July 2023

COFFE MACHINE PROJECT

Prepared by:P.M.Wickramarathna
Student ID: CS/2020/035

Description

This is a simple C code for a coffee machine. The coffee machine offers three types of coffee: cappuccino, latte, and espresso. Customers can enter money and select their desired coffee type, and the machine will dispense the coffee accordingly.

USAGE

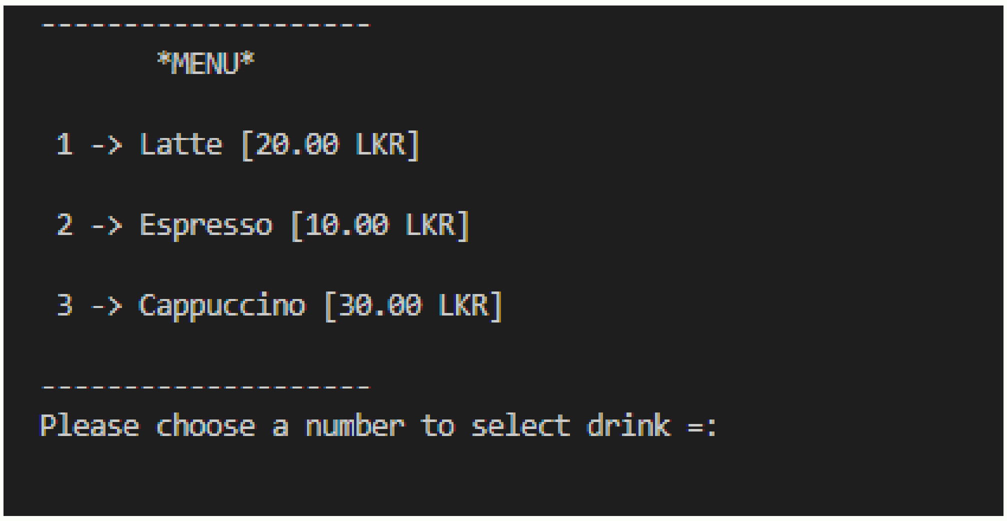
1. The program will prompt the customer to enter the coffee type (1 for cappuccino, 2 for latte, and 3 for espresso).

2. Next, the customer can enter the amount of money they want to insert to purchase the coffee.

3. If the entered amount is sufficient, the coffee will be dispensed, and the change (if any) will be returned to the customer.

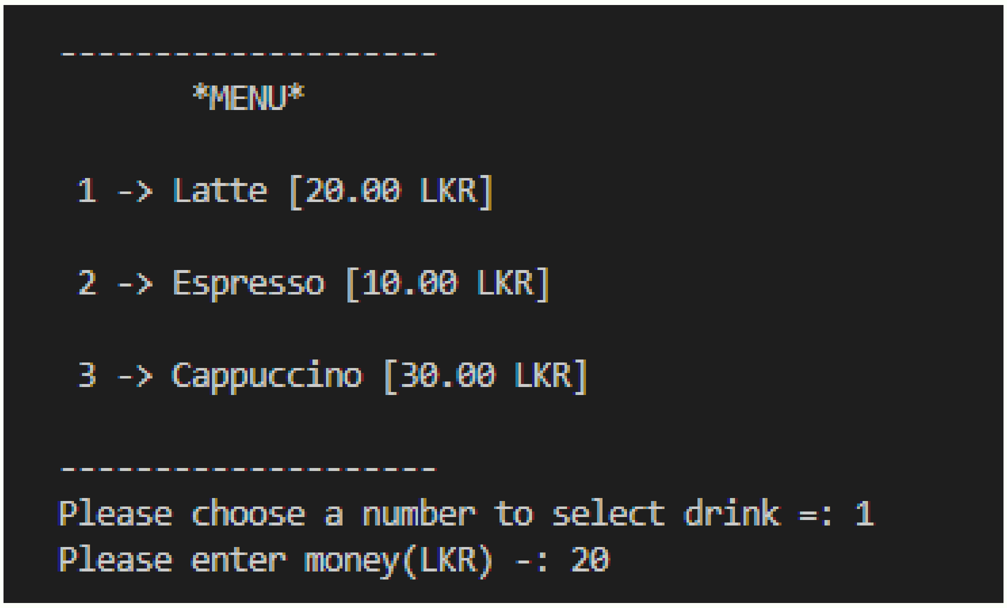
4. If the entered amount is not sufficient, transaction will cancel and money is retuned to the user and user will redirect to the menu

