**Restaurant Management System**

# Abstract:

*This report presents the design and implementation of a menu management system for a restaurant, aimed at providing an intuitive user interface for both management and customer interaction. The system is structured into two main modules: Management and Customer, each offering different functionalities to enhance the restaurant's operations. The Management module allows authorized users to view, add, update, and remove menu items, as well as manage customer feedback. The Customer module provides users with the ability to place orders, review the menu, and give feedback on their experience.*

*Key features of the system include a tax-inclusive payment system, customer feedback management, and options for order customization. The system integrates payment options, including card and cash payments, with built-in discounts based on order value and card type. Additionally, the software uses a file-based storage mechanism for saving and displaying menu items, customer orders, and feedback.*

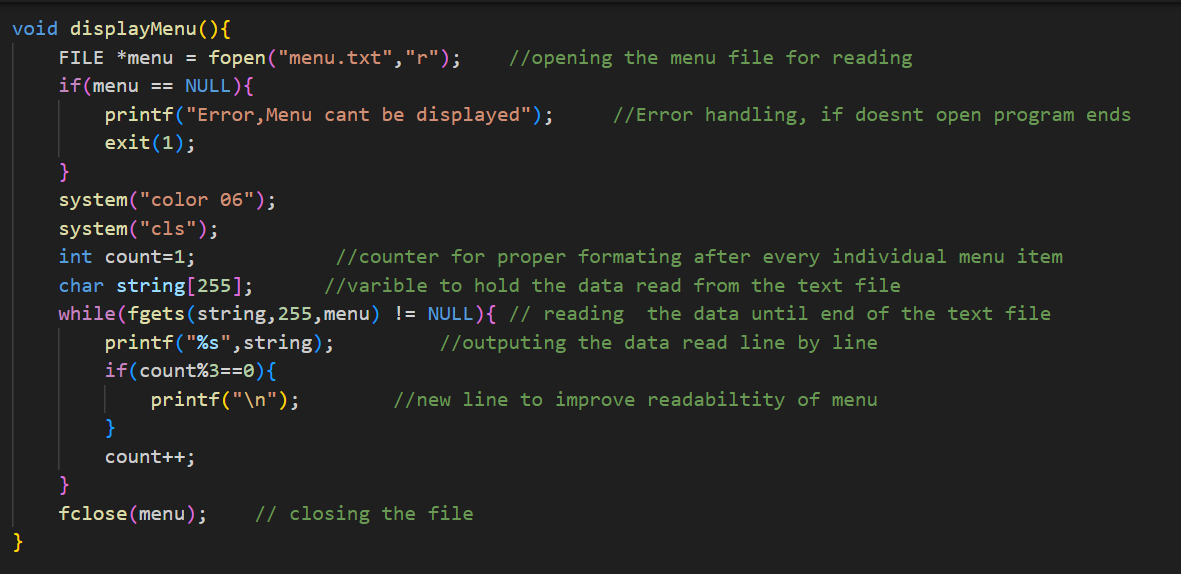
*The system employs robust input validation, error handling, and dynamic menu adjustments to accommodate changing menu items. A well-structured design and modular approach ensure scalability, with support for up to 50 menu items and efficient order processing.*

*This report details the functionalities implemented, the system design, and challenges encountered during the development, along with suggestions for future improvements. The system is intended to streamline restaurant operations, improve customer satisfaction, and provide real-time feedback for better service quality.*

## Main Functionality of Code:

A screenshot of a computer program

Description automatically generatedCreate Menu Function: It is called at the start of the menu function to create the menu text file and to write the initial menu.

Display menu function: This function is used for displaying the data written in the menu text file, it also clears the screen and changes the color of the text on the screen to yellow.

A computer screen shot of text

Description automatically generated

Save menu function: This writes the data menu item structure array to menu file

A computer screen shot of text

Description automatically generated

A computer code on a black background

Description automatically generated

Add item function: This asks for no of items to be updated, then runs a loop that many times. At each iteration, it asks name and price of the item to be added and adds it to at the end of the menu item’s structure array. No of items are limited to 50. There are checks in place which will not let the user enter more than 50 items in the menu.

