I’m going to create a class for the products, one for the discounts and one for the cart

I’m using a library to read the products and YAML file rulest

Then I can add products to the cart and apply different discounts so I can check the results and amounts in the cart

### DEFINING PRODUCT CLASS

public class Product {

private String code;

private String name;

private double price;

public Product(String code, String name, double price) {

this.code = code;

this.name = name;

this.price = price;

}

public String getCode() { return code; }

public String getName() { return name; }

public double getPrice() { return price; }

}

### DEFINING DiscountRule:

public interface DiscountRule { double apply(Product product, int quantity); }

#### 3 CLASSES TO IMPLEMENT THE DiscountRule:

public class FreeRule implements DiscountRule {

private int n;

public FreeRule(int n) { this.n = n; }

@Override public double apply(Product product, int quantity) {

int freeItems = quantity / n;

return (quantity - freeItems) \* product.getPrice();

}

}

public class ReducedPriceRule implements DiscountRule {

private int n;

private double reducedPrice;

public ReducedPriceRule(int n, double reducedPrice) {

this.n = n;

this.reducedPrice = reducedPrice;

}

@Override public double apply(Product product, int quantity) {

if (quantity > n) { return n \* product.getPrice() + (quantity - n) \* reducedPrice; }

else {

return quantity \* product.getPrice(); }

}

}

public class FractionPriceRule implements DiscountRule {

private int n;

private double fraction;

public FractionPriceRule(int n, double fraction) {

this.n = n;

this.fraction = fraction;

}

@Override public double apply(Product product, int quantity) {

if (quantity > n) { return n \* product.getPrice() + (quantity - n) \* product.getPrice() \* fraction; } else { return quantity \* product.getPrice(); }

}

}

### DEFINING Cart class (THIS ONE WILL HOLD THE PRODUCTS AND APPLY THE DISCOUNTS):

import java.util.\*;

public class Cart {

private Map<String, Product> products;

private Map<String, DiscountRule> rules;

private Map<String, Integer> quantities;

public Cart(Map<String, Product> products, Map<String, DiscountRule> rules) {

this.products = products; this.rules = rules; this.quantities = new HashMap<>();

}

public void addProduct(String productCode, int quantity) {

int currentQuantity = quantities.getOrDefault(productCode, 0);

quantities.put(productCode, currentQuantity + quantity);

}

public double getTotal() {

double total = 0.0;

for (Map.Entry<String, Integer> entry : quantities.entrySet()) {

String productCode = entry.getKey();

int quantity = entry.getValue();

Product product = products.get(productCode);

DiscountRule rule = rules.get(productCode);

if (rule != null) {

total += rule.apply(product, quantity);

} else {

total += product.getPrice() \* quantity; } } return total;

}

}

## TESTING THE RULES

import org.junit.Test;

import org.yaml.snakeyaml.TypeDescription;

import org.yaml.snakeyaml.Yaml;

import org.yaml.snakeyaml.constructor.Constructor;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.io.InputStream;

import java.util.List;

import static org.junit.Assert.assertEquals;

public class ShoppingCartTest {

@Test

public void testCalculateTotalPrice() {

// Read products from YAML file

Constructor constructor = new Constructor(Product.class);

TypeDescription typeDescription = new TypeDescription(Product.class);

constructor.addTypeDescription(typeDescription);

Yaml yaml = new Yaml(constructor);

List<Product> products;

try {

InputStream inputStream = new FileInputStream("priv/assets/products.yml");

products = yaml.load(inputStream);

} catch (FileNotFoundException e) {

System.err.println("Error: products.yml not found");

return;

}

// Read rules from YAML file

constructor = new Constructor(DiscountRule.class);

typeDescription = new TypeDescription(DiscountRule.class);

constructor.addTypeDescription(typeDescription);

yaml = new Yaml(constructor);

List<DiscountRule> rules;

try {

InputStream inputStream = new FileInputStream("priv/assets/rules.yml");

rules = yaml.load(inputStream);

} catch (FileNotFoundException e) {

System.err.println("Error: rules.yml not found");

return;

}

// Create shopping cart

Cart cart = new Cart();

// Add products to shopping cart

cart.addProduct(products.get(0)); // GR1 Green Tea £3.11

cart.addProduct(products.get(1)); // SR1 Strawberries £5.00

cart.addProduct(products.get(2)); // CF1 Coffee £11.23

// Add discounts to shopping cart

cart.addDiscount(rules.get(0)); // buy 1 get 1 free

cart.addDiscount(rules.get(1)); // buy more than 3 pay £4.50 per product

// assert total price

assertEquals(22.45, cart.calculateTotalPrice(), 0.001);

}

}