A blue sign with white text

Description automatically generated

Name: Shubham Shah

STUDENT ID – 23085252

BSc Computing 4 years

Level – 3

Semester – 1ST

COURSE NAME – Introduction to Programming

PROJECT WORK

About code:

This C++ code implements a simple shopping cart system for a supermarket, allowing both administrators and customers to interact with the system. Here's a brief overview and analysis of the code:

**1. Class Hierarchy**:

- The code defines a hierarchy of classes related to products, including the base class `Product` and derived classes `Electronics`, `Clothing`, and `Groceries`.

- Each product class implements pure virtual functions for displaying product information and calculating the total cost.

**2. Customer and Invoice Classes**:

- There are classes for `Customer` and `Invoice`.

- The `Invoice` class contains a vector of pointers to `Product` objects, representing the products included in the invoice.

**3. ShoppingCart Class:**

- Manages available products, customers, and invoices.

- Provides methods for adding, editing, and removing products, as well as adding customers, creating invoices, and listing available products, customers, and invoices.

**4. Menu-Driven Interface:**

- The program uses a menu-driven interface for both administrators and buyers, allowing them to perform various actions like adding/editing products, adding customers, creating invoices, and listing information.

**5. Implementation Details**:

- The code uses vectors to store available products, customers, and invoices.

- Product information is added, edited, and displayed through user inputs.

- Invoices are created by associating products with a customer and calculating the total amount.

**6. Improvements and Considerations:**

- The code assumes a fixed tax for electronics and a fixed discount for clothing. These values could be made configurable or based on more complex rules.

- The use of pointers and dynamic memory allocation for products could potentially be replaced with smart pointers or other ownership models to improve memory management.

- Error handling and validation of user inputs are minimal; additional checks could be added to enhance robustness.

- The program doesn't persist data between runs; incorporating file I/O for data storage could be considered for a more practical application.

**7. Testing and Usage:**

- The program is launched from the `main()` function, and the `ShoppingCart` class handles the user interaction.

- The code seems to work with a loop, allowing users to perform multiple actions until they choose to exit.

In summary, the code provides a basic framework for a shopping cart system, but there's room for improvement in terms of flexibility, error handling, and memory management. Additionally, the code could be extended to include features like file I/O for data persistence and a more sophisticated pricing strategy for products.

**FLOWCHART:**

**A computer screen shot of a diagram

Description automatically generated**

**ALGORITHM:**

Sure, here's a high-level algorithm for the provided C++ code:

**1. Class Definitions:**

- Define the abstract class `Product` with pure virtual functions `display()` and `calculateCost()`.

- Create derived classes `Electronics`, `Clothing`, and `Groceries` with specific implementations of `display()` and `calculateCost()`.

**2. Customer Class:**

- Create a class `Customer` to store customer information, including a name and ID.

**3. Invoice Class:**

- Define an `Invoice` class with an invoice number, customer, and a vector of products.

- Include a method to add products to the invoice and a method to generate the invoice with total cost.

**4. Shopping Cart Class:**

- implement a `Shopping Cart` class to manage available products, customers, and invoices.

- Include methods for adding, editing, and removing products, listing products and customers, and creating invoices.

**5. Menu Function:**

- Create a `menu()` function in `Shopping Cart` to display the main menu.

- Inside the menu, include options for administrators, buyers, and exiting the program.

**6. Administrator Functions:**

- Implement an `administrator()` function in `Shopping Cart` to handle administrator actions.

- Include options to add, edit, remove, and list products, add customers, create invoices, and list invoices.

**7. Buyer Function**:

- Implement a `buyer()` function in `Shopping Cart` to handle buyer actions.

- Prompt the user for their customer ID and allow them to add products to their invoice.

**8. Product Management Functions:**

- Implement functions in `Shopping Cart` to add, edit, remove, and list available products.

**9. Customer Management Functions:**

- Implement functions in `Shopping Cart` to add and list customers.

**10. Invoice Management Functions:**

- Implement functions in `Shopping Cart` to create and list invoices.

**11. Main Function:**

- Create an instance of `Shopping Cart` in the `main()` function.

