * boxPrice;
113
114     return price;
115 }
116
117 public static boolean validateString( String newString )
118 {
119     boolean status = false;
120     //Makes sure the string is not empty
121     if ( newString.trim().length() > 1 )
122         status = true;
123
124     return status;
125 }
126
127 public static boolean validateState( String newString )
128 {
129     boolean status = false;
130     int i = 0; //Initializes for loop
131
132     String states[] = { "AL", "AK", "AZ", "AR", "CA",
133                        "CO", "CT", "DE", "FL", "GA",
134                        "HI", "ID", "IL", "IN", "IA",
135                        "KS", "KY", "LA", "MA", "MI",
136                        "ME", "MD", "MN", "MS", "MO",
137                        "MT", "NE", "NV", "NH", "NJ",
138                        "NM", "NY", "NC", "ND", "OH",
139                        "OK", "OR", "PA", "RI", "SC",
140                        "SD", "TN", "TX", "UT", "VT",

```

```

141         "VA", "WA", "WV", "WI", "WY" };
142
143     for ( i = 0; i < states.length; i++ )
144     {
145         //Makes sure 2 letter abbr. is an actual state
146         if ( newString.equals( states[ i ] ) )
147             status = true;
148     }
149     return status;
150 }
151
152 public static boolean validateQuantity( int Quantity )
153 {
154     boolean status = false;
155
156     if ( ( Quantity >= 1 ) && ( Quantity <= MAX_QUANTITY ) )
157         status = true;
158
159     return status;
160 }
161
162 public static boolean validateZipcode( int Zipcode )
163 {
164     boolean status = false;
165
166     if ( ( Zipcode >= 01001 ) && ( Zipcode <= 99950 ) )
167         status = true;
168
169     return status;
170 }
171 }

```