Bachelor of Information Technology (Hons)

Assignment Cover Sheet

Course Code:_EC3121_________ Course Title:_OOP______________________

Assignment Title: __Billing system______ Due Date: _26th Aug 2023_________________

Date Submitted: _21th Aug 2023_____ Lecturer Name: _Ram Prasad Chaudhary______

To be completed if this is an individual assignment

I declare that this assignment is my individual work. I have not worked collaboratively nor have I copied from any other student's work or from any other source except where due acknowledgement is made explicitly in the text, nor has any part been written for me by another person.

Student Name:_ Amrita Tamang___________________ Student ID:_00020709_______________

Signature: __ [illegible] ______________

To be completed if this is a group assignment

We declare that this is a group assignment and that no part of this submission has been copied from any other student's work or from any other source except where due acknowledgement is made explicitly in the text, nor has any part been written for us by another person.

Student ID Student Name Signature

_________ ____ __________

_________ ____ __________

_________ ____ __________

_________ ____ __________

Lecturer's comments: ____

____

Total Marks: ______________ Lecturer's Signature:_________________

Feedback to Student:

I/We acknowledged receiving feedback from the lecturer on this assignment.

Student's Signature: ____________ _______

Assessed Learning Outcomes This assignment is designed to test your attainment of the following learning outcomes:

1.Develop an understanding of the problems in the form of requirements.

2.Demonstrate the knowledge of creating and instantiating programming solution using the programming constructs and methods of Java program.

3.Use of intermediate object oriented programming facilities

4.Design, implement, test and document computer programs.

Problem Specification:

You are required to create a simple billing system to take customer orders for a local handicraft store in Patan. The store is a start-up and offers limited products. We take 10 products for the sake of this system. You are required to create a prototype of billing system to ease the working process within the store. Your approach to solution needs to be object oriented. Your programming solution should create and implement suitable classes and objects. Programming solution with only main class will not qualify to become a submission for this task. Your program should have the following functionalities:

•The program should display the list of items on sale at the start (with 10 products) •Each customer will make an order based on the menu shown. For each order, the system should calculate the total bill.

•For each sale, 12% of the sold money goes to local charity. For each purchase made by the customer, your system should show the amount that goes for the charity, the amount that goes to the store and the total payment that the customer needs to make.

•For every new instance where a customer (new or repeating) makes an order, a new bill is generated.

•Altering a confirmed order would not be a necessary feature for the MVP phase of the system. •

Your MVP should be able to manage order and billing of 5 customers at once. The system should prompt (ask) to repeat after each billing.

•The menu prices and the dishes in them could be initially hardwired into your MVP for test purposes (in a file or an array)

[NILAI UNIVERSITY]

****Self-Declaration Page****

I, declare that the work presented in this assignment, titled "[Assignment Title]," is entirely my own. I have not used any unauthorized resources or assistance in completing this assignment, except where explicitly mentioned and properly cited.

I understand and acknowledge that any form of plagiarism or academic dishonesty is a serious violation of university policies and the principles of academic integrity. I am fully aware of the consequences of such actions, which may include academic penalties.

By signing this declaration, I affirm that:

1. The ideas, concepts, and code presented in this assignment are my original work, except where referenced appropriately.

2. I have not copied, reproduced, or used in any way the work of another person, whether published or unpublished, without proper citation.

3. Any external sources used for reference have been appropriately cited in accordance with the referencing style required for this assignment.

4. I have not shared, lent, or allowed others to access my work before submission.

I acknowledge that my assignment may be subjected to plagiarism detection software or other means of verification.

Contents

Introduction

In today's dynamic and fast-paced retail environment, efficient and user-friendly billing systems play a pivotal role in enhancing customer satisfaction and business operations. This assignment presents the development of a simple billing system tailored for a local handicraft store located in Patan. The objective of this project is to demonstrate our understanding of fundamental programming concepts, particularly in the Java programming language, and to showcase our ability to design, implement, test, and document a computer program in an object-oriented manner.

Problem Statement

The local handicraft store in Patan is a burgeoning start-up, offering a curated selection of handicraft products. To streamline their daily operations and provide a seamless shopping experience to their customers, our task is to create a prototype of a billing system. This system will serve as a Minimum Viable Product (MVP) that manages customer orders and calculates their bills.

Object-Oriented Approach

Our approach to solving this problem is rooted in object-oriented programming principles. We will design and implement suitable classes and objects that encapsulate the various aspects of the billing system. These classes will facilitate the creation of reusable, organized, and maintainable code.

Key Functionalities

The billing system will offer the following core functionalities:

- Product Listing: The system will display a list of items available for sale, initially comprising ten products.
- Order Management: Customers can place orders based on the displayed menu. For each order, the system will calculate the total bill.
- Charity Contribution: A philanthropic aspect is integrated into the system. For every sale, 12% of the total amount will be contributed to a local charity. The system will display the charity contribution, the store revenue, and the customer's total payment.
- Multiple Customers: The MVP should be capable of handling orders and billing for up to five customers concurrently. After each billing, the system will prompt the user for further orders.

- Order Isolation: For each new instance where a customer (whether new or returning) makes an order, a new bill is generated. Altering a confirmed order will not be a necessary feature for the MVP phase.