1. The program will prompt the customer to enter the coffee type .



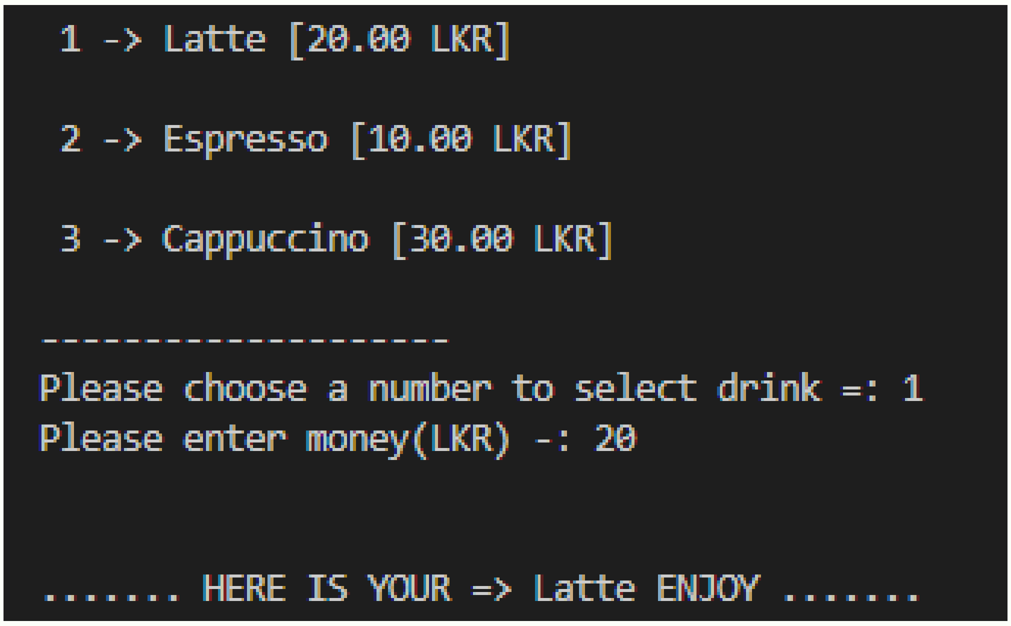
```
 --------------------
       *MENU*

  1 -> Latte [20.00 LKR]

  2 -> Espresso [10.00 LKR]

  3 -> Cappuccino [30.00 LKR]

 --------------------
 Please choose a number to select drink =:
```

2. customer can enter the amount of money they want to insert to purchase the coffee.



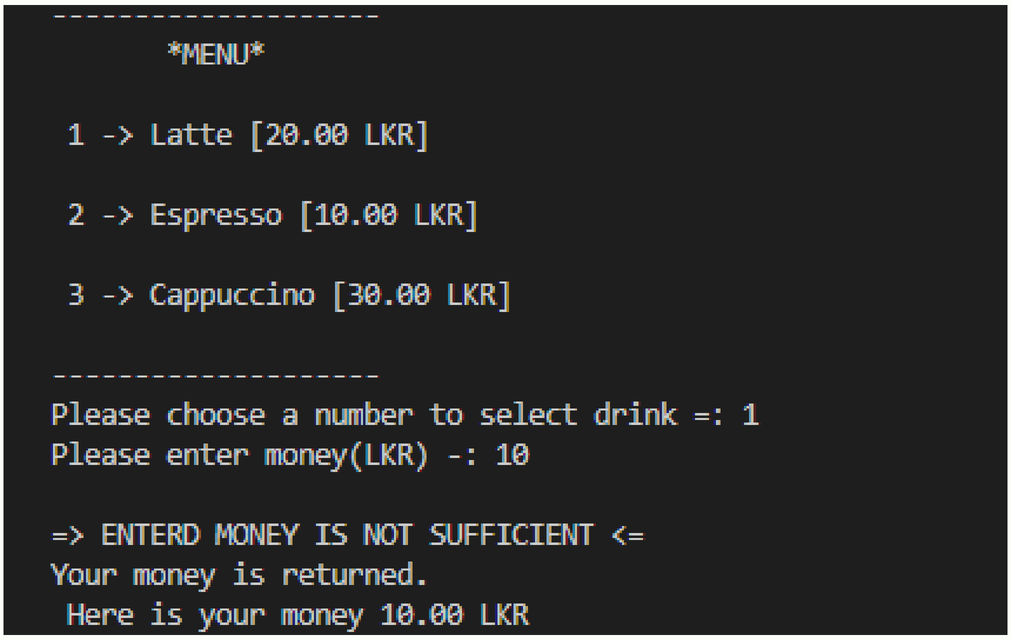
```
 --------------------
       *MENU*

  1 -> Latte [20.00 LKR]

  2 -> Espresso [10.00 LKR]

  3 -> Cappuccino [30.00 LKR]

 --------------------
 Please choose a number to select drink =: 1
 Please enter money(LKR) -: 20
```

3. If the entered amount is sufficient, the coffee will be dispensed, and the change (if any) will be returned to the customer.



```
 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

--------------------
Please choose a number to select drink =: 1
Please enter money(LKR) -: 20


....... HERE IS YOUR => Latte ENJOY .......
```