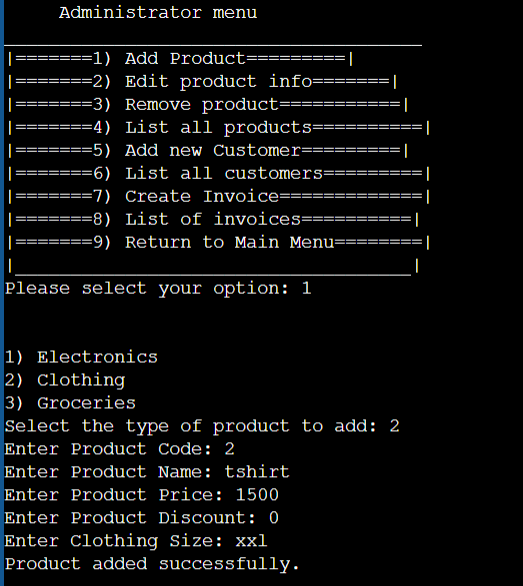
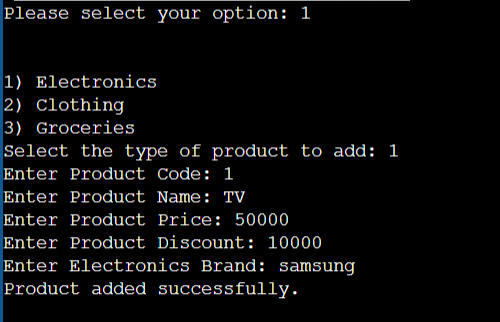
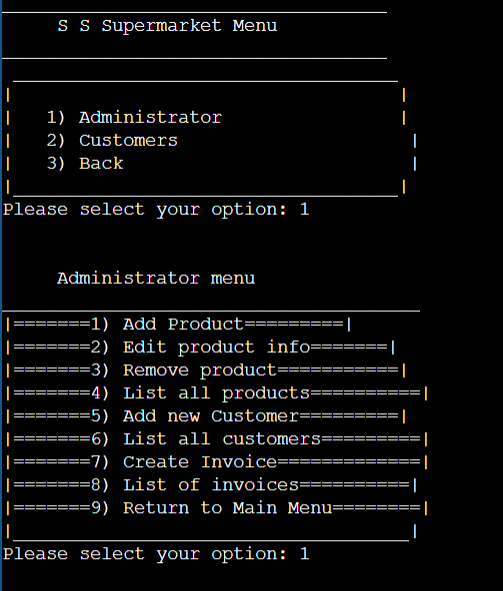
- Call the `menu()` function to start the program.