Initial Data Hardcoding

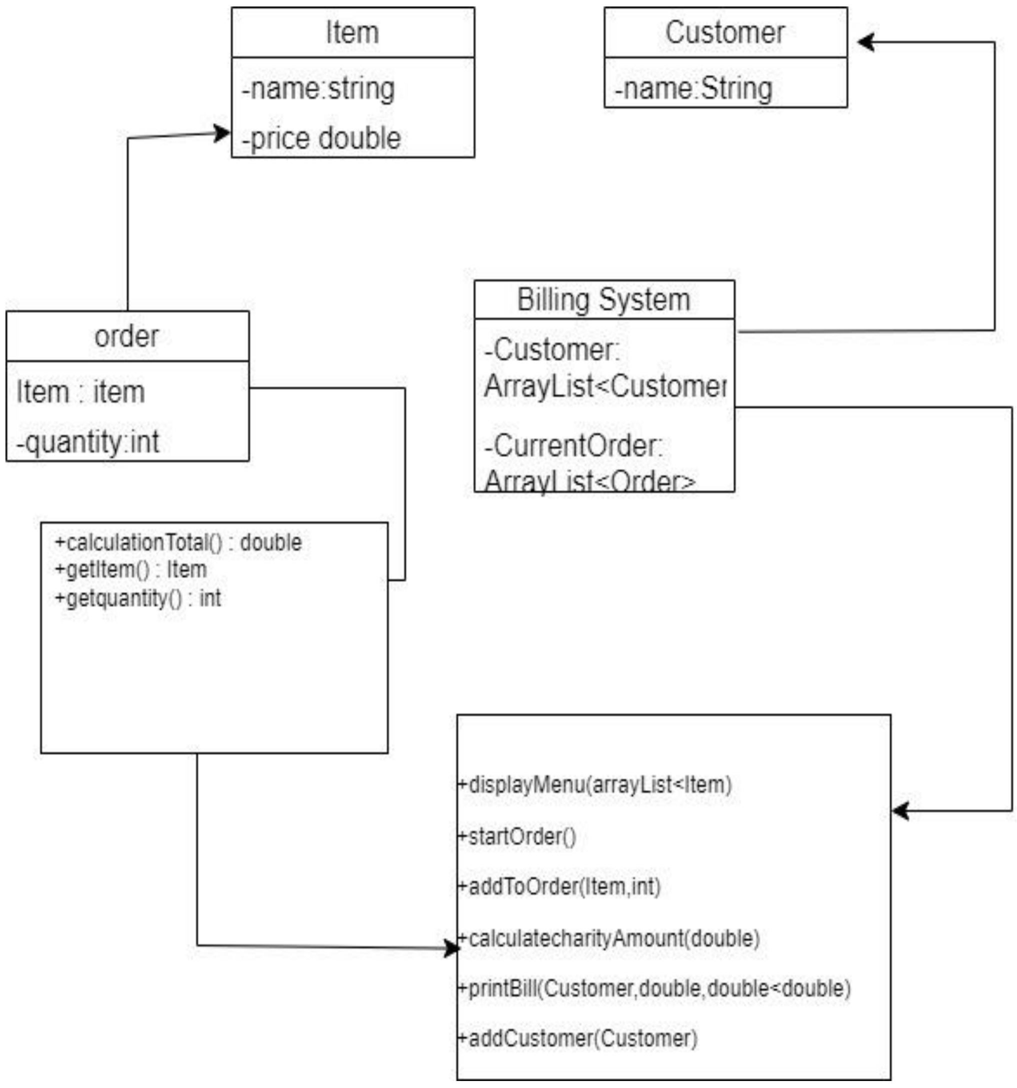
For the initial MVP, we will hardcode the menu prices and product details into the program, simplifying testing and demonstration purposes.

In the subsequent sections of this assignment, we will delve into the UML classes, detailed algorithm, test plan, and implementation code, providing a comprehensive insight into the creation and operation of this billing system.

This assignment not only serves as an educational exercise in programming but also emphasizes the real-world relevance of object-oriented design and software development in a retail context. By the end of this assignment, we aim to present a functional billing system that embodies the principles of object-oriented programming and can be readily extended for future enhancements.

UML classes

Billing System


Item
-name:string
-price double
Customer
-name:String
order
Item : item
-quantity:int
Billing System
-Customer: ArrayList<Customer
-CurrentOrder: ArrayList<Order>
+calculationTotal() : double
+getItem() : Item
+getquantity() : int
+displayMenu(arrayList<Item)
+startOrder()
+addToOrder(Item,int)
+calculatecharityAmount(double)
+printBill(Customer,double,double<double)
+addCustomer(Customer)

Algorithm for the Billing System

Initialize the System

- Create an empty list of customers.
- Create an empty list for the current order.

Add Customers

- Allow the user to add customers to the system. Repeat this step as needed.
- For each customer, record their name and add them to the list of customers.

Display Menu

- Create a list of items available for sale, including their names and prices.
- Display the menu to the user, listing each item with a unique identifier.
- Process Orders
- For each customer in the list:
- Display a welcome message with the customer's name.
- Start a new order by clearing the current order list.
- Allow the customer to add items to their order:
- Prompt the customer to enter the item number (0 to finish the order).
- If the item number is valid (within the range of available items):
- Prompt the customer to enter the quantity of the selected item.
- Add the selected item and quantity to the current order.
- If the item number is 0, finish the order.
- If the item number is invalid, display an error message and ask for input again.
- Calculate the total bill for the current order by summing up the prices of items multiplied by their quantities.
- Calculate the charity contribution, which is 12% of the total bill.
- Calculate the amount to pay by subtracting the charity contribution from the total bill.
- Display the customer's order details:
- Customer's name
- Items ordered (item names, quantities, and individual prices)
- Total amount
- Amount for charity
- Amount to pay.

