```java
1 public interface Food {
2     void Order();
3     float Cost();
4 }
5
```

```java
1  import java.util.Vector;
2
3  public class Main {
4      public static void main(String[] args) {
5          Customer customer = new Customer("Jason", CustomerStatus.PREMIUM);
6
7          Vector<BurgerToppings> burger_1_toppings = new Vector<>();
8          burger_1_toppings.add(BurgerToppings.CHEESE);
9          burger_1_toppings.add(BurgerToppings.PICKLES);
10         customer.AddBurger(burger_1_toppings);
11
12         Vector<BurgerToppings> burger_2_toppings = new Vector<>();
13         burger_2_toppings.add(BurgerToppings.CHEESE);
14         customer.AddBurger(burger_2_toppings);
15
16         customer.AddDrink(DrinkFlavour.PEPSI);
17         customer.AddFries(FriesType.WAFFLE);
18
19         CustomerLoyaltyDiscount.Discount(customer);
20
21         customer.Order();
22     }
23 }
24
```

```java
 1 public class Drink implements Food {
 2     @Override
 3     public void Order() {
 4         System.out.println("Drink");
 5     }
 6
 7     @Override
 8     public float Cost() {
 9         return 1.25f;
10     }
11 }
12
```

```java
public class Fries implements Food {
    @Override
    public void Order() {
        System.out.println("Fries");
    }

    @Override
    public float Cost() {
        return 1.0f;
    }
}
```

```java
 1 public class Burger implements Food {
 2     @Override
 3     public void Order() {
 4         System.out.println("Burger");
 5     }
 6
 7     @Override
 8     public float Cost() {
 9         return 2.0f;
10     }
11 }
12
```

```java
1  import java.util.Vector;
2
3  public class Customer {
4      private Vector<Food> foodList;
5      private final String name;
6      CustomerStatus status;
7      private float totalCost = 0.0f;
8      private float discount = 0.0f;
9
10     Customer(String customer_name, CustomerStatus customer_status) {
11         name = customer_name;
12         foodList = new Vector<>();
13         status = customer_status;
14     }
15
16     void AddBurger(Vector<BurgerToppings> toppings) {
17         Food burger = new Burger();
18
19         for (BurgerToppings topping : toppings) {
20             if (topping == BurgerToppings.CHEESE) {
21                 burger = new BurgerWithCheeseDecorator(burger);
22             }
23
24             if (topping == BurgerToppings.PICKLES) {
25                 burger = new BurgerWithPicklesDecorator(burger);
26             }
27         }
28
29         foodList.add(burger);
30     }
31
32     void AddDrink(DrinkFlavour flavour) {
33         Food drink = new Drink();
34
35         if (flavour == DrinkFlavour.PEPSI) {
36             drink = new DrinkPepsiDecorator(drink);
37         }
38
39         foodList.add(drink);
40     }
41
42     void AddFries(FriesType fries_type) {
43         Food fries = new Fries();
44
45         if (fries_type == FriesType.WAFFLE) {
46             fries = new FriesWaffleDecorator(fries);
47         }
48
49         foodList.add(fries);
50     }
51
52     CustomerStatus GetStatus() {
53         return status;
54     }
55
56     void AddDiscount(float discount) {
57         this.discount = discount;
58     }
59
60     void Order() {
61         for (Food food : foodList) {
62             food.Order();
63             totalCost += food.Cost();
64         }
65
66         System.out.println("\nFood Total Cost for " + name + ": $" + (totalCost-discount
   ) + " (original cost: $" + totalCost + ")");
```

```
67        }
68 }
69
```

```java
1 public enum FriesType {
2     WAFFLE;
3 }
4
```

```java
1 public enum DrinkFlavour {
2     PEPSI;
3 }
4
```

```java
1  public abstract class FoodDecorator implements Food {
2      protected Food foodItem;
3      protected float cost = 0.0f;
4
5      FoodDecorator(Food food_item) {
6          foodItem = food_item;
7      }
8      @Override
9      public void Order() {
10         foodItem.Order();
11         cost += foodItem.Cost();
12     }
13
14     @Override
15     public float Cost() {
16         return cost;
17     }
18 }
19
```

```java
1  public abstract class FoodDecorator implements Food {
2      protected Food foodItem;
3      protected float cost = 0.0f;
4
5      FoodDecorator(Food food_item) {
6          foodItem = food_item;
7      }
8      @Override
9      public void Order() {
10         foodItem.Order();
11         cost += foodItem.Cost();
```

Page 1 of 1

```java
1 public enum BurgerToppings {
2     CHEESE,
3     PICKLES;
4 }
5
```

```java
1 public enum CustomerStatus {
2     NORMAL,
3     PREMIUM,
4     ROYAL;
5 }
6
```

```java
1  public class DrinkPepsiDecorator extends FoodDecorator {
2      DrinkPepsiDecorator(Food food_item) {
3          super(food_item);
4      }
5      @Override
6      public void Order() {
7          foodItem.Order();
8
9          cost += foodItem.Cost();
10         cost += 0.0;
11
12         System.out.println("+ Pepsi Drink");
13     }
14 }
15
```

```java
1  public class FriesWaffleDecorator extends FoodDecorator {
2      FriesWaffleDecorator(Food food_item) {
3          super(food_item);
4      }
5      @Override
6      public void Order() {
7          foodItem.Order();
8
9          cost += foodItem.Cost();
10         cost += 0.25;
11
12         System.out.println("+ Waffle Fries");
13     }
14 }
15
```

```java
 1 public class CustomerLoyaltyDiscount {
 2     static void Discount (Customer customer) {
 3
 4         if (customer.GetStatus() == CustomerStatus.PREMIUM) {
 5             System.out.println("Customer Premium Discount applied: $1.0");
 6             customer.AddDiscount(1.0f);
 7         }
 8         else if (customer.GetStatus() == CustomerStatus.ROYAL) {
 9             System.out.println("Customer Royal Discount applied: $1.25");
10             customer.AddDiscount(1.25f);
11         }
12     }
13 }
14
```

```java
public class BurgerWithCheeseDecorator extends FoodDecorator {
    BurgerWithCheeseDecorator(Food food_item) {
        super(food_item);
    }
    @Override
    public void Order() {
        foodItem.Order();

        cost += foodItem.Cost();
        cost += 1.0;

        System.out.println("+ Add Cheese");
    }
}
```

```java
1  public class BurgerWithPicklesDecorator extends FoodDecorator {
2
3      BurgerWithPicklesDecorator(Food food_item) {
4          super(food_item);
5      }
6      @Override
7      public void Order() {
8          foodItem.Order();
9
10         cost += foodItem.Cost();
11         cost += 0.5;
12
13         System.out.println("+ Add Pickles");
14     }
15 }
16
```