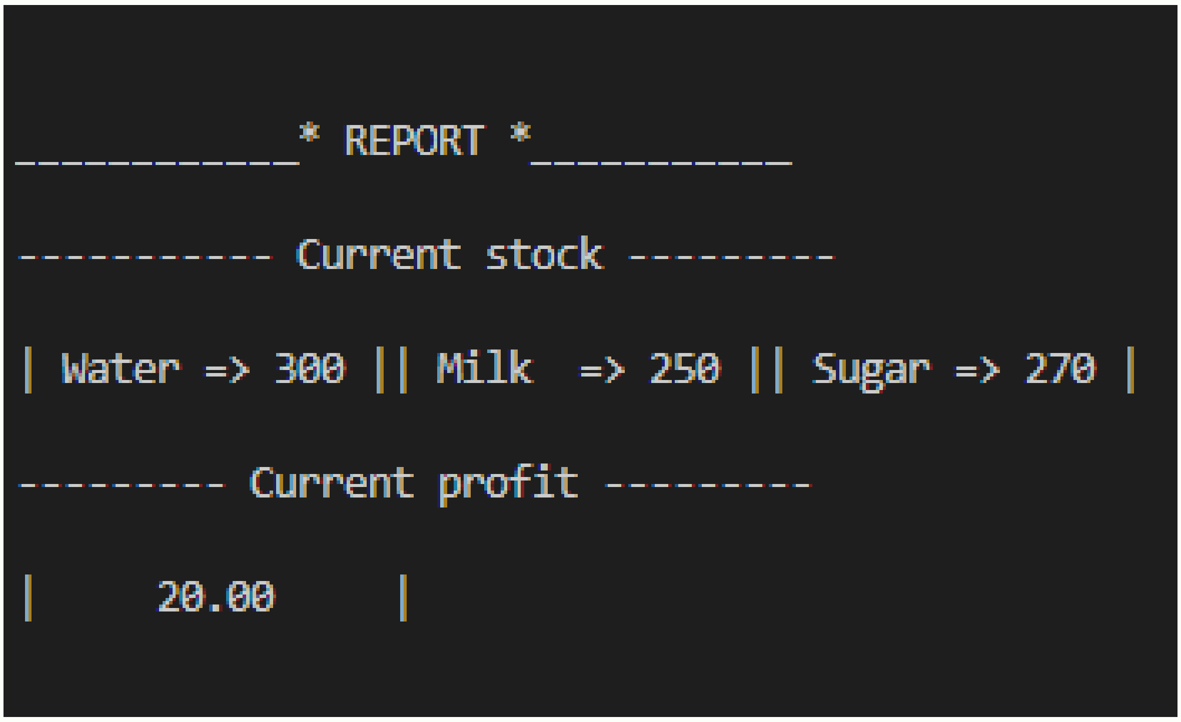
4. If the entered amount is not sufficient, transaction will cancel and money is retuned to the user and user will redirect to the menu



```
--------------------
       *MENU*

 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

--------------------
Please choose a number to select drink =: 1
Please enter money(LKR) -: 10

=> ENTERD MONEY IS NOT SUFFICIENT <=
Your money is returned.
 Here is your money 10.00 LKR
```

Owner's Report

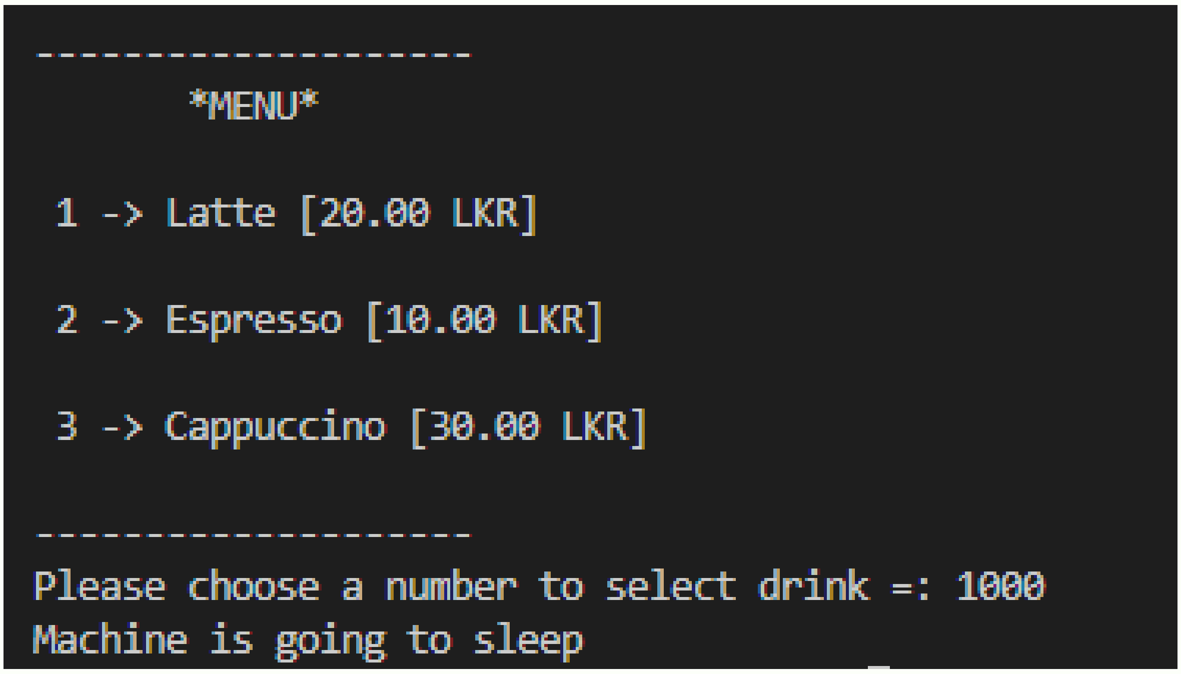
The owner of the machine can get a report of the current resource status , current profit of the coffee machine by using code 2000 as input.



```
_____________* REPORT *____________

------------ Current stock ----------

| Water => 300 || Milk  => 250 || Sugar => 270 |

---------- Current profit ----------

|     20.00     |
```

Turning Off the Machine

If the owner wants to turn off the coffee machine, they can press 1000.