Repeat Orders (Optional)

After processing orders for all customers, ask if the system should continue with new orders.

If yes, return to step 3 to process new orders; otherwise, end the program.

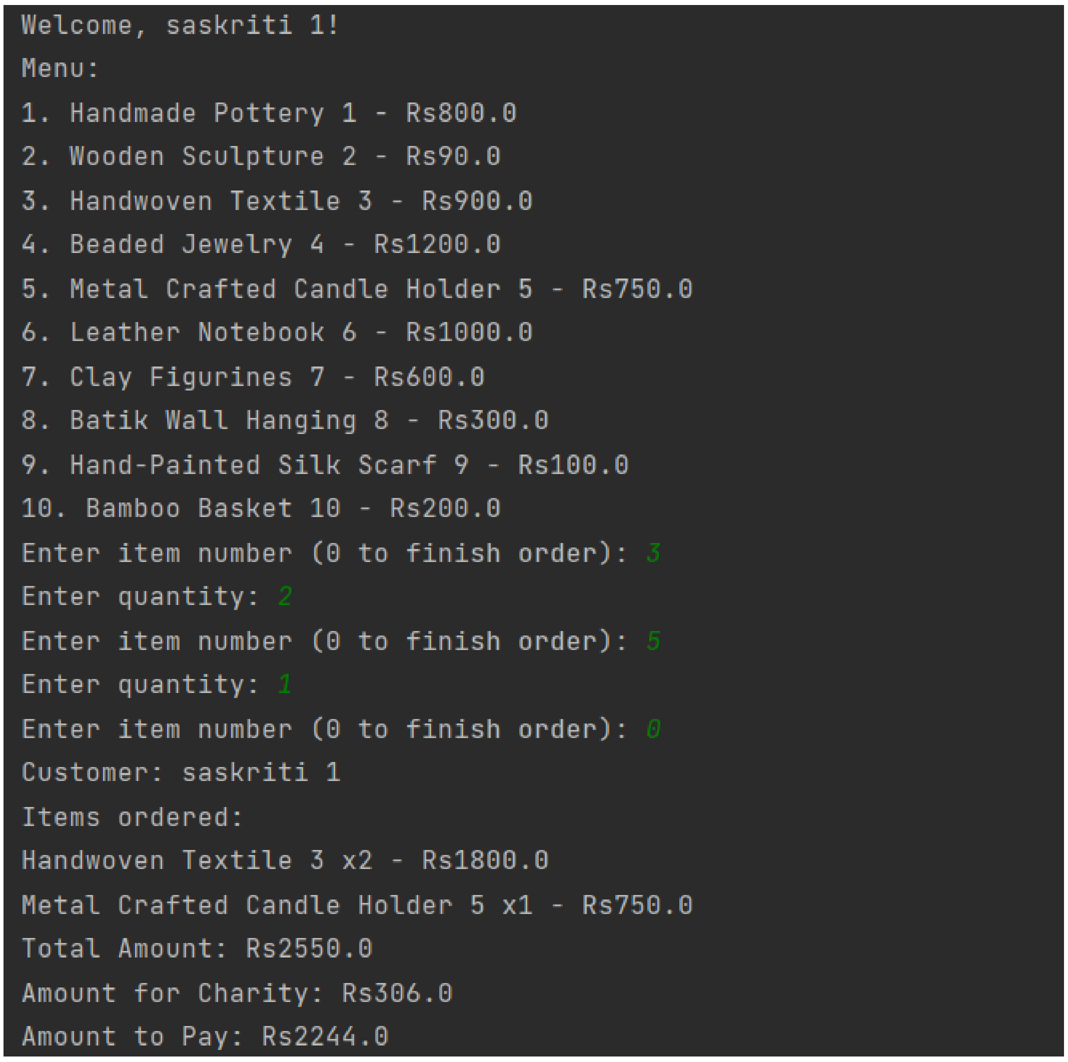
End Program

Exit the billing system

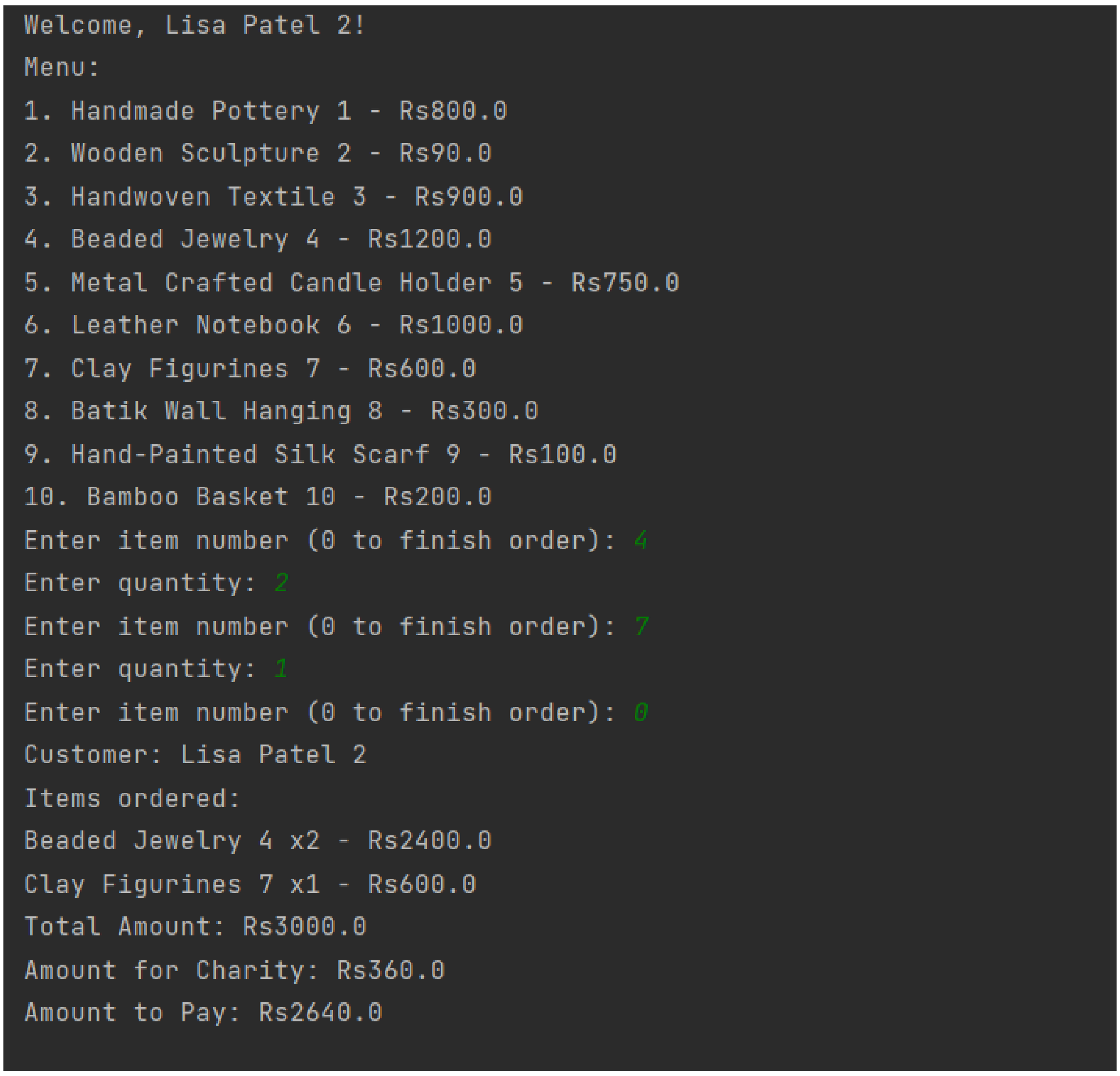
Test plan and test data to be used with testing

Customer no	Customer name	Item No	No of quantity	Price of item	Total amount	
1	Saskriti	3 5	2 1	Handmade Pottery=Rs800 Metal Crafted Candle holder=rs750	3*2= Rs 1800 5*5=Rs 750 1800+750=Rs2244	
2	Lisa Patel	4 7	2 1	Beaded jewelry =Rs1200 Clay Figuriness =Rs600	4*2=Rs2400 7*1=Rs600 2400+600=Rs3000	
3	Anshu	10 6	1 2	Bamboo Basket=Rs200 Leather Note book=Rs1000	10*1=Rs200 6*2=Rs2000 200+2000=2200	
4	Anuja	4 8	2 3	Beaded Jewelry=Rs1200 Batik Wall hanging= Rs300	4*2=Rs2400 8*3=Rs900 2400+900=2904	
5	Jenni	4 6	1 2	Beaded Jewelry=Rs1200 Leather notebook=Rs1000	4*1=Rs1200 6*2=Rs2000 1200+2000=3200	

