Bakery Management System in C

PROJECT IN C LANGUAGE USING SOME ALGORITHMS

Project Submitted by:Raj Sharma
UID:24BCA10553

SECTION:BCA{8A}

SUBJECT:Data Structure And Algorithms Using C Language

Acknowledgement

I would like to express my sincere gratitude to my faculty and mentors for their constant guidance and support throughout the development of this project. Their valuable insights, suggestions, and encouragement have played a crucial role in completing this project successfully. I would also like to extend my thanks to my friends and peers for their help and motivation during the entire course of the project. Without their constant assistance, this project would not have been completed in time. I am also thankful for all the resources provided by my college and the available references, which helped me understand the concepts more effectively.

Abstract

The Bakery Management System is a simple yet powerful C-based console application designed to assist bakeries in handling customer orders efficiently. It allows users to select items from a list, input the quantity, and automatically calculates the final bill, including item prices, total cost, and applicable discounts. This system eliminates the need for manual calculations, saving time and reducing errors, and allows bakery managers to focus on the quality of their products and services. The system also gives a user-friendly interface that can be operated with minimal technical knowledge. The main objective of this project is to demonstrate the application of fundamental C programming concepts such as loops, conditionals, structures, and functions in a real-world scenario. This project aims to streamline the order processing system and improve customer service by providing an error-free, efficient, and reliable automated solution.

Introduction

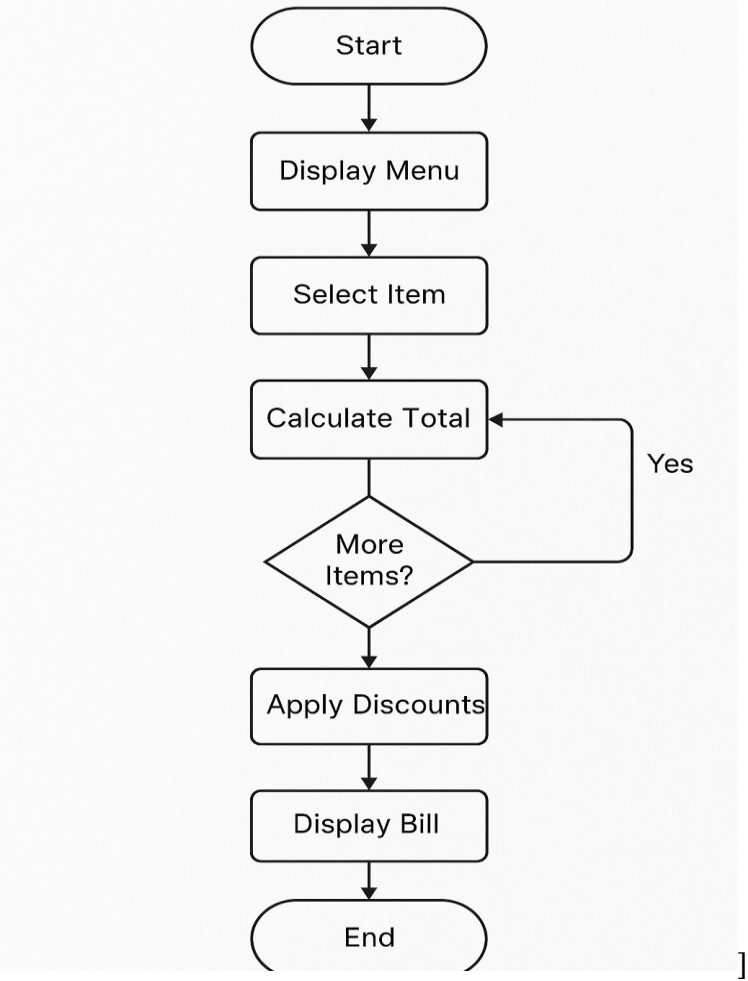
In today's fast-paced world, managing customer orders in a bakery can be a time-consuming and error-prone process. As the customer base grows, manual tracking and billing become increasingly difficult. The Bakery Management System, designed using the C programming language, aims to simplify and automate the process of order management. This system is designed to handle multiple types of bakery items, allowing customers to select items, specify quantities, and generate a final bill. This project involves core programming concepts such as input/output operations, control structures, arrays, and user-defined functions. The system also handles basic discounts and provides a summary of the transaction, making it a highly efficient and cost-effective solution for small bakeries. With the growing need for automation in businesses, this system can be further enhanced by adding features such as inventory management, customer order history, and real-time updates.

Problem Statement

Manual handling of bakery orders is time-consuming and prone to human error, especially when the number of customers and the variety of products increases. Bakery staff often struggle to maintain an accurate record of orders, calculate total costs, and provide a bill in a timely manner. Furthermore, handling discounts and applying them manually can result in inconsistencies. As a result, there is a pressing need for an automated solution that can accurately track orders, calculate totals, and generate bills without errors, improving both the efficiency and customer satisfaction in bakery services. The main problem addressed by this project is the lack of automation in the order processing and billing system. This project eliminates the chances of human error, speeds up the ordering process, and ensures customers receive a clear and accurate bill.

Flow Chart

The flowchart below illustrates the working of the Bakery Management System: This flowchart begins with the user selecting an item from the menu, entering the quantity, and then the system calculating the total. Once the user confirms the items, the system prints the final bill. The process allows the user to keep adding multiple items until the order is completed.