A screenshot of a computer program

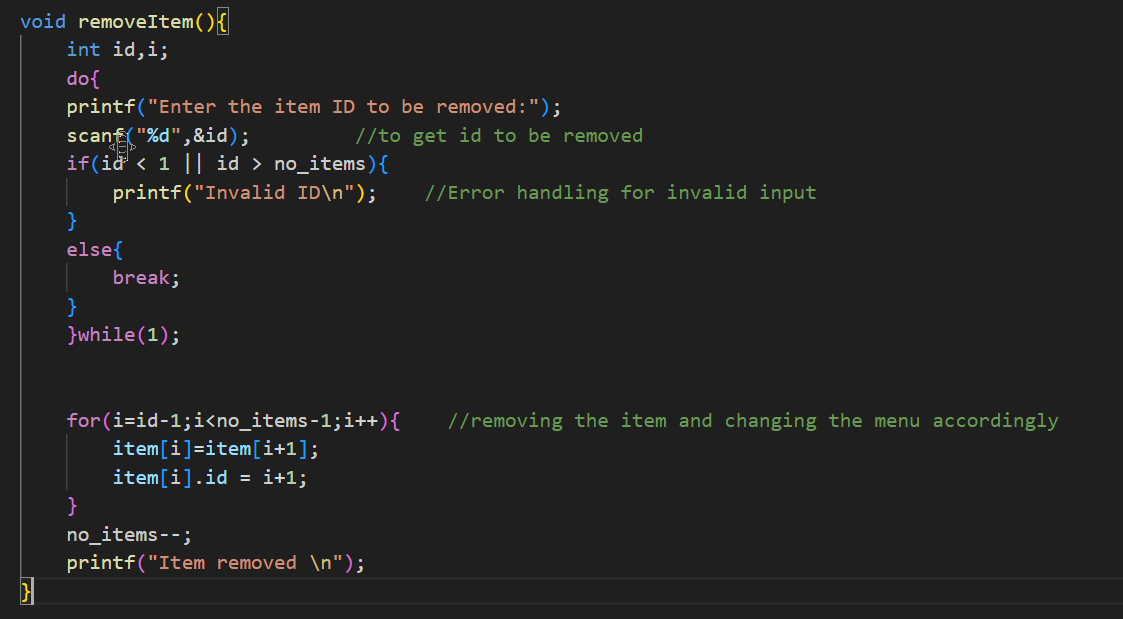
Description automatically generated

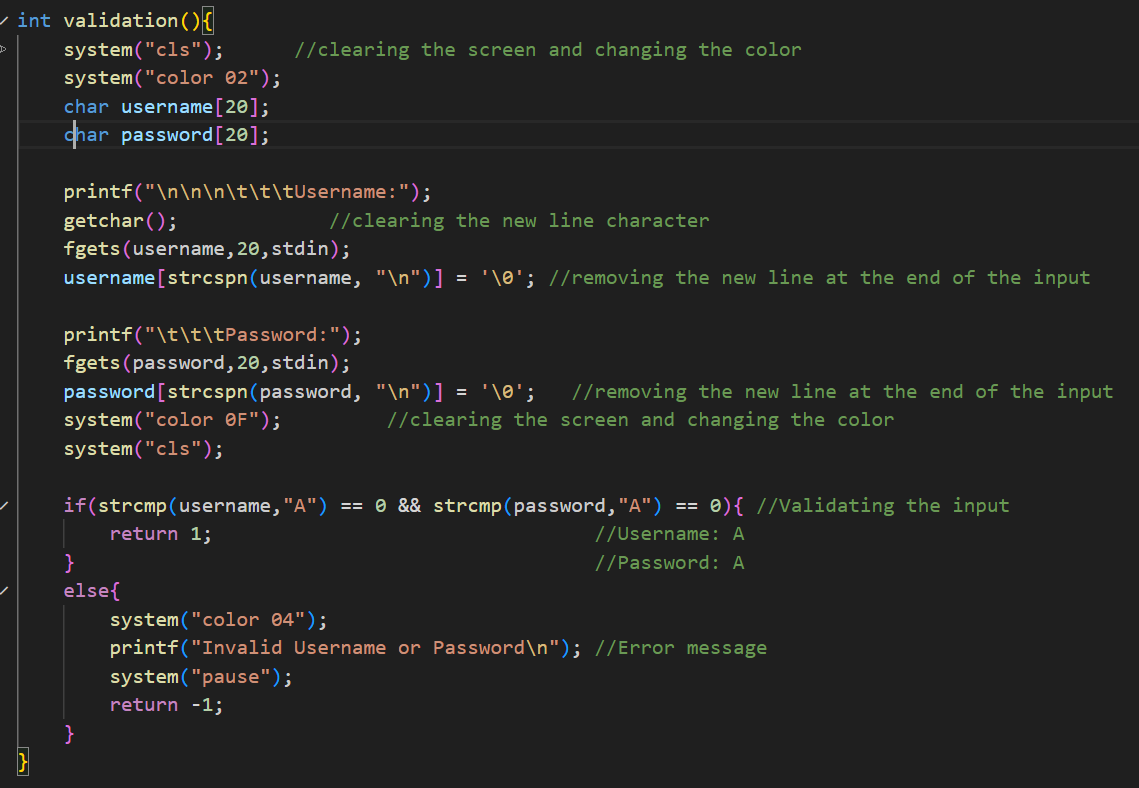
A screen shot of a computer program

Description automatically generated

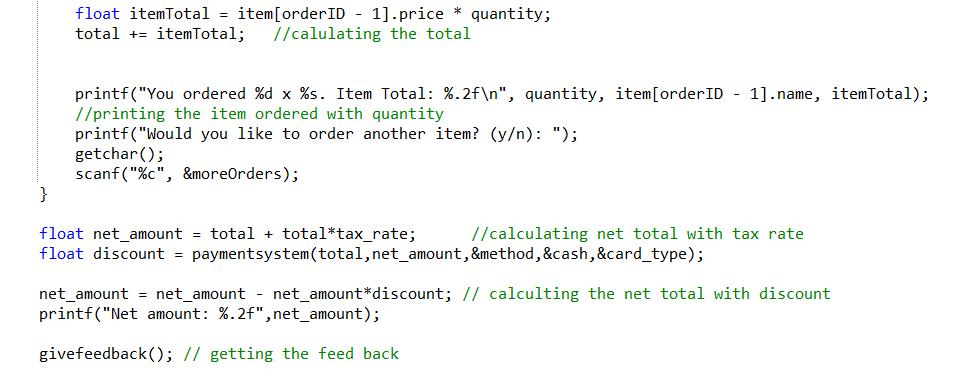
Update item function: This function asks for the id of the item to be updated, it also checks if it is valid or not. Then it presents a menu to ask what does the user wants to update. The menu has a option to exit it.