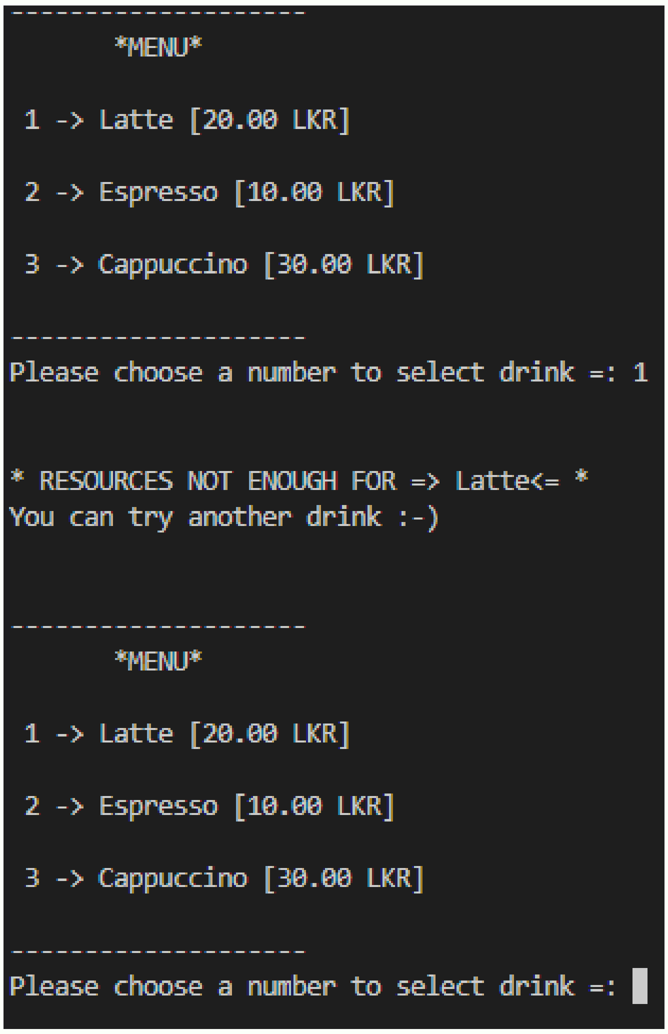


```
---------------------
       *MENU*

 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

---------------------
Please choose a number to select drink =: 1000
Machine is going to sleep
```

Important Notices

1

Resource Over Warning : If the resources of the coffee machine (e.g., coffee beans, milk, water) are depleted, the machine will not be able to grant coffee to the customers. In such cases, a warning message will be displayed, informing the customer that the resources of the machine are over. The customer will then be redirected to the main menu.

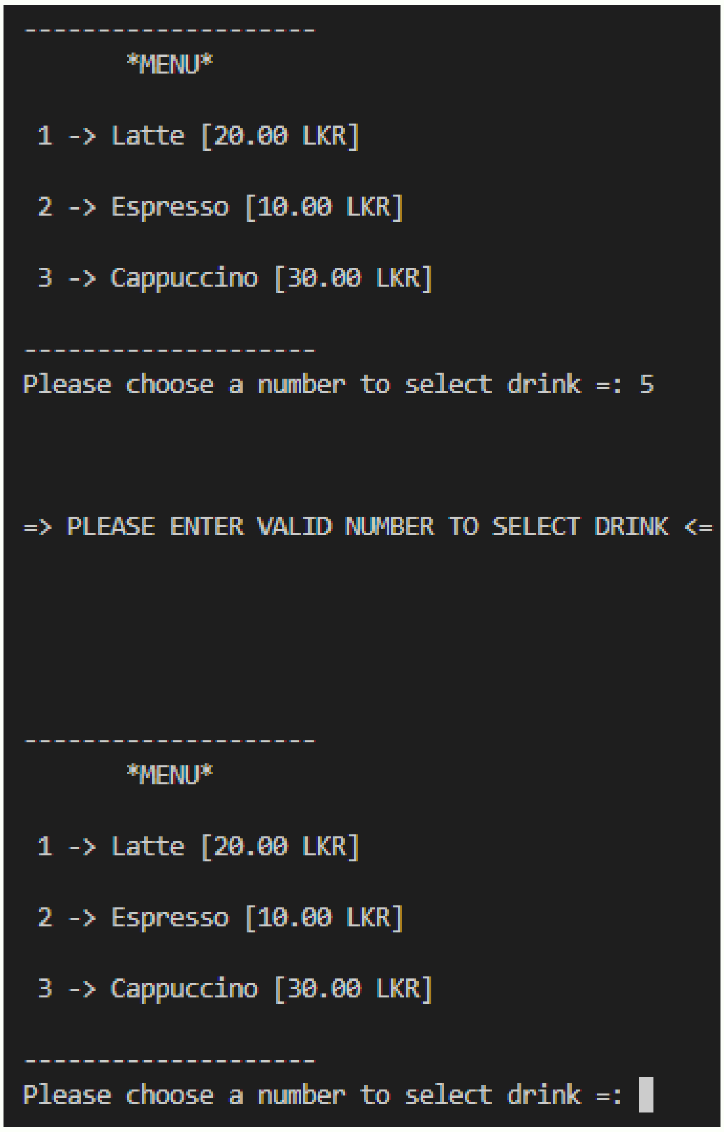


```
--------------------
       *MENU*

 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

--------------------
Please choose a number to select drink =: 1


* RESOURCES NOT ENOUGH FOR => Latte<= *
You can try another drink :-)


--------------------
       *MENU*

 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

--------------------
Please choose a number to select drink =:
```

Important Notices

2

Invalid Selection Warning : If the customer chooses a number that does not correspond to any of the available coffee types, a warning message will be displayed. The customer will be directed back to the main menu to make a valid selection.



```
--------------------
       *MENU*

 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

--------------------
Please choose a number to select drink =: 5



=> PLEASE ENTER VALID NUMBER TO SELECT DRINK <=





--------------------
       *MENU*

 1 -> Latte [20.00 LKR]

 2 -> Espresso [10.00 LKR]

 3 -> Cappuccino [30.00 LKR]

--------------------
Please choose a number to select drink =:
```

HOW CODE WORKS?