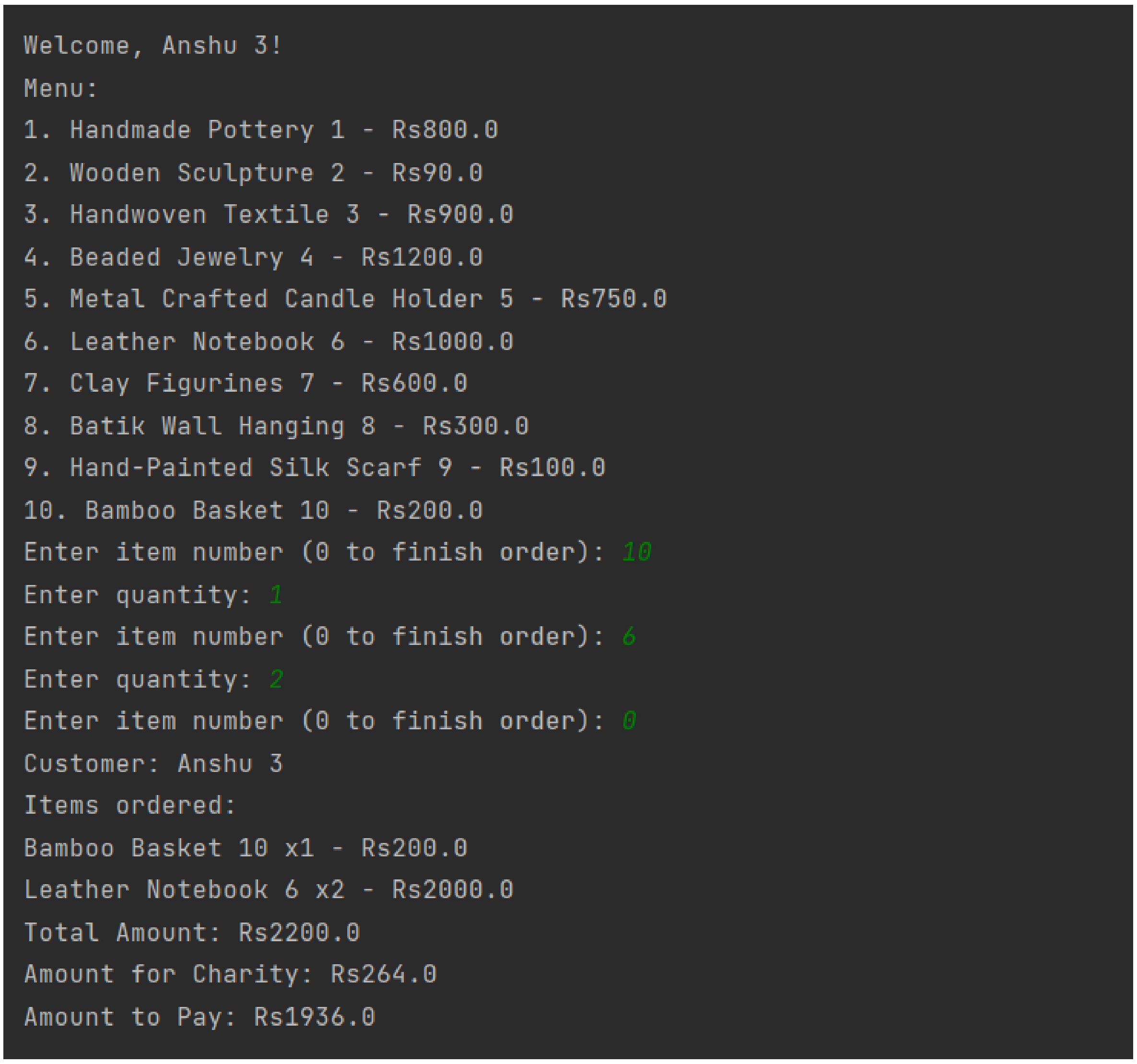
Screen output



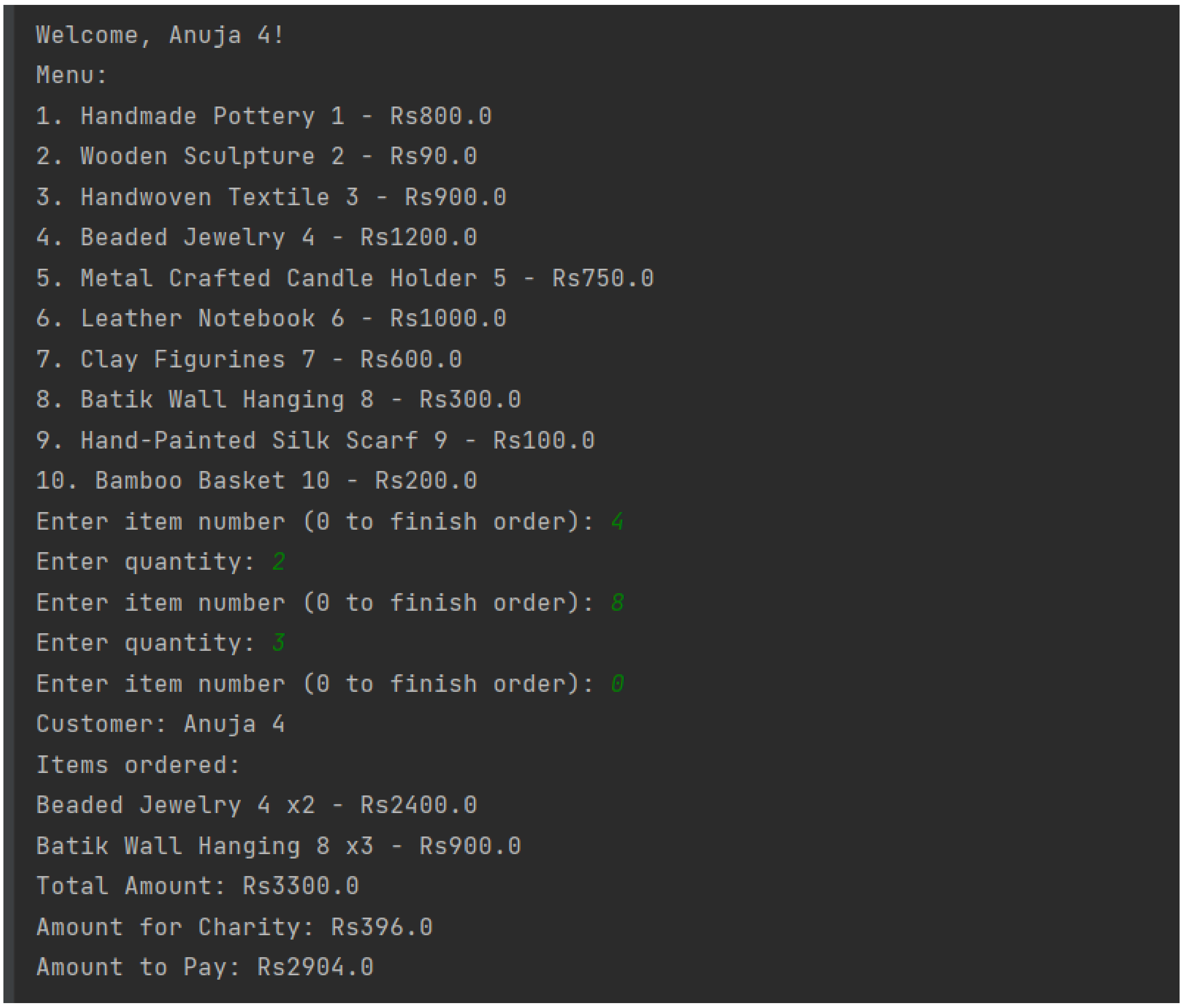
```
Welcome, saskriti 1!
Menu:
1. Handmade Pottery 1 - Rs800.0
2. Wooden Sculpture 2 - Rs90.0
3. Handwoven Textile 3 - Rs900.0
4. Beaded Jewelry 4 - Rs1200.0
5. Metal Crafted Candle Holder 5 - Rs750.0
6. Leather Notebook 6 - Rs1000.0
7. Clay Figurines 7 - Rs600.0
8. Batik Wall Hanging 8 - Rs300.0
9. Hand-Painted Silk Scarf 9 - Rs100.0
10. Bamboo Basket 10 - Rs200.0
Enter item number (0 to finish order): 3
Enter quantity: 2
Enter item number (0 to finish order): 5
Enter quantity: 1
Enter item number (0 to finish order): 0
Customer: saskriti 1
Items ordered:
Handwoven Textile 3 x2 - Rs1800.0
Metal Crafted Candle Holder 5 x1 - Rs750.0
Total Amount: Rs2550.0
Amount for Charity: Rs306.0
Amount to Pay: Rs2244.0
```