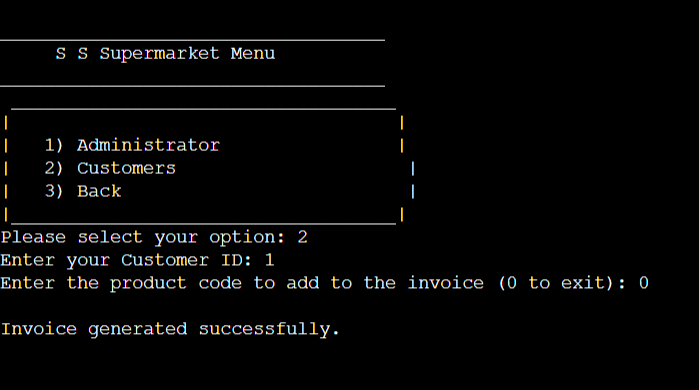
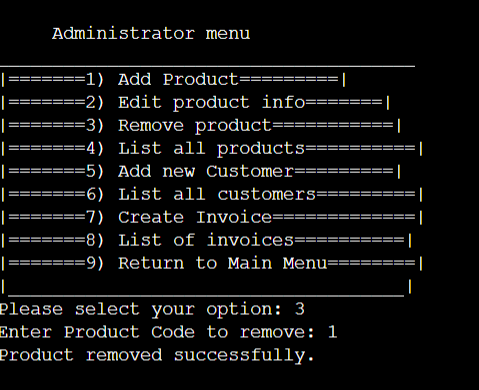
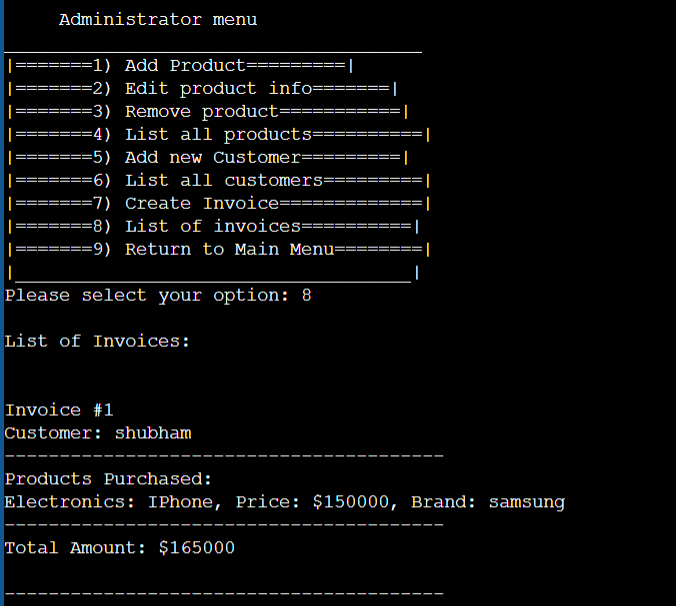
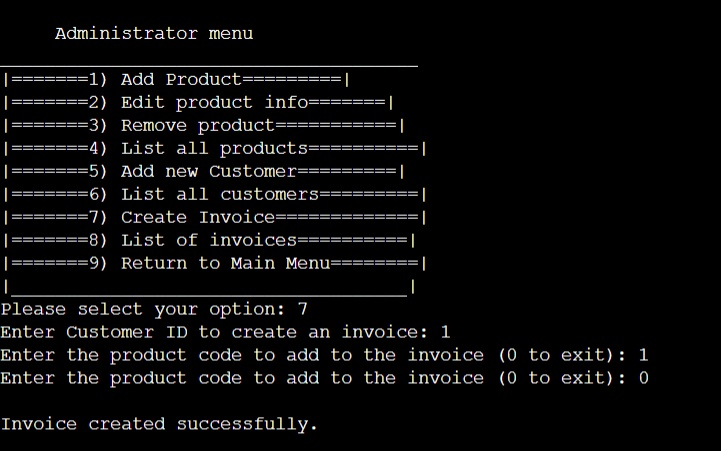
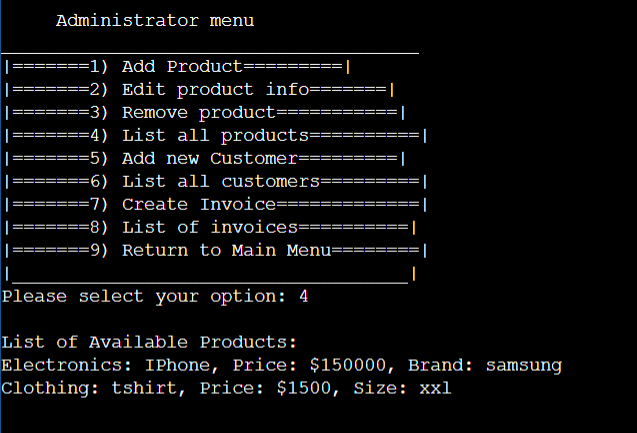
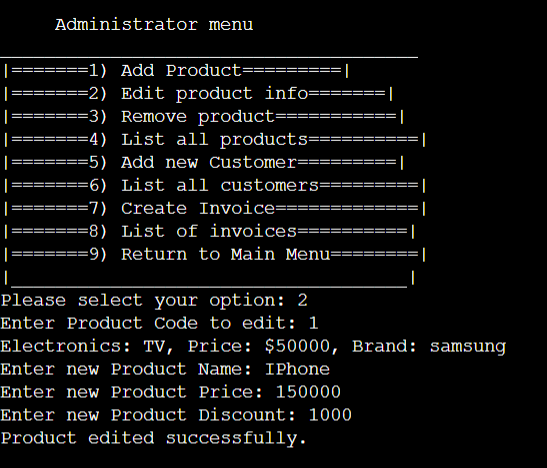
This algorithm provides an overview of the structure and functionality of the C++ code. Each step corresponds to a specific feature or action within the program.

SOURCE CODE:

<https://drive.google.com/file/d/1bT4afJC-ZkEG5odh_grG4ERASabkblZV/view?usp=drive_link>

**OUTPUT:**

A screenshot of a menu

Description automatically generated

THANK YOU