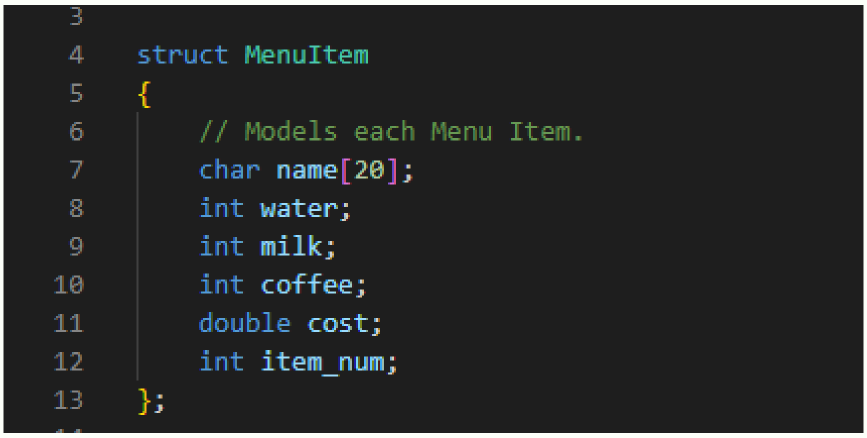
In our coffee machine system, the first step is to obtain the coffee number from the user. We then verify if the entered coffee number is present in our menu. If the selected coffee is not available, we promptly display an error message and redirect the user to the main menu. However, if the order is valid, we proceed to check if the machine has sufficient resources, such as milk, coffee, and sugar, to fulfill the user's request. If the necessary resources are not available, we display an error message and suggest the user choose another drink, redirecting them back to the main menu.

Once the machine verifies that the required resources are adequate, we ask the customer to enter the payment amount. If the entered money is insufficient for the selected coffee, we raise an error and return the money entered by the customer. However, if the payment is sufficient, the machine deducts the appropriate resources and prepares the coffee. We provide the customer with their desired coffee along with any change owed, if applicable.

For the machine owner, there are special codes to access certain functionalities. By pressing "2000," the owner can obtain a report on the current status of the machine's resources. This report helps the owner stay informed about the availability of coffee beans, milk, water, and other essential supplies. Additionally, if the owner wishes to turn off the coffee machine, they can do so by entering "1000." These codes are secret and known only to the owner, ensuring security and control over the machine's operations.

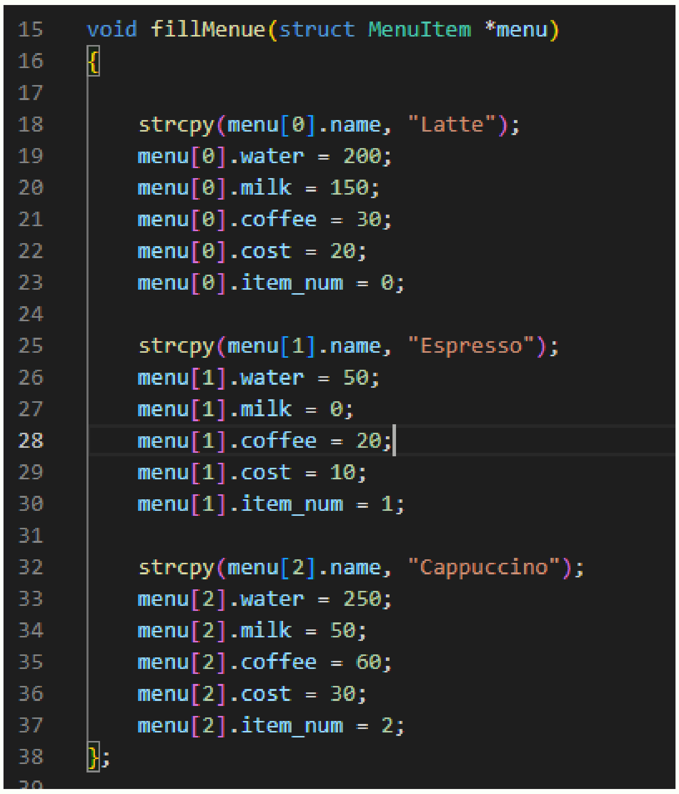
ABOUT CODE

This data structure use to store the characteristic of each coffee type



```
struct MenuItem
{
    // Models each Menu Item.
    char name[20];
    int water;
    int milk;
    int coffee;
    double cost;
    int item_num;
};
```

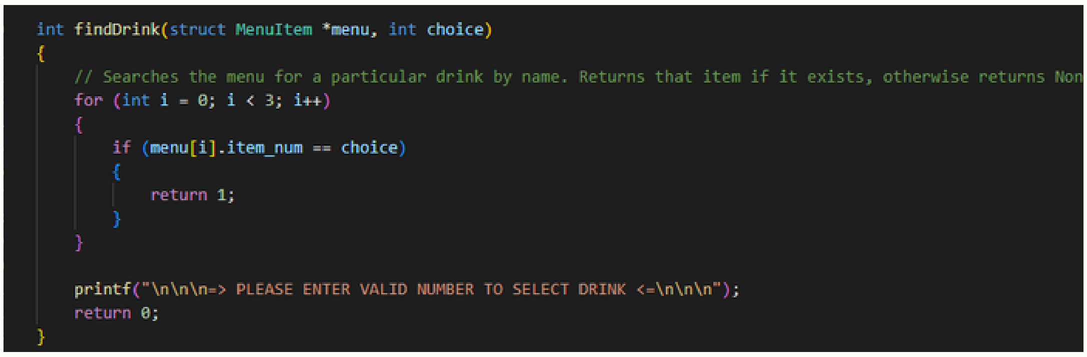
This function is responsible for specifying the amount of resources required for each coffee type to be made.



```
void fillMenue(struct MenuItem *menu)
{

    strcpy(menu[0].name, "Latte");
    menu[0].water = 200;
    menu[0].milk = 150;
    menu[0].coffee = 30;
    menu[0].cost = 20;
    menu[0].item_num = 0;

    strcpy(menu[1].name, "Espresso");
    menu[1].water = 50;
    menu[1].milk = 0;
    menu[1].coffee = 20;
    menu[1].cost = 10;
    menu[1].item_num = 1;

    strcpy(menu[2].name, "Cappuccino");
    menu[2].water = 250;
    menu[2].milk = 50;
    menu[2].coffee = 60;
    menu[2].cost = 30;
    menu[2].item_num = 2;
};
```

