Documentation file

Odinakachukwu Nzekwe

Athabasca University

**Computer Science 308:**

Java for Programmers

COMP 308

**# Documentation file for Assignment Two Part 1 and Part 2**

## Overview

The code package implements an order processing system in Java. It comprises classes for defining products, orders, and an order processor. The system allows the creation of orders for various types of products and provides functionality for processing and dispatching these orders.

## Classes

### Product

- \*\*Description:\*\* Represents a generic product with attributes such as name and price.

- \*\*Attributes:\*\*

- `name`: String, the name of the product.

- `price`: double, the price of the product.

- \*\*Methods:\*\*

- `Product(String name, double price)`: Constructor to initialize the product with a name and price.

- `getName()`: Returns the name of the product.

- `getPrice()`: Returns the price of the product.

- `toString()`: Overrides the `toString()` method to provide a customized string representation of the product.

### ComputerPart, Peripheral, Cheese, Fruit, Service

- \*\*Description:\*\* Subclasses of Product representing specific types of products.

- \*\*Attributes:\*\* Inherits attributes from the Product class.

- \*\*Methods:\*\* Inherits methods from the Product class.

### GenericOrder\<T extends Product>

- \*\*Description:\*\* Generic class representing an order for a specific type of product.

- \*\*Attributes:\*\*

- `orderId`: int, a unique identifier for the order.

- `items`: List\<T>, a list of items in the order.

- \*\*Methods:\*\*

- `GenericOrder()`: Constructor to initialize the order with a unique order ID and an empty list of items.

- `getOrderId()`: Returns the order ID.

- `addItem(T item)`: Adds an item to the order.

- `getItems()`: Returns the list of items in the order.

### ComputerOrder, PartyTrayOrder

- \*\*Description:\*\* Subclasses of GenericOrder\<Product> representing specific types of orders.

- \*\*Methods:\*\* Additional methods specific to each type of order can be added here.

### OrderProcessor

- \*\*Description:\*\* Manages the processing and dispatching of orders.

- \*\*Attributes:\*\*

- `orderMap`: Map\<Integer, List\<Product>>, maps order IDs to lists of products.

- \*\*Methods:\*\*

- `OrderProcessor()`: Constructor to initialize the order map.

- `accept(GenericOrder<? extends Product> order)`: Accepts an order and adds it to the order map.

- `process()`: Processes orders by printing the items along with their order numbers.

- `dispatchComputerParts()`, `dispatchPeripherals()`, `dispatchCheeses()`, `dispatchFruits()`, `dispatchServices()`: Dispatches specific types of products by iterating over the order map and checking the type of each product.

- `getOrderNumber(Product product)`: Retrieves the order number for a given product.

### Main

- \*\*Description:\*\* Contains the main method to test the functionality of the OrderProcessor class.

- \*\*Methods:\*\*

- `main(String[] args)`: Creates orders, adds items to the orders, accepts orders, and processes them.

## Usage

1. \*\*Creating Orders:\*\*

- Instantiate objects of ComputerOrder or PartyTrayOrder classes.

- Add items to the orders using the `addItem()` method.

2. \*\*Processing Orders:\*\*

- Initialize an OrderProcessor object.

- Accept orders using the `accept()` method of OrderProcessor.

- Process orders using the `process()` method of OrderProcessor.

3. \*\*Dispatching Products:\*\*

- Use specific dispatch methods of OrderProcessor to dispatch Computer Parts, Peripherals, Cheeses, Fruits, or Services.

## Notes

- The system assumes that each order contains a unique set of items, even if the same item is added multiple times, it's considered as a separate entry in the order.

- The dispatch methods print out the products along with their order numbers, indicating that they are ready for dispatch.

This documentation provides detailed information about the classes, their attributes, methods, and usage, enabling users to understand and utilize the code package effectively for order processing and dispatching tasks.