Remove item function: it asks for the id of the item to be removed and validates it. Then it deletes the items and adjusts the rest of the menu accordingly.

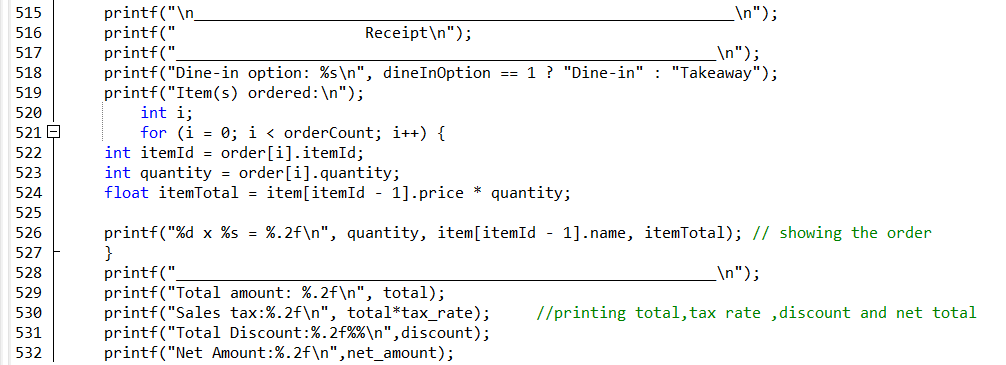




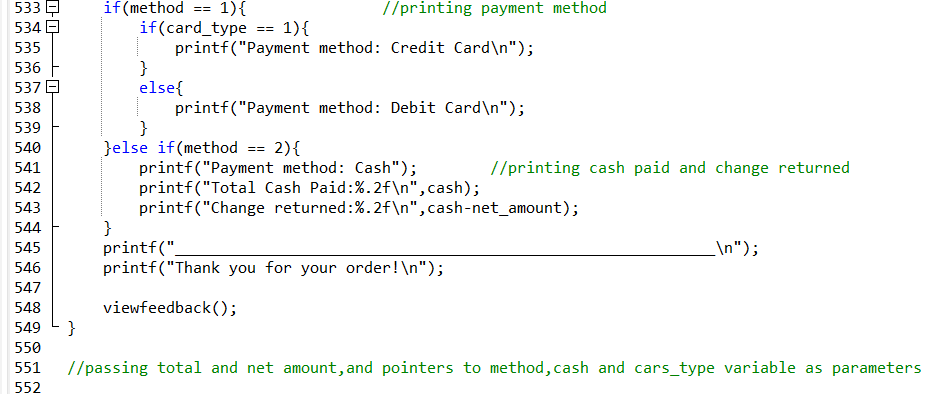
Validation Function: This function is called when the user accesses the management menu, it prompts the user to enter the username and password, if it matches the username and password set in the code then the access is granted else the access is denied. The text turns green when the function is called and turns red when invalid username or Password it entered.

**

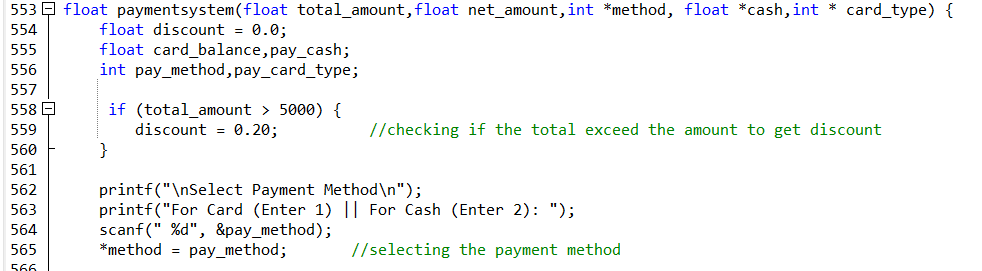
* *Have used* ***Float*** *(as the price of item can be in decimal so the* ***itemTotal*** *same for the* ***net\_amount*** *and* ***discount*** *as well).*
* *Used* ***Scanf*** *to ask from the user.*
* *Using* ***Printf*** *for printing the Net amount.*
* *Used* ***getchar*** *for buffer handling.*
* ***Givefeedback()*** *is already explained. Used it to in a function to connect within* ***takeorder*** *function  
  attach Ss of this function and explain.*



