

```

package Assignment6;

import java.math.BigDecimal;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class CartCheckout {

    public static void main(String[] args) {

        Scanner scanner = new Scanner(System.in);
        String input = scanner.nextLine().trim();
        Map<String, BigDecimal> cart = parseCartInput(input);
        Double taxPercent = parseTaxPercent(input);
        CartCheckout checkout = new CartCheckout();
        String result = checkout.billGenerator(cart, taxPercent);
        System.out.println(result);
    }

    public String billGenerator(Map<String, BigDecimal> cart, Double taxPercent){
        // for validate input Map
        if(cart == null || cart.isEmpty()){
            return "The cart Map is empty";
        }

        // for validate taxPercent
        if(taxPercent == null){
            return "The taxPercent cannot be null";
        } else if (taxPercent < 0){
            return "The taxPercent cannot be negative";
        }
    }
}

```

```

// for validate Map values

boolean invalidValue = cart.values().stream()

    .anyMatch(value -> value == null || value.compareTo(BigDecimal.ZERO) <= 0);

if(invalidValue){

    return "The cart contains invalid items";

}

// to calculate total including tax

BigDecimal total = cart.values().stream()

    .reduce(BigDecimal.ZERO, BigDecimal::add);

BigDecimal taxMultiplier = BigDecimal.valueOf(1 + (taxPercent/100));

BigDecimal totalWithTax = total.multiply(taxMultiplier);

return totalWithTax.toString();

}

private static Double parseTaxPercent(String input){

    // to extract the tax percentage part from the input

    int splitIndex = input.lastIndexOf('}');

    String taxPercentString = input.substring(splitIndex + 2).trim();

    // to handle the case where taxPercent is null

    if ("null".equalsIgnoreCase(taxPercentString)) {

        return null;

    }

    return Double.parseDouble(taxPercentString);

}

```

```

private static Map<String, BigDecimal> parseCartInput(String input){
    Map<String, BigDecimal> cart = new HashMap<>();

    // to extract cart part from the input
    int splitIndex = input.lastIndexOf('}');
    String cartInput = input.substring(1, splitIndex).trim();

    // to handle the case where the cart is empty
    if(cartInput.isEmpty()){
        return cart;
    }
    String[] items = cartInput.split(",");
    for(String item : items){
        String[] keyValue = item.split("=");
        String itemName = keyValue[0].trim();
        BigDecimal itemPrice = new BigDecimal(keyValue[1].trim());
        cart.put(itemName, itemPrice);
    }
    return cart;
}
}

```

eclipse-workspace2 - Assignments/src/Assignment6/CartCheckout.java - Eclipse IDE

File Edit Source Refactor Source Navigate Search Project Run Window Help

CartCheckout.java ×

```
1 package Assignment6;
2 import java.math.BigDecimal;
3 import java.util.HashMap;
4 import java.util.Map;
5 import java.util.Scanner;
6
7 public class CartCheckout {
8     public static void main(String[] args) {
9
10         Scanner scanner = new Scanner(System.in);
11         String input = scanner.nextLine().trim();
12         Map<String, BigDecimal> cart = parseCartInput(input);
13         Double taxPercent = parseTaxPercent(input);
14         CartCheckout checkout = new CartCheckout();
15         String result = checkout.billGenerator(cart, taxPercent);
16         System.out.println(result);
17     }
18
19     public String billGenerator(Map<String, BigDecimal> cart, Double taxPercent){
20         // for validate input Map
21         if(cart == null || cart.isEmpty()){
22             return "The cart Map is empty";
23         }
24
25         // for validate taxPercent
26         if(taxPercent == null){
27             return "The taxPercent cannot be null";
28         } else if (taxPercent < 0){
29             return "The taxPercent cannot be negative";
30         }
31
32         // for validate Map values
33         boolean invalidValue = cart.values().stream()
34             .anyMatch(value -> value == null || value.compareTo(BigDecimal.ZERO) <= 0);
35         if(invalidValue){
36             return "The cart contains invalid items";
37         }
38
39         // to calculate total including tax
40         BigDecimal total = cart.values().stream()
41             .reduce(BigDecimal.ZERO, BigDecimal::add);
42         BigDecimal taxMultiplier = BigDecimal.valueOf(1 + (taxPercent/100));
43         BigDecimal result = total.multiply(taxMultiplier);
44         return result.toString();
45     }
46
47     private Map<String, BigDecimal> parseCartInput(String input){
48         Map<String, BigDecimal> cart = new HashMap<>();
49         if(input.isEmpty()) return cart;
50         String[] items = input.split(",");
51         for(String item : items){
52             String[] parts = item.split(":");
53             String itemStr = parts[0].trim();
54             double value = Double.parseDouble(parts[1].trim());
55             cart.put(itemStr, new BigDecimal(value));
56         }
57         return cart;
58     }
59
60     private Double parseTaxPercent(String input){
61         if(input.isEmpty()) return null;
62         String[] parts = input.split(":");
63         double taxPercent = Double.parseDouble(parts[0].trim());
64         return taxPercent;
65     }
66 }
```

