**Vendor-Customer**

**Documentation**

**Overview**

The VendorCustomer class in the ShopVendor package simulates a simple shopping experience where a customer purchases items, pays the vendor, and receives change. The program calculates the total price of the items, accepts payment from the customer, and returns the minimum number of notes/coins needed for the change.

Class: VendorCustomer

main Method

The main method orchestrates the overall flow of the program. Here’s a step-by-step explanation:

**Initialization and Input:**

A Scanner object sc is created to read user input. The program prompts the user for the number of items they want to purchase. If the input is less than or equal to zero, the program requests a valid number of items. Two arrays are initialized: arr for item prices and sarr for item names. Item Details and Total Calculation:

* For each item, the user is prompted to enter the item name and price.
* If the entered price is less than or equal to zero, the program requests a valid price.
* The total amount is calculated by summing up the prices of all items.

**Payment and Change Calculation:**

The program prompts the customer to enter the amount they are paying. If the paid amount is less than the total amount, the program requests the additional amount needed. The change to be returned is calculated as the difference between the paid amount and the total amount.

**Change Distribution:**

The changeToBeGiven method is called to determine the minimum number of notes/coins required to make the change. The number of notes/coins and their denominations are printed.

**changeToBeGiven Method**

This method calculates the minimum number of notes/coins required for a given change amount.

**Parameters**:

change: The amount of change to be returned.

num\_of\_notes: A counter for the number of notes/coins.

**Logic**:

The method uses a while loop to repeatedly check and subtract the largest possible denomination from the change. The denominations are checked in descending order: 5000, 1000, 500, 200, 100, 50, 20, 10, 5, 2, 1. For each denomination, the method calculates how many notes/coins of that denomination are needed and prints the details. The remaining change is then updated by taking the modulus of the current change with the denomination. The process continues until the change becomes zero.

**Example**

Here's a step-by-step example to illustrate the program:

The user decides to buy 6 items.

They enter the item names and prices:

"pen" with price 12

"paper" with price 40

"pencil" with price 55

"scale" with price 90

"book" with price 100

"eraser" with price 55

The total amount is calculated as 352.

The user pays 300 initially.

The program prompts the user to give 52 more for the items purchased.

The user provides an additional 60, making the total payment 360.

The change to be returned is 8.

The changeToBeGiven method calculates and prints the denominations and number of notes/coins needed to make 8 as:

1 \* 5 = 5

1 \* 2 = 2

1 \* 1 = 1

The program outputs the total amount, the amount given, and the change rendered along with the minimum number of notes/coins (3 in this case).

This program provides a straightforward simulation of a purchase transaction and demonstrates basic control structures, array handling, and modular programming in Java.