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

```
72
73
         public static int mediumBoxesNeeded( int bagsToBox )
74
             /*** local Variables ***/
75
76
77
             int boxes = 0;
78
             boxes = ( bagsToBox % LARGE_BOX ) / MEDIUM_BOX;
79
80
             if ( ( bagsToBox % LARGE_BOX ) % MEDIUM_BOX > SMALL_BOX )
81
                 boxes ++; //It is cheaper to send 1 medium then 2 small
82
83
             return boxes;
84
         }
85
86
87
         public static int smallBoxesNeeded( int bagsToBox )
88
             /*** local Variables ***/
89
90
91
             int boxes = 0;
92
             boxes = ( ( ( bagsToBox % LARGE_BOX ) % MEDIUM_BOX ) );
93
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
         public static double boxSizeCost( int numberOfBoxes, double boxPrice )
104
105
             /*** local Variables ***/
106
107
             double price = 0;
108
109
             /*** Calculates shipping ***/
110
111
             price = numberOfBoxes * boxPrice;
112
113
             return price;
114
         }
115
116
         public static boolean validateString( String newString )
117
118
119
             boolean status = false;
                             //Makes sure the string is not empty
120
             if ( newString.trim().length() > 1 )
121
122
                 status = true;
123
124
             return status;
         }
125
126
         public static boolean validateState( String newString )
127
128
129
             boolean status = false;
130
             int i = 0; //Initializes for loop
131
             132
133
134
135
136
137
138
139
140
```

}

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