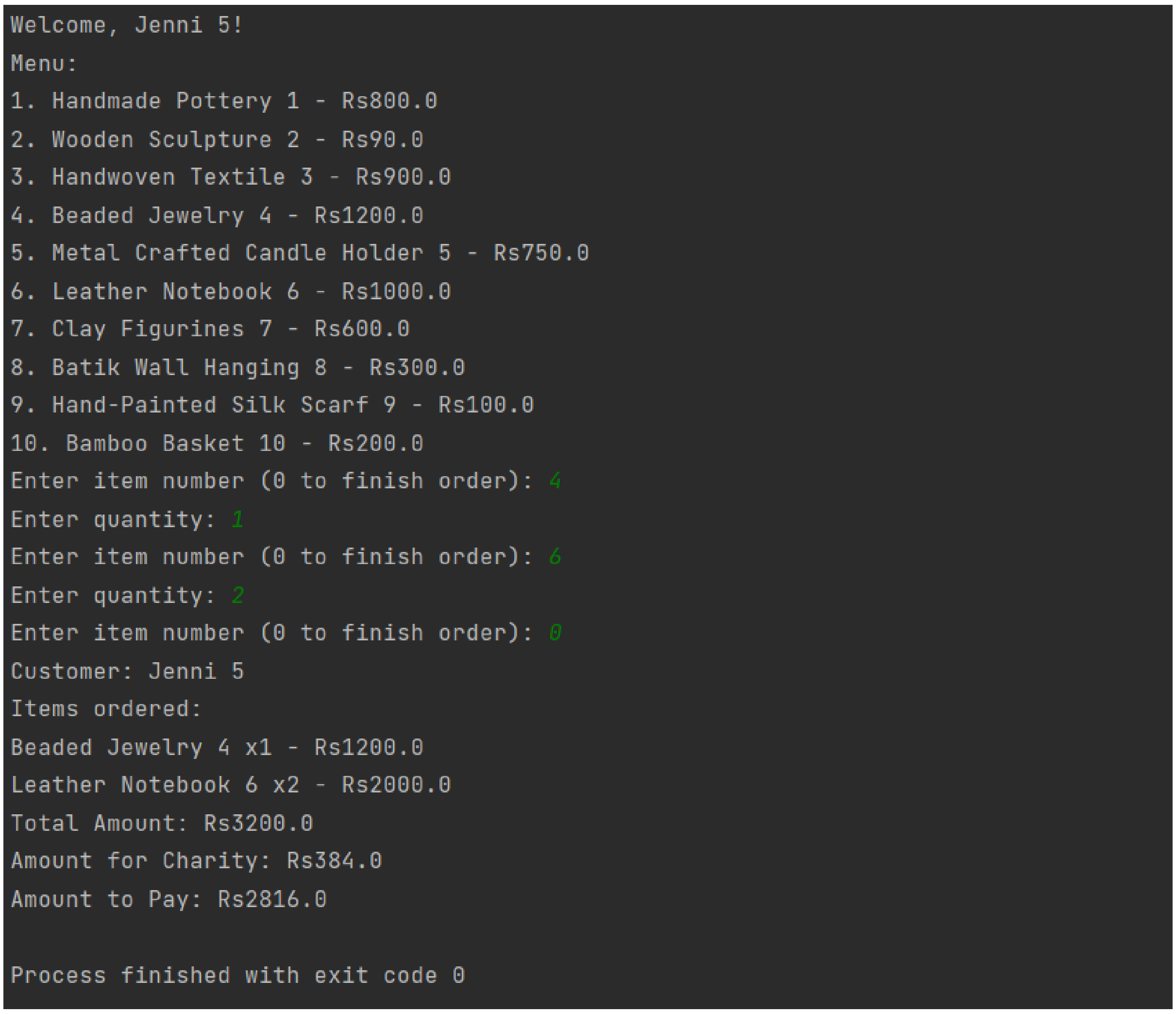
```
Welcome, Lisa Patel 2!
Menu:
1. Handmade Pottery 1 - Rs800.0
2. Wooden Sculpture 2 - Rs90.0
3. Handwoven Textile 3 - Rs900.0
4. Beaded Jewelry 4 - Rs1200.0
5. Metal Crafted Candle Holder 5 - Rs750.0
6. Leather Notebook 6 - Rs1000.0
7. Clay Figurines 7 - Rs600.0
8. Batik Wall Hanging 8 - Rs300.0
9. Hand-Painted Silk Scarf 9 - Rs100.0
10. Bamboo Basket 10 - Rs200.0
Enter item number (0 to finish order): 4
Enter quantity: 2
Enter item number (0 to finish order): 7
Enter quantity: 1
Enter item number (0 to finish order): 0
Customer: Lisa Patel 2
Items ordered:
Beaded Jewelry 4 x2 - Rs2400.0
Clay Figurines 7 x1 - Rs600.0
Total Amount: Rs3000.0
Amount for Charity: Rs360.0
Amount to Pay: Rs2640.0
```



```
Welcome, Anshu 3!
Menu:
1. Handmade Pottery 1 - Rs800.0
2. Wooden Sculpture 2 - Rs90.0
3. Handwoven Textile 3 - Rs900.0
4. Beaded Jewelry 4 - Rs1200.0
5. Metal Crafted Candle Holder 5 - Rs750.0
6. Leather Notebook 6 - Rs1000.0
7. Clay Figurines 7 - Rs600.0
8. Batik Wall Hanging 8 - Rs300.0
9. Hand-Painted Silk Scarf 9 - Rs100.0
10. Bamboo Basket 10 - Rs200.0
Enter item number (0 to finish order): 10
Enter quantity: 1
Enter item number (0 to finish order): 6
Enter quantity: 2
Enter item number (0 to finish order): 0
Customer: Anshu 3
Items ordered:
Bamboo Basket 10 x1 - Rs200.0
Leather Notebook 6 x2 - Rs2000.0
Total Amount: Rs2200.0
Amount for Charity: Rs264.0
Amount to Pay: Rs1936.0
```



```
Welcome, Anuja 4!
Menu:
1. Handmade Pottery 1 - Rs800.0
2. Wooden Sculpture 2 - Rs90.0
3. Handwoven Textile 3 - Rs900.0
4. Beaded Jewelry 4 - Rs1200.0
5. Metal Crafted Candle Holder 5 - Rs750.0
6. Leather Notebook 6 - Rs1000.0
7. Clay Figurines 7 - Rs600.0
8. Batik Wall Hanging 8 - Rs300.0
9. Hand-Painted Silk Scarf 9 - Rs100.0
10. Bamboo Basket 10 - Rs200.0
Enter item number (0 to finish order): 4
Enter quantity: 2
Enter item number (0 to finish order): 8
Enter quantity: 3
Enter item number (0 to finish order): 0
Customer: Anuja 4
Items ordered:
Beaded Jewelry 4 x2 - Rs2400.0
Batik Wall Hanging 8 x3 - Rs900.0
Total Amount: Rs3300.0
Amount for Charity: Rs396.0
Amount to Pay: Rs2904.0
```



```
Welcome, Jenni 5!
Menu:
1. Handmade Pottery 1 - Rs800.0
2. Wooden Sculpture 2 - Rs90.0
3. Handwoven Textile 3 - Rs900.0
4. Beaded Jewelry 4 - Rs1200.0
5. Metal Crafted Candle Holder 5 - Rs750.0
6. Leather Notebook 6 - Rs1000.0
7. Clay Figurines 7 - Rs600.0
8. Batik Wall Hanging 8 - Rs300.0
9. Hand-Painted Silk Scarf 9 - Rs100.0
10. Bamboo Basket 10 - Rs200.0
Enter item number (0 to finish order): 4
Enter quantity: 1
Enter item number (0 to finish order): 6
Enter quantity: 2
Enter item number (0 to finish order): 0
Customer: Jenni 5
Items ordered:
Beaded Jewelry 4 x1 - Rs1200.0
Leather Notebook 6 x2 - Rs2000.0
Total Amount: Rs3200.0
Amount for Charity: Rs384.0
Amount to Pay: Rs2816.0

Process finished with exit code 0
```