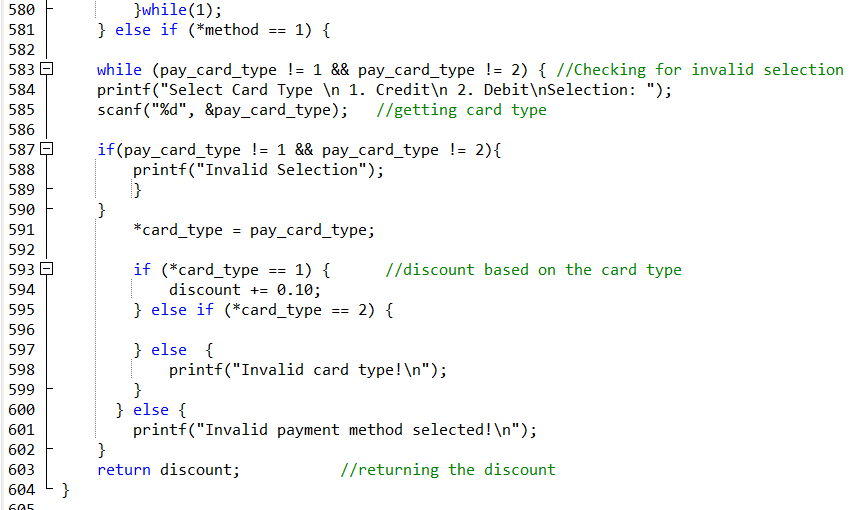
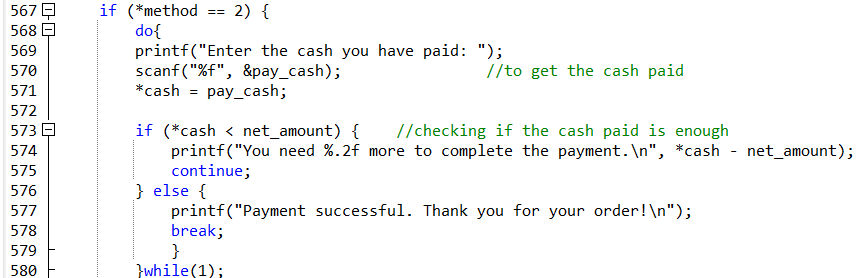
* *Printing the receipt of the order*



* *Using* ***If-else-if*** *for payment method.*
* *Used* ***nested if-else*** *in method 1 for further asking for card type. Credit or Debit.*

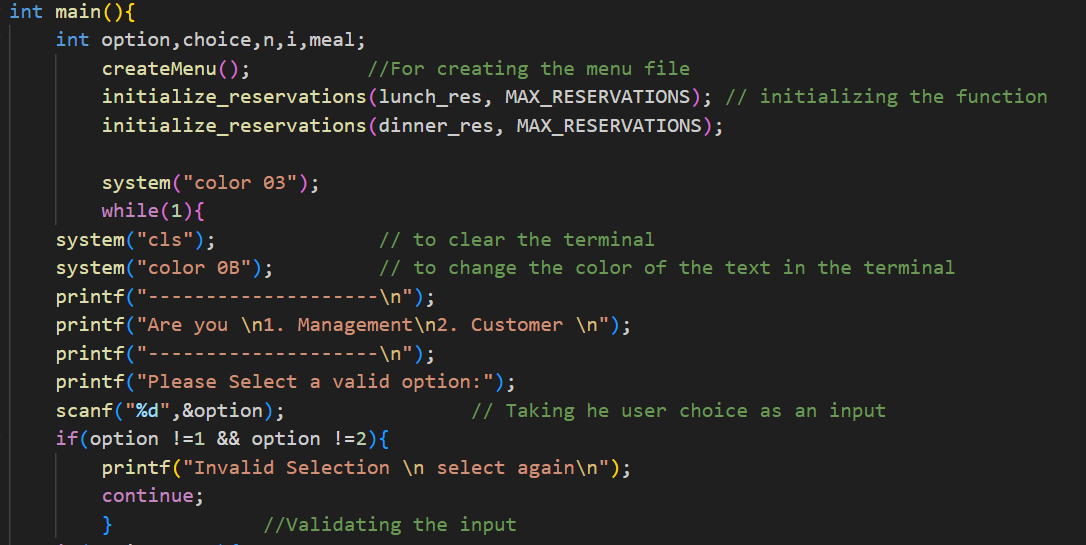
**

* *Using* ***pointers*** *to connect the* ***\*method,\*cash,\*card\_type*** *from a different function. In my case I am connection* ***method,cash,card\_type*** *from function* ***takeorder*** *to function* ***paymentsystem*** *so the total\_amount can be connected with the other function.*