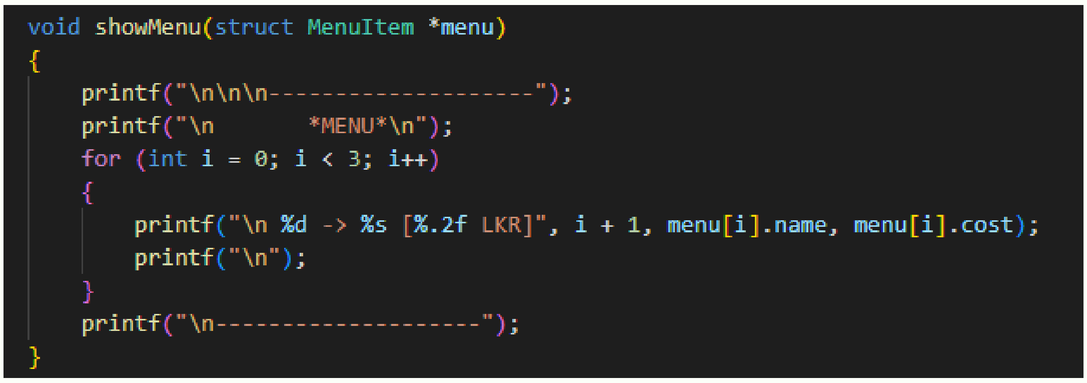
ABOUT CODE

This **findDrink** function serves the purpose of validating the user's drink selection by matching their input number with the available coffee types in the system. If the chosen drink exists, the function returns 1 to indicate its presence. However, if the input does not correspond to any available drink, the function provides a warning message and returns 0, prompting the user to make a valid selection.



```c
int findDrink(struct MenuItem *menu, int choice)
{
    // Searches the menu for a particular drink by name. Returns that item if it exists, otherwise returns Non
    for (int i = 0; i < 3; i++)
    {
        if (menu[i].item_num == choice)
        {
            return 1;
        }
    }

    printf("\n\n\n=> PLEASE ENTER VALID NUMBER TO SELECT DRINK <=\n\n\n");
    return 0;
}
```

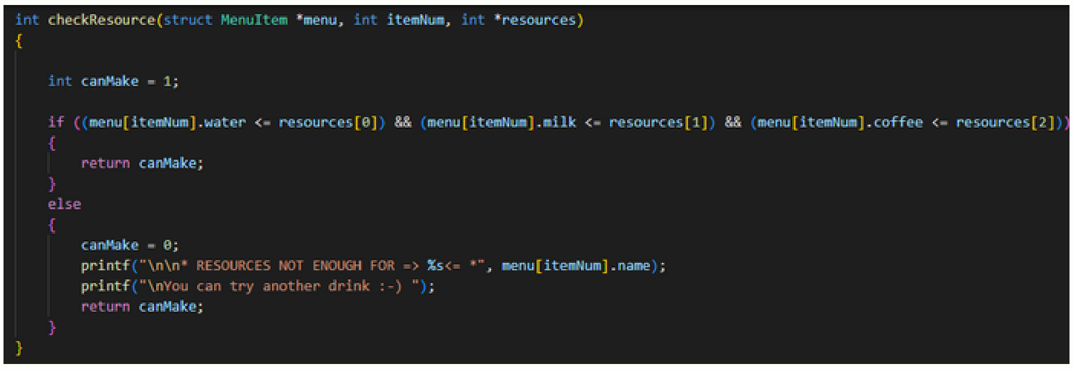
This function is responsible for showing menu to the user



```c
void showMenu(struct MenuItem *menu)
{
    printf("\n\n\n--------------------");
    printf("\n       *MENU*\n");
    for (int i = 0; i < 3; i++)
    {
        printf("\n %d -> %s [%.2f LKR]", i + 1, menu[i].name, menu[i].cost);
        printf("\n");
    }
    printf("\n--------------------");
}
```

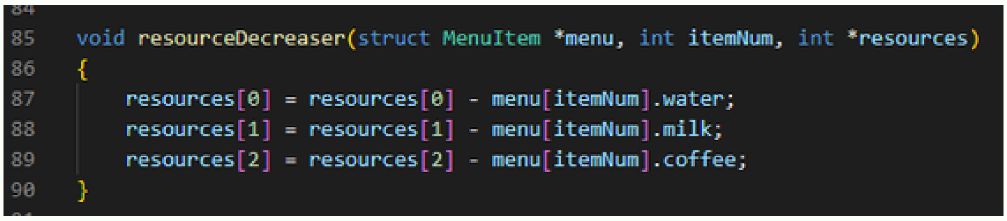
ABOUT CODE

When the user selects a coffee, this **checkResource** function verifies whether the available resources in the system are sufficient to fulfill the order. If the resources are adequate, the coffee is prepared; otherwise, an error message is displayed.



```c
int checkResource(struct MenuItem *menu, int itemNum, int *resources)
{

    int canMake = 1;

    if ((menu[itemNum].water <= resources[0]) && (menu[itemNum].milk <= resources[1]) && (menu[itemNum].coffee <= resources[2]))
    {
        return canMake;
    }
    else
    {
        canMake = 0;
        printf("\n\n* RESOURCES NOT ENOUGH FOR => %s<= *", menu[itemNum].name);
        printf("\nYou can try another drink :-) ");
        return canMake;
    }
}
```