[Flowchart Placeholder - 

]

Source Code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_ITEMS 100

struct Item {
    char name[50];
    float price;
    int quantity;
};

void showMenu() {
    printf("\n--- Welcome to Sweet Treats Bakery ---\n");
    printf("Code\tItem\t\tPrice\n");
    printf("1\tCake Slice\tRs. 50\n");
    printf("2\tPastry\t\tRs. 30\n");
    printf("3\tBread\t\tRs. 25\n");
    printf("4\tCookies\t\tRs. 20\n");
    printf("5\tMuffin\t\tRs. 40\n");
    printf("6\tExit Order\n");
}

void generateBill(struct Item orderList[], int count) {
    printf("\n======= Final Bill =======\n");
    printf("Item\t\tQty\tPrice\tTotal\n");
```

```
    float total = 0;
    int totalQuantity = 0;

    for (int i = 0; i < count; i++) {
        float itemTotal = orderList[i].price * orderList[i].quantity;
        total += itemTotal;
        totalQuantity += orderList[i].quantity;

        printf("%-12s\t%d\tRs. %.2f\tRs. %.2f\n",
               orderList[i].name, orderList[i].quantity,
orderList[i].price, itemTotal);
    }

    float discount = 0;
    if (total >= 200) {
        discount = total * 0.1;
    }

    printf("\nTotal Unique Items: %d", count);
    printf("\nTotal Quantity Ordered: %d", totalQuantity);
    printf("\nSubtotal: Rs. %.2f", total);
    printf("\nDiscount: Rs. %.2f", discount);
    printf("\nGrand Total: Rs. %.2f", total - discount);
    printf("\n\nThank you for visiting Sweet Treats Bakery!\n");
}

int main() {
    int choice, quantity, count = 0;
    struct Item orderList[MAX_ITEMS];
```

```
    do {
        showMenu();
        printf("\nEnter item code (1-6): ");
        scanf("%d", &choice);

        if (choice >= 1 && choice <= 5) {
            printf("Enter quantity: ");
            scanf("%d", &quantity);

            switch(choice) {
                case 1: strcpy(orderList[count].name, "Cake Slice");
orderList[count].price = 50; break;
                case 2: strcpy(orderList[count].name, "Pastry");
orderList[count].price = 30; break;
                case 3: strcpy(orderList[count].name, "Bread");
orderList[count].price = 25; break;
                case 4: strcpy(orderList[count].name, "Cookies");
orderList[count].price = 20; break;
                case 5: strcpy(orderList[count].name, "Muffin");
orderList[count].price = 40; break;
            }

            orderList[count].quantity = quantity;
            count++;
        } else if (choice != 6) {
            printf("Invalid item code! Please try again.\n");
        }

    } while (choice != 6);
```

```
    if (count > 0) {
        generateBill(orderList, count);
    } else {
        printf("\nNo items ordered. Thank you!\n");
    }

    return 0;
}
```

Output

The Bakery Management System generates the following output after a customer completes their order. The output displays all the ordered items, their quantity, price, and the total amount. A sample output is as follows:

```
--- Welcome to Sweet Treats Bakery ---
Code    Item        Price
1   Cake Slice  Rs. 50
2   Pastry      Rs. 30
3   Bread       Rs. 25
4   Cookies     Rs. 20
5   Muffin      Rs. 40
6   Exit Order

Enter item code (1-6): 1
Enter quantity: 2

--- Welcome to Sweet Treats Bakery ---
Code    Item        Price
1   Cake Slice  Rs. 50
2   Pastry      Rs. 30
3   Bread       Rs. 25
4   Cookies     Rs. 20
5   Muffin      Rs. 40
6   Exit Order

Enter item code (1-6): 2
Enter quantity: 3
```

```
======= Final Bill =======
Item        Qty Price   Total
Pastry          2   Rs. 30.00   Rs. 60.00
Pastry          3   Rs. 30.00   Rs. 90.00

Total Unique Items: 2
Total Quantity Ordered: 5
Subtotal: Rs. 150.00
Discount: Rs. 0.00
Grand Total: Rs. 150.00

Thank you for visiting Sweet Treats Bakery!
```

Future Enhancement

The Bakery Management System has several possibilities for future enhancements, including:

1. **Graphical User Interface (GUI):** The system can be developed further by adding a GUI, which would make it more user-friendly and accessible to bakery staff who are not comfortable with command-line interfaces.

2. **Inventory Management System:** Implementing an inventory management system to automatically track ingredient stock and alert the user when supplies are running low.

3. **Customer Database and Order History:** The system can be enhanced to maintain a database of regular customers and their order histories. This would allow for personalized recommendations and a better customer experience.

4. **Real-Time Updates:** Implementing real-time updates to track orders, customer status, and bakery performance could increase productivity.

5. **Online Ordering System:** Adding a feature for online ordering could extend the bakery's reach and customer base, allowing users to place orders remotely and have them ready for pickup or delivery.

Conclusion

This Bakery Management System project in C demonstrates the practical application of key programming concepts such as loops, conditionals, structures, and functions. By automating the process of handling bakery orders, the system helps reduce errors, improve efficiency, and save valuable time for bakery staff. The solution is scalable and can be adapted for various bakery sizes. As technology continues to play a critical role in business operations, projects like these serve as an introduction to real-world problem-solving with programming. In the future, this system can be expanded further to include advanced features such as a GUI, online ordering, and inventory management.

References

1. **Let Us C** by Yashavant Kanetkar - Provides a strong foundation in C programming.
2. **GeeksforGeeks C Programming** - A valuable online resource for understanding basic C concepts.
3. **TutorialsPoint C Language** - A helpful tutorial for beginners.
4. **Stack Overflow** - For solutions to common programming challenges and syntax-related issues.
5. **Online C Programming Resources** - Various websites offering tutorials and example code for C projects.