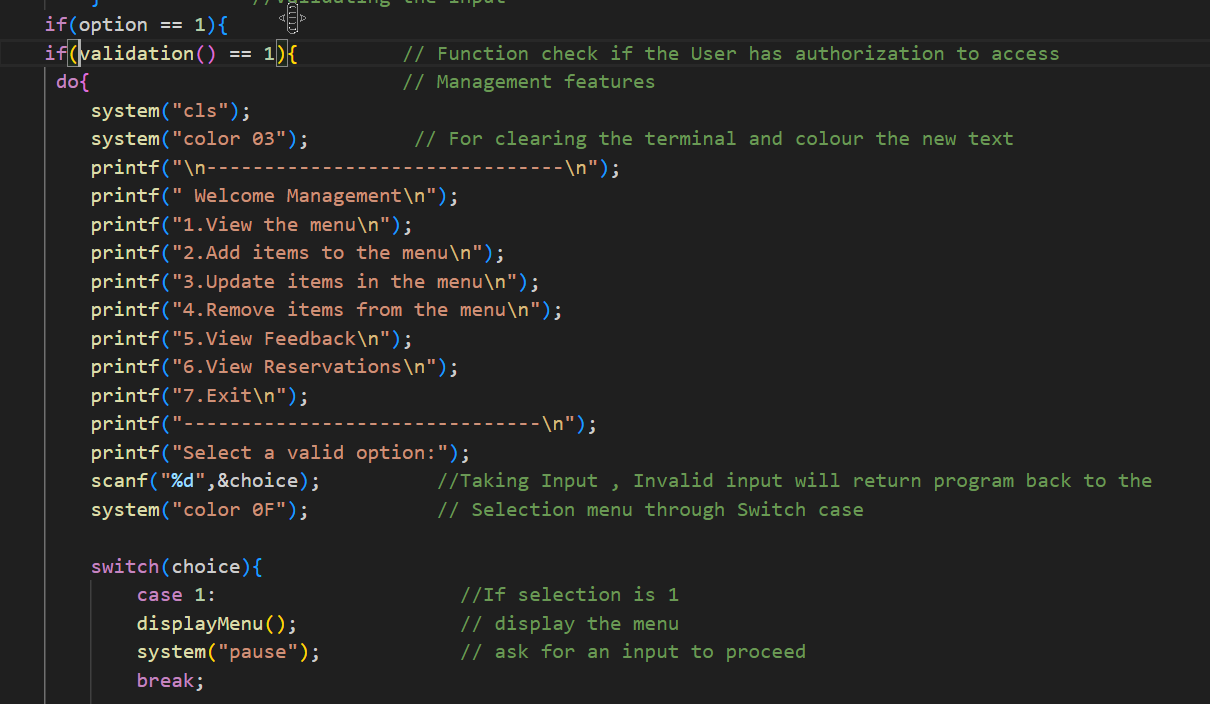
**

* *Using* ***nested if-else*** *with* ***pointer*** *to connect the order with the* ***cash system****. And giving discounts which will be applied to the order’s bill.*

**Main Function:**

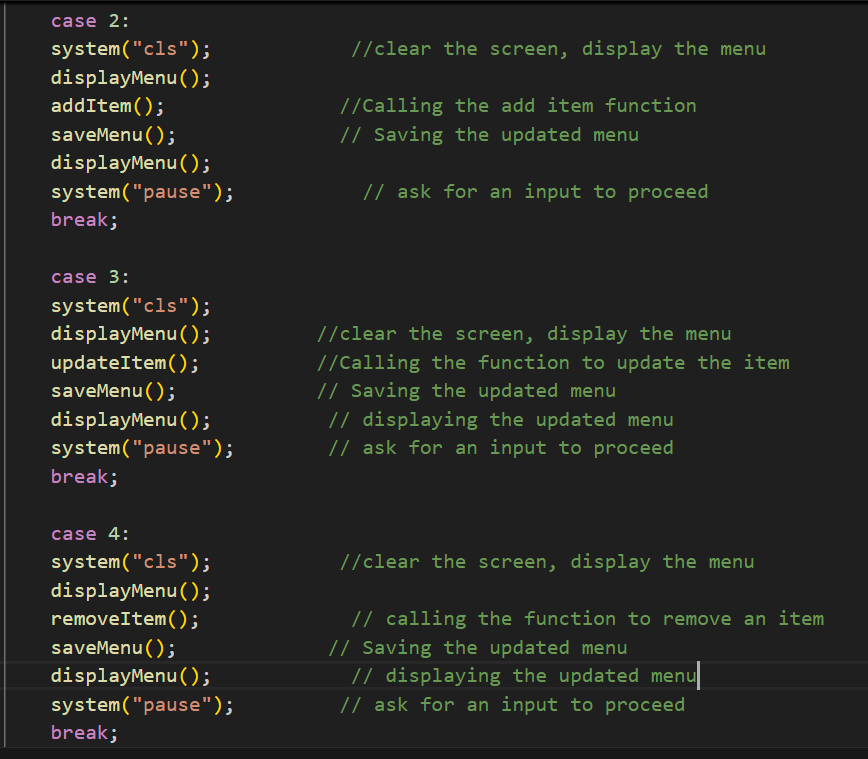
****

In the main function at the start few variables are declared. Them create menu function and initialize reservation functions are called. After that a user asked whether he/she wants to access management menu or customer menu. If an invalid input is given, user is asked to enter a valid selection again, until a valid input is given.



If user selects the management option, then validation function is called, if validation fails user is send to the first menu. If validation is successful user is shown the management menu and prompted to select an option. If an invalid selection is made, user is asked again. The menu loop ends when the user selects the exit option.