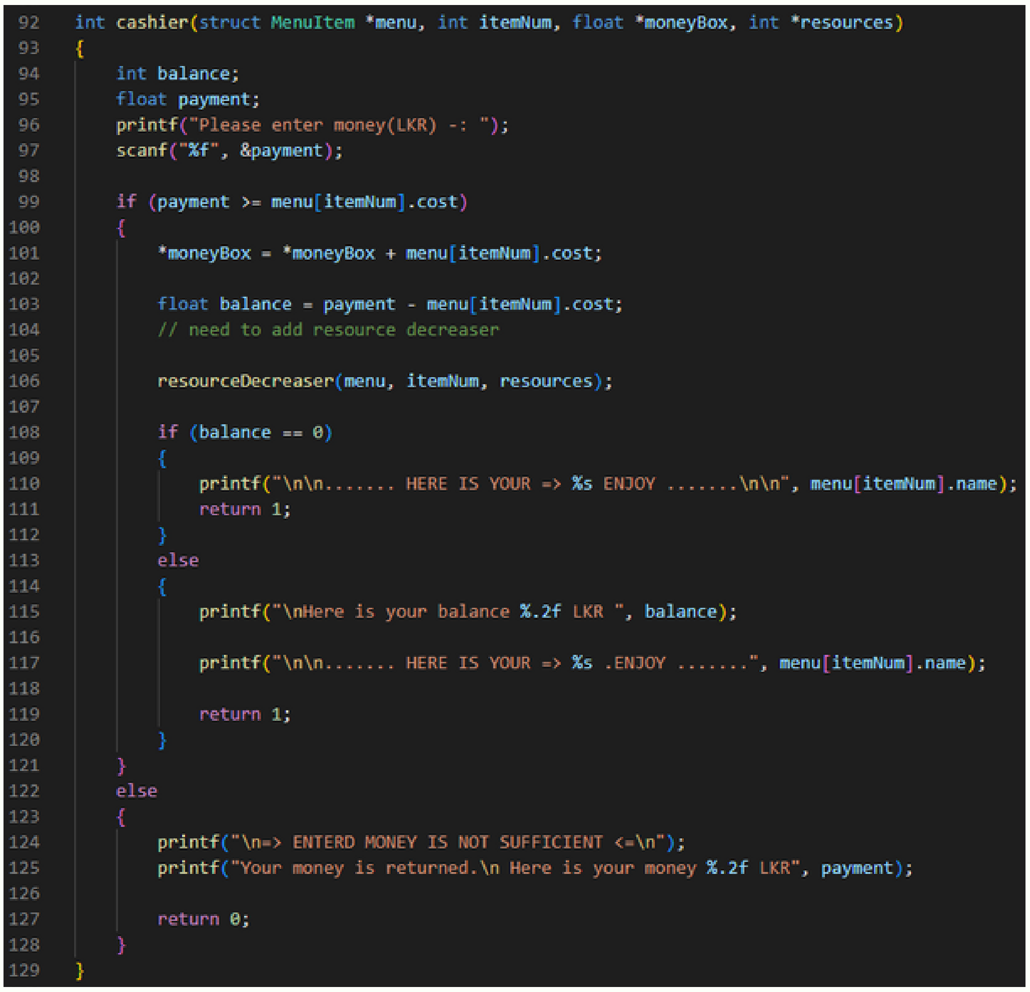
When coffe is made this function will decrease the system resources



```c
void resourceDecreaser(struct MenuItem *menu, int itemNum, int *resources)
{
    resources[0] = resources[0] - menu[itemNum].water;
    resources[1] = resources[1] - menu[itemNum].milk;
    resources[2] = resources[2] - menu[itemNum].coffee;
}
```