Console ×

```
<terminated> CartCheckout [Java Application] C:\Users\vinee\p2\pool\plugins\org.eclipse.justi.openjdk hotspot.jre.full.win32.x86_64_22.0.1.v...
{Apple=54, Grapes=36.78, Papaya=27.89, Orange=23.6, Banana=10.2},10.5
168.47935
```

eclipse-workspace2 - Assignments/src/Assignment6/CartCheckout.java - Eclipse IDE

File Edit Source Refactor Source Navigate Search Project Run Window Help

CartCheckout.java ×

```
1 package Assignment6;
2 import java.math.BigDecimal;
3 import java.util.HashMap;
4 import java.util.Map;
5 import java.util.Scanner;
6
7 public class CartCheckout {
8     public static void main(String[] args) {
9
10         Scanner scanner = new Scanner(System.in);
11         String input = scanner.nextLine().trim();
12         Map<String, BigDecimal> cart = parseCartInput(input);
13         Double taxPercent = parseTaxPercent(input);
14         CartCheckout checkout = new CartCheckout();
15         String result = checkout.billGenerator(cart, taxPercent);
16         System.out.println(result);
17     }
18
19     public String billGenerator(Map<String, BigDecimal> cart, Double taxPercent){
20         // for validate input Map
21         if(cart == null || cart.isEmpty()){
22             return "The cart Map is empty";
23         }
24
25         // for validate taxPercent
26         if(taxPercent == null){
27             return "The taxPercent cannot be null";
28         } else if (taxPercent < 0){
29             return "The taxPercent cannot be negative";
30         }
31
32         // for validate Map values
33         boolean invalidValue = cart.values().stream()
34             .anyMatch(value -> value == null || value.compareTo(BigDecimal.ZERO) <= 0);
35         if(invalidValue){
36             return "The cart contains invalid items";
37         }
38
39         // to calculate total including tax
40         BigDecimal total = cart.values().stream()
41             .reduce(BigDecimal.ZERO, BigDecimal::add);
42         BigDecimal taxMultiplier = BigDecimal.valueOf(1 + (taxPercent/100));
43         BigDecimal result = total.multiply(taxMultiplier);
44         return result.toString();
45     }
46
47     private Map<String, BigDecimal> parseCartInput(String input){
48         Map<String, BigDecimal> cart = new HashMap<>();
49         if(input.isEmpty()) return cart;
50         String[] items = input.split(",");
51         for(String item : items){
52             String[] parts = item.split(":");
53             String itemStr = parts[0].trim();
54             double value = Double.parseDouble(parts[1].trim());
55             cart.put(itemStr, new BigDecimal(value));
56         }
57         return cart;
58     }
59
60     private Double parseTaxPercent(String input){
61         if(input.isEmpty()) return null;
62         String[] parts = input.split(":");
63         double taxPercent = Double.parseDouble(parts[0].trim());
64         return taxPercent;
65     }
66 }
```

Console ×

```
<terminated> CartCheckout [Java Application] C:\Users\vinee\p2\pool\plugins\org.eclipse.justi.openjdk hotspot.jre.full.win32.x86_64_22.0.1.v...
{},{13
The cart Map is empty
```