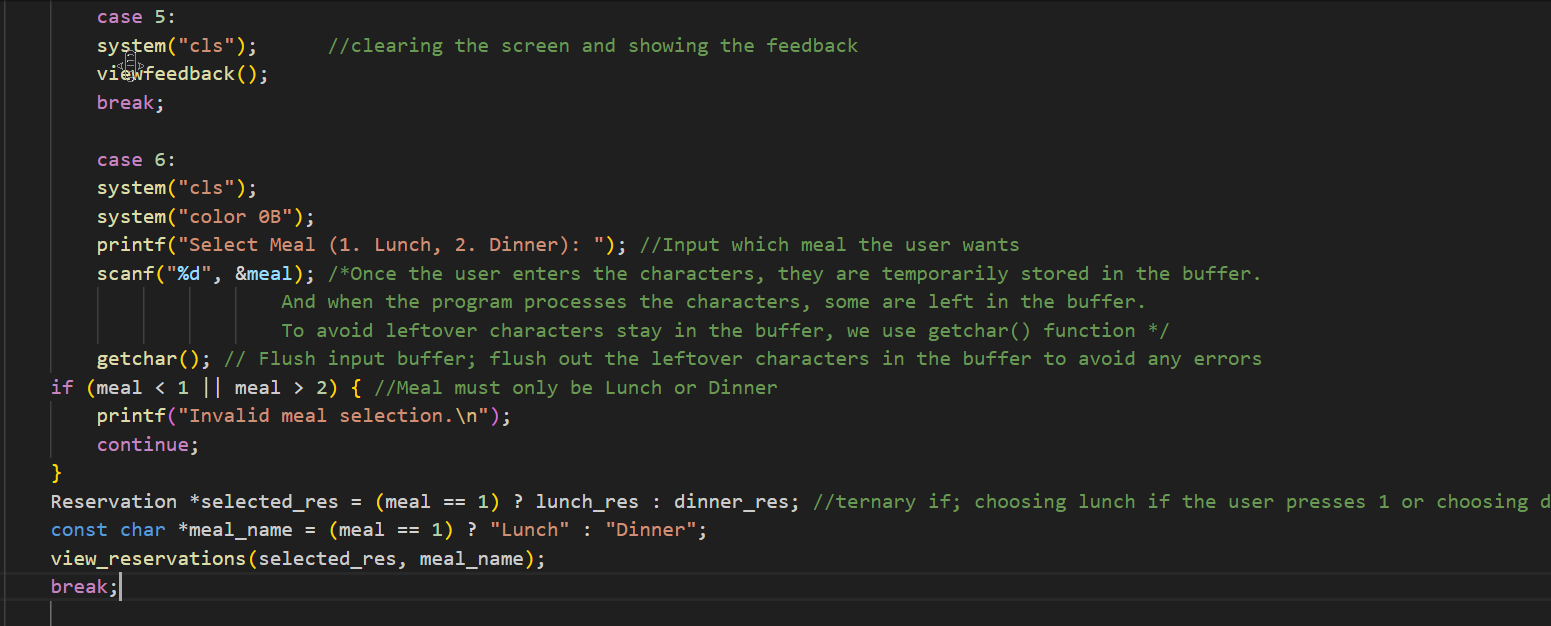
Option 1 calls the display menu function and waits for and input to proceed again with the program

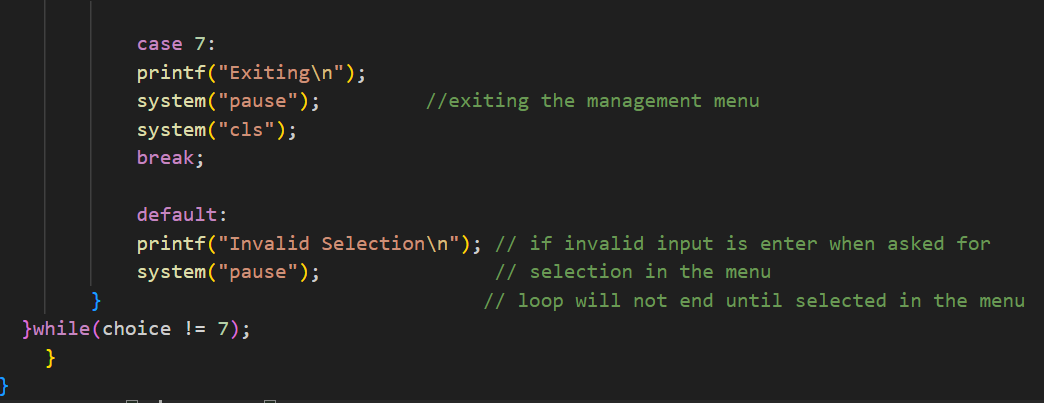


Option 2 clears the screen, calls the display menu function and calls the add item function. Then it saves the menu and calls the display menu function and waits for the user to enter to proceed.

Option 3 clears the screen, calls the display menu function and calls the update item function. Then it saves the menu and calls the display menu function and waits for the user to enter to proceed.

Option 4 clears the screen, calls the display menu function and calls the remove item function. Then it saves the menu and calls the display menu function and waits for the user to enter to proceed.