Complete listing of java program

```
import java.util.ArrayList;

import java.util.Scanner;


record Item(String name, double price) {

}


record Customer(String name) {

}


record Order(Item item, int quantity) {


    public double calculateTotal() {

        return item.price() * quantity;

    }

}


class BillingSystem {

    private final ArrayList<Customer> customers;

    private final ArrayList<Order> currentOrder;


    public BillingSystem() {

        customers = new ArrayList<>();

        currentOrder = new ArrayList<>();

    }


    public void displayMenu(ArrayList<Item> items) {

        System.out.println("Menu:");
        for (int i = 0; i < items.size(); i++) {
            Item item = items.get(i);
            System.out.println((i + 1) + ". " + item.name() + " - Rs" + item.price());
        }
    }

    public void startOrder() {
        currentOrder.clear();
    }

    public void addToOrder(Item item, int quantity) {
        currentOrder.add(new Order(item, quantity));
    }

    public double calculateBill() {
        double totalAmount = 0;
        for (Order order : currentOrder) {
            totalAmount += order.calculateTotal();
        }
        return totalAmount;
    }

    public double calculateCharityAmount(double totalAmount) {
        return 0.12 * totalAmount;
    }

    public void printBill(Customer customer, double totalAmount, double charityAmount) {
        System.out.println("Customer: " + customer.name());

        System.out.println("Items ordered:");

        for (Order order : currentOrder) {

            System.out.println(order.item().name() + " x" + order.quantity() + " - Rs" +
order.calculateTotal());

        }

        System.out.println("Total Amount: Rs" + totalAmount);

        System.out.println("Amount for Charity: Rs" + charityAmount);

        System.out.println("Amount to Pay: Rs" + (totalAmount - charityAmount));

    }


    public void addCustomer(Customer customer) {

        customers.add(customer);

    }


    public void processOrders(ArrayList<Item> items) {

        Scanner scanner = new Scanner(System.in);

        for (Customer customer : customers) {

            System.out.println("\nWelcome, " + customer.name() + "!");

            displayMenu(items);

            startOrder();

            while (true) {

                System.out.print("Enter item number (0 to finish order): ");

                int itemNumber = scanner.nextInt();

                if (itemNumber == 0) {

                    break;

                }

                if (itemNumber >= 1 && itemNumber <= items.size()) {
```

```
                System.out.print("Enter quantity: ");
                int quantity = scanner.nextInt();
                Item selectedItem = items.get(itemNumber - 1);
                addToOrder(selectedItem, quantity);
            } else {
                System.out.println("Invalid item number. Please try again.");
            }
        }
        double totalAmount = calculateBill();
        double charityAmount = calculateCharityAmount(totalAmount);
        printBill(customer, totalAmount, charityAmount);
    }
  }
}

public class Main {
  public static void main(String[] args) {
    ArrayList<Item> items = new ArrayList<>();
    items.add(new Item("Handmade Pottery 1", 800.0));
    items.add(new Item("Wooden Sculpture 2", 90.0));
    items.add(new Item("Handwoven Textile 3", 900.0));
    items.add(new Item("Beaded Jewelry 4", 1200.0));
    items.add(new Item("Metal Crafted Candle Holder 5", 750.0));
    items.add(new Item("Leather Notebook 6", 1000.0));
    items.add(new Item("Clay Figurines 7", 600.0));
    items.add(new Item("Batik Wall Hanging 8", 300.0));
    items.add(new Item("Hand-Painted Silk Scarf 9", 100.0));
```

```
        items.add(new Item("Bamboo Basket 10", 200.0));



        BillingSystem billingSystem = new BillingSystem();
        billingSystem.addCustomer(new Customer("saskriti 1"));
        billingSystem.addCustomer(new Customer("Lisa Patel 2"));
        billingSystem.addCustomer(new Customer("Anshu 3"));
        billingSystem.addCustomer(new Customer("Anuja 4"));
        billingSystem.addCustomer(new Customer("Jenni 5"));



        billingSystem.processOrders(items);
    }
}
```