11 Writable Smart Insert 37 : 10 : 1322

eclipse-workspace2 - Assignments/src/Assignment6/CartCheckout.java - Eclipse IDE

File Edit Source Refactor Source Navigate Search Project Run Window Help

CartCheckout.java

```
1 package Assignment6;
2 import java.math.BigDecimal;
3 import java.util.HashMap;
4 import java.util.Map;
5 import java.util.Scanner;
6
7 public class CartCheckout {
8     public static void main(String[] args) {
9
10         Scanner scanner = new Scanner(System.in);
11         String input = scanner.nextLine().trim();
12         Map<String, BigDecimal> cart = parseCartInput(input);
13         Double taxPercent = parseTaxPercent(input);
14         CartCheckout checkout = new CartCheckout();
15         String result = checkout.billGenerator(cart, taxPercent);
16         System.out.println(result);
17     }
18
19     public String billGenerator(Map<String, BigDecimal> cart, Double taxPercent){
20         // for validate input Map
21         if(cart == null || cart.isEmpty()){
22             return "The cart Map is empty";
23         }
24
25         // for validate taxPercent
26         if(taxPercent == null){
27             return "The taxPercent cannot be null";
28         } else if (taxPercent < 0){
29             return "The taxPercent cannot be negative";
30         }
31
32         // for validate Map values
33         boolean invalidValue = cart.values().stream()
34             .anyMatch(value -> value == null || value.compareTo(BigDecimal.ZERO) <= 0);
35         if(invalidValue){
36             return "The cart contains invalid items";
37         }
38
39         // to calculate total including tax
40         BigDecimal total = cart.values().stream()
41             .reduce(BigDecimal.ZERO, BigDecimal::add);
42         BigDecimal taxMultiplier = BigDecimal.valueOf(1 + (taxPercent/100));
```

Console

```
<terminated> CartCheckout [Java Application] C:\Users\vineel.p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.22.0.1.v...
[Apple=54, Grapes=36.78, Papaya=27.89, Orange=23.6, Banana=10.2],-2.5
The taxPercent cannot be negative
```

11 Writable Smart Insert 53:43:2022

eclipse-workspace2 - Assignments/src/Assignment6/CartCheckout.java - Eclipse IDE

File Edit Source Refactor Source Navigate Search Project Run Window Help

CartCheckout.java

```
1 package Assignment6;
2 import java.math.BigDecimal;
3 import java.util.HashMap;
4 import java.util.Map;
5 import java.util.Scanner;
6
7 public class CartCheckout {
8     public static void main(String[] args) {
9
10         Scanner scanner = new Scanner(System.in);
11         String input = scanner.nextLine().trim();
12         Map<String, BigDecimal> cart = parseCartInput(input);
13         Double taxPercent = parseTaxPercent(input);
14         CartCheckout checkout = new CartCheckout();
15         String result = checkout.billGenerator(cart, taxPercent);
16         System.out.println(result);
17     }
18
19     public String billGenerator(Map<String, BigDecimal> cart, Double taxPercent){
20         // for validate input Map
21         if(cart == null || cart.isEmpty()){
22             return "The cart Map is empty";
23         }
24
25         // for validate taxPercent
26         if(taxPercent == null){
27             return "The taxPercent cannot be null";
28         } else if (taxPercent < 0){
29             return "The taxPercent cannot be negative";
30         }
31
32         // for validate Map values
33         boolean invalidValue = cart.values().stream()
34             .anyMatch(value -> value == null || value.compareTo(BigDecimal.ZERO) <= 0);
35         if(invalidValue){
36             return "The cart contains invalid items";
37         }
38
39         // to calculate total including tax
40         BigDecimal total = cart.values().stream()
41             .reduce(BigDecimal.ZERO, BigDecimal::add);
42         BigDecimal taxMultiplier = BigDecimal.valueOf(1 + (taxPercent/100));
```

Console

```
<terminated> CartCheckout [Java Application] C:\Users\vineel.p2\pool\plugins\org.eclipse.justi.openjdk.hotspot.jre.full.win32.x86_64.22.0.1.v...
[Apple=54, Grapes=36.78, Papaya=27.89, Orange=23.6, Banana=10.2],null
The taxPercent cannot be null
```

11 Writable Smart Insert 34:70:1211