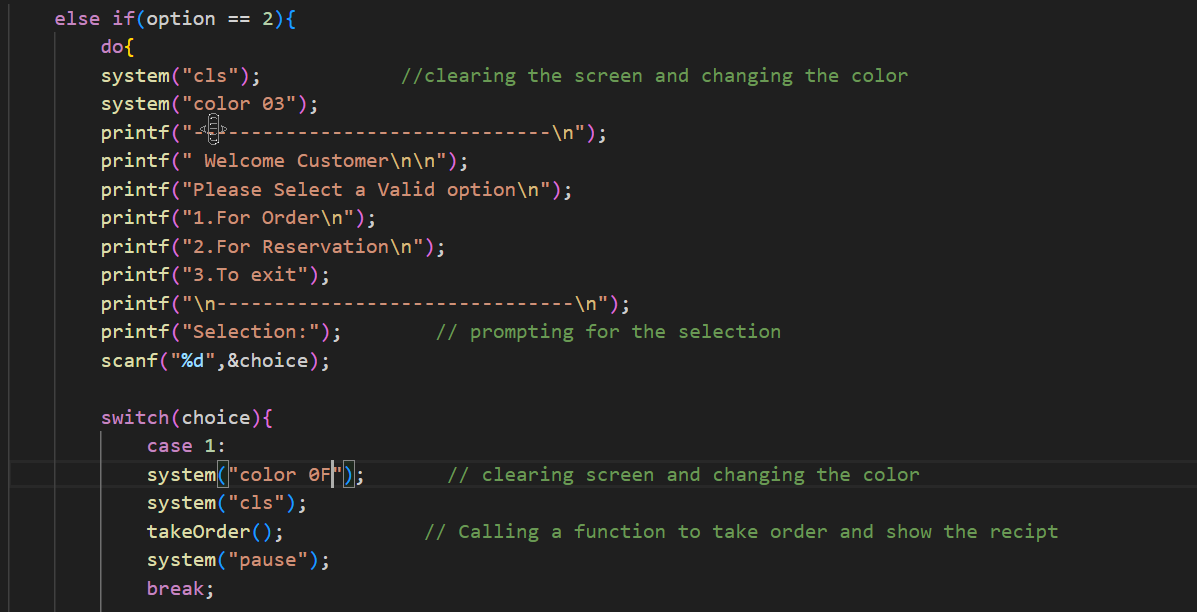




Option 5 clears the screen and displays the feedback if there are any

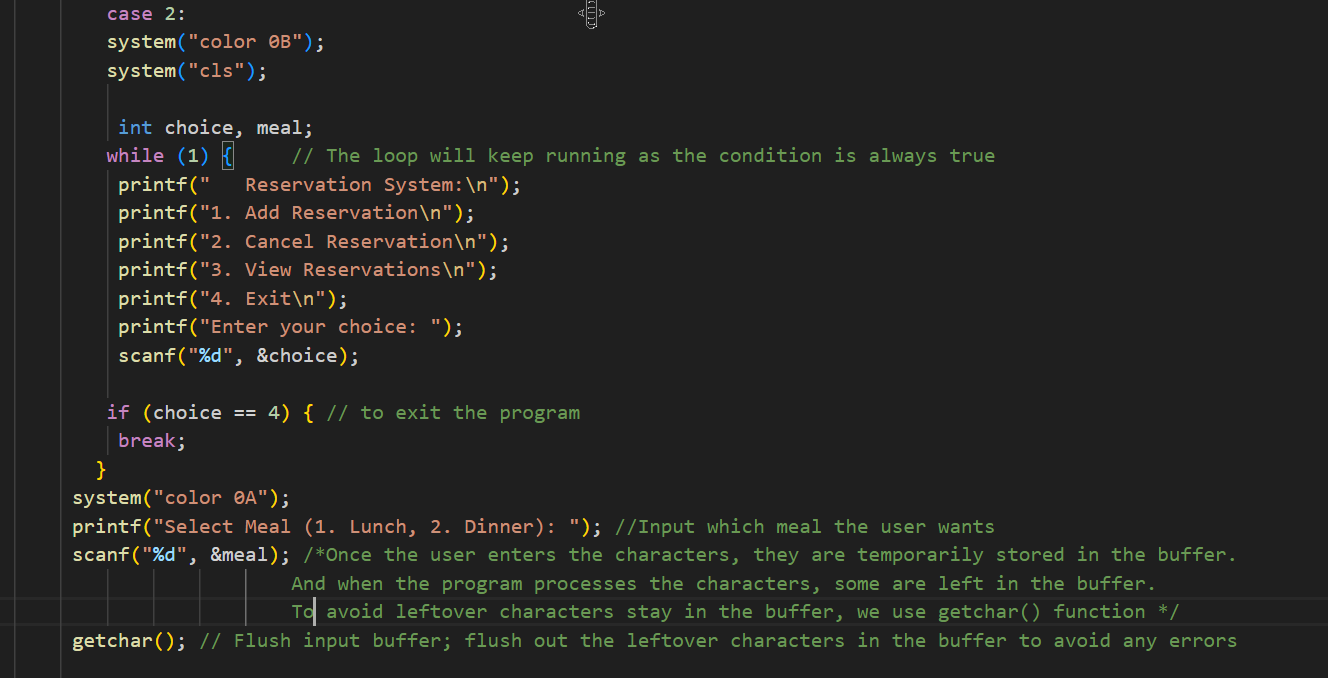
Option 6 clears the screen and changes the colour, then it prompts the user to select which reservation list user wants to see and then it calls the view reservation function according to the input