Conclusion

Based on the assignment specifications provided, the following conclusions can be drawn:

Objective of the Assignment: The primary objective of the assignment is to develop a simple billing system for a local handicraft store in Patan. The system is designed to facilitate customer orders, calculate bills, allocate a portion to charity, and manage multiple customers' orders simultaneously.

Object-Oriented Approach: The assignment emphasizes an object-oriented programming approach. It requires the creation of suitable classes and objects to model the different aspects of the billing system, including items, customers, orders, and the billing system itself. This approach promotes code modularity and maintainability.

Key Features and Functionalities:

- Display a menu of items for sale.
- Allow customers to place orders and specify quantities.
- Calculate the total bill for each order.
- Allocate 12% of the total bill to charity.
- Show the amounts for charity, store revenue, and the total payment for each customer.
- Support multiple customers, generating new bills for each customer.

MVP (Minimum Viable Product) Considerations: The assignment specifies the development of an MVP that can handle the order and billing processes for at least five customers simultaneously. This suggests a focus on core functionalities with room for potential future enhancements.

Data Representation: The assignment requires the representation of items, customers, and orders using classes and objects. This data structure allows for organized management of information and supports the implementation of the billing system's features.

Data Hardcoding for Testing: The assignment permits the initial hardcoding of menu items and their prices for testing purposes. This simplifies the development phase by allowing the use of predefined data before implementing more complex data management solutions.

User Interaction: The assignment assumes user interaction through a command-line interface. Customers can select items and quantities via text-based prompts.

Data Validation and Error Handling: The assignment does not explicitly mention data validation and error handling, but in practice, it would be essential to include these elements to ensure the reliability and robustness of the billing system.

Documentation: The assignment emphasizes the importance of documenting the computer program. This includes providing clear explanations of code functionality and usage.

Future Enhancements: While the assignment specifies an MVP, there is room for future enhancements such as improving data storage, adding user authentication, and creating a graphical user interface (GUI) for a more user-friendly experience.

In conclusion, the assignment aims to develop a basic billing system for a handicraft store, emphasizing an object-oriented approach and core functionalities. It provides an opportunity to apply programming skills to solve a real-world problem and lays the foundation for potential future improvements and feature expansions.

References

https://pariskathmandu.com/patan/

https://www.linkedin.com/pulse/health-benefits-handicrafts-ang%C3%A9lica-elisa-sonaglio

Marking Scheme

20% Design

•UML class diagram with essential attributes and operations.

•Pseudo code/algorithm to illustrate your solution logic

15% quality of Java program implementation attributes and operations

•Data structure and variable allocation.

•Clear and straightforward code to solve the problem

20% Test plan and cases

•Test cases and test data

•Output screenshots

40% Correctness

•Code produces correct result 10% Documentation

•Well-structured with content page, introduction design, test plan, program listing, screen shots and conclusion with reference.

THE END…………