Option 7 exits the menu

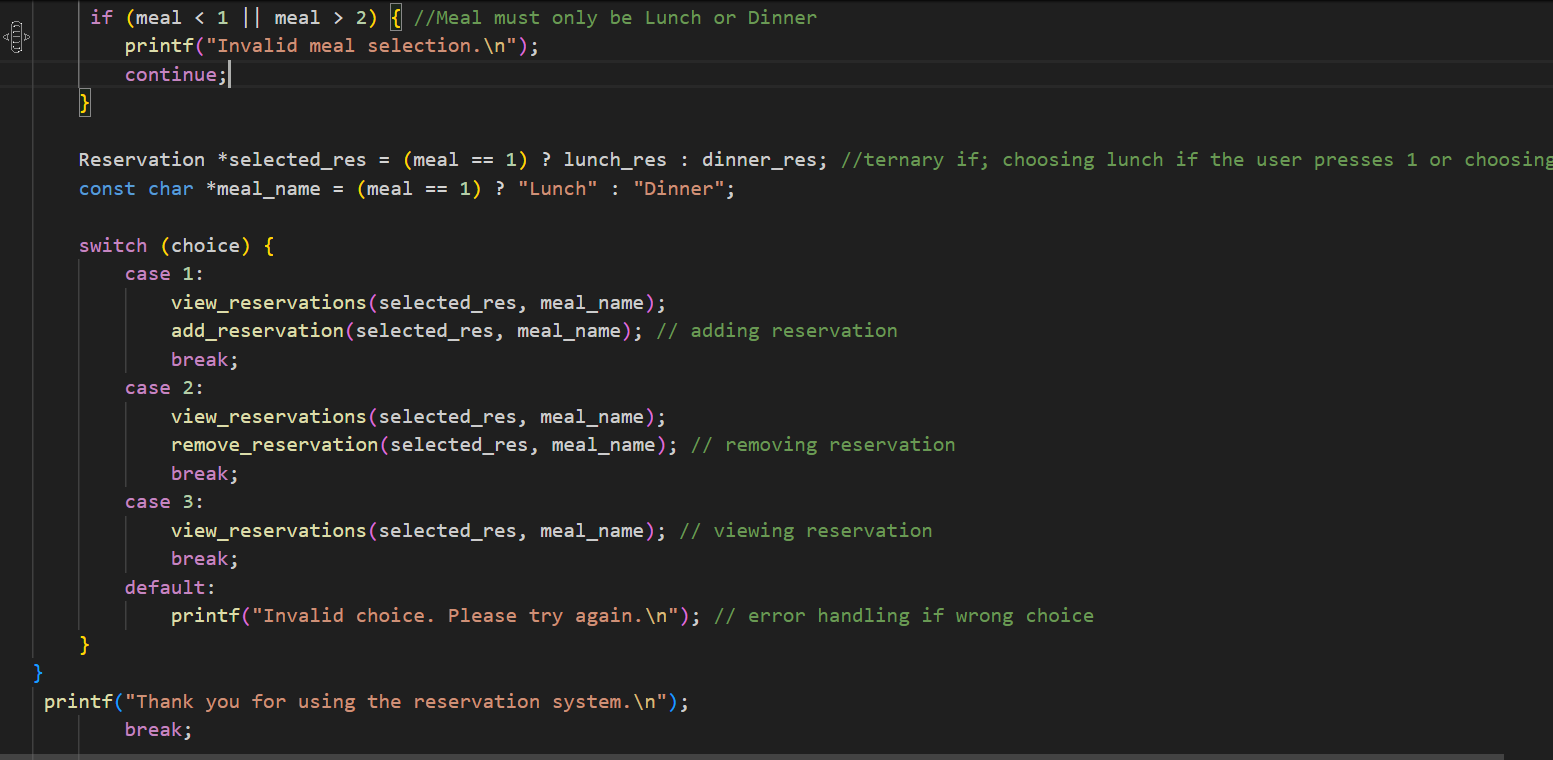


If customer option is selected then the customer menu is shown to the user, it clears the screen and changes the colour and prompts the user for the selection. If an invalid selection is made, user is asked to select again until a valid selection is made:

Option 1 clears the screen and changes the colour. Then take order function is called after which the programs pause before proceeding.



Option 2 clears the screen and changes the colour. An option to add, cancel, view reservation and exit is shown and user is asked for an input. A loop runs until user enters a valid selection. Then system colour is changes

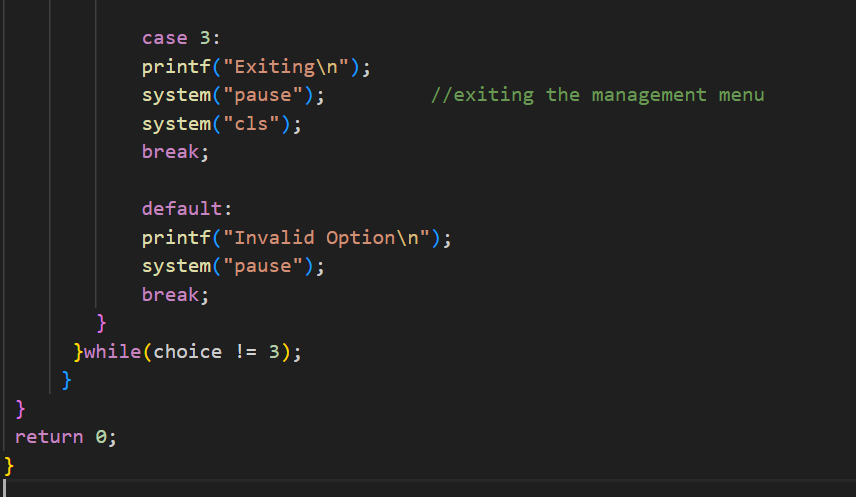


User is then prompted to select the meal for the reservation, in case option 1 in the reservation view reservation and add reservation functions are called.

In case of option 2 view reservation and remove reservation functions are called.

In case of option 3 view reservation function is called

in case of option 4 the menu loop is exited



When option 3 on the customer menu is selected the program ends.