ABOUT CODE

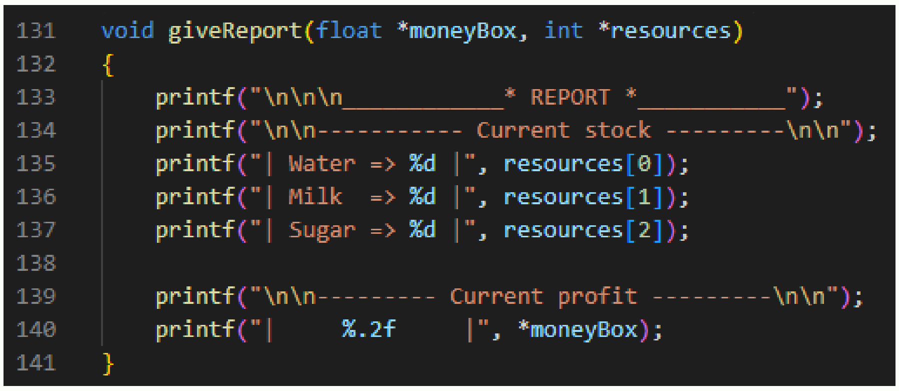
This function plays a crucial role in the coffee machine's operation. Its primary task is to receive money from the user. After receiving the payment, it checks if the amount is sufficient to purchase the selected coffee. If the user has provided enough funds, the function proceeds to verify if the coffee machine has an adequate supply of resources to fulfill the order. Only when both the user's payment and the machine's resources are sufficient will the coffee be prepared and dispensed. However, in cases where the user's payment or the machine's resources are insufficient, appropriate error messages are generated to inform the user of the issue.



```c
int cashier(struct MenuItem *menu, int itemNum, float *moneyBox, int *resources)
{
    int balance;
    float payment;
    printf("Please enter money(LKR) -: ");
    scanf("%f", &payment);

    if (payment >= menu[itemNum].cost)
    {
        *moneyBox = *moneyBox + menu[itemNum].cost;

        float balance = payment - menu[itemNum].cost;
        // need to add resource decreaser

        resourceDecreaser(menu, itemNum, resources);

        if (balance == 0)
        {
            printf("\n\n....... HERE IS YOUR => %s ENJOY .......\n\n", menu[itemNum].name);
            return 1;
        }
        else
        {
            printf("\nHere is your balance %.2f LKR ", balance);

            printf("\n\n....... HERE IS YOUR => %s .ENJOY .......", menu[itemNum].name);

            return 1;
        }
    }
    else
    {
        printf("\n=> ENTERD MONEY IS NOT SUFFICIENT <=\n");
        printf("Your money is returned.\n Here is your money %.2f LKR", payment);

        return 0;
    }
}
```

ABOUT CODE

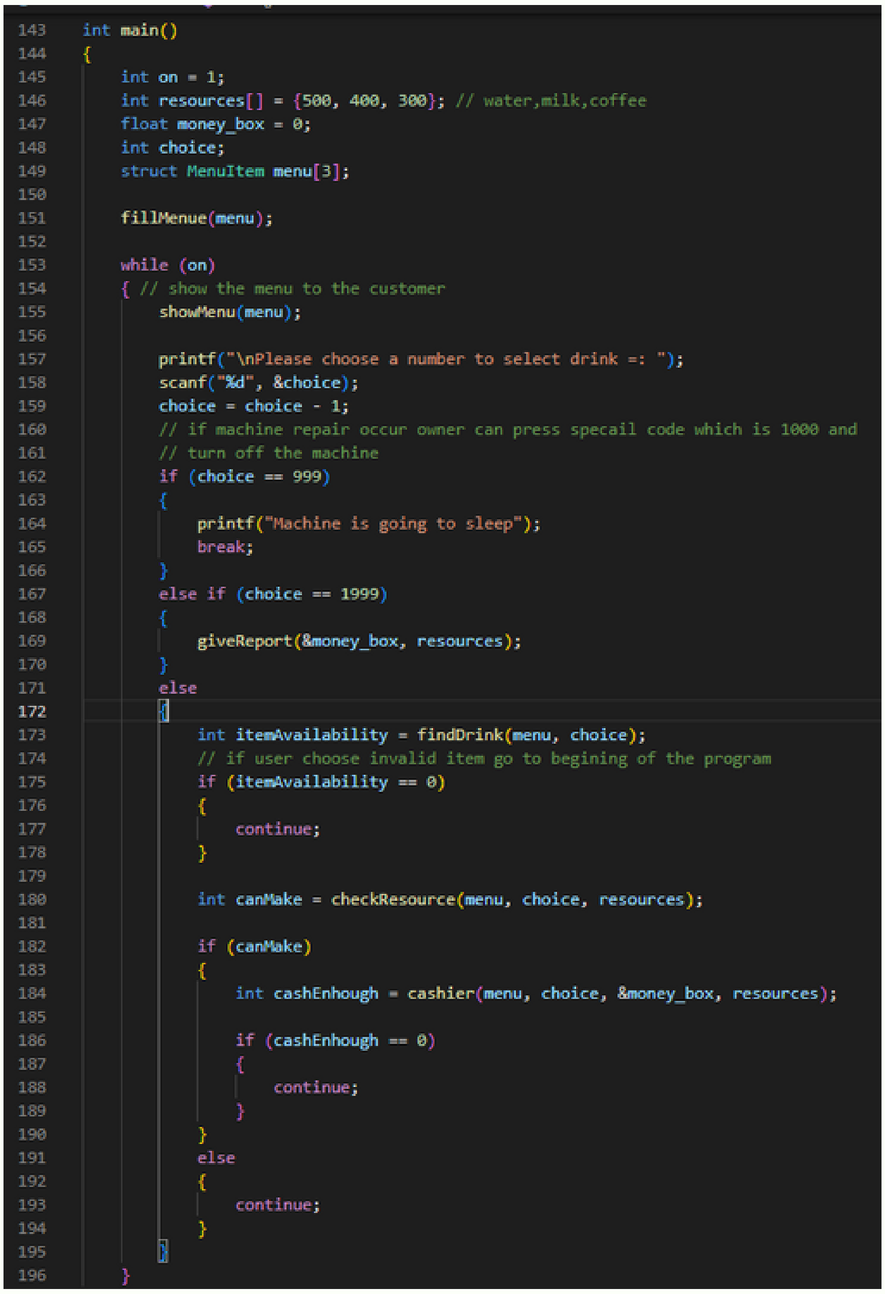
This function serves the purpose of providing the machine's owner with a status report upon request.



```c
void giveReport(float *moneyBox, int *resources)
{
    printf("\n\n\n____________* REPORT *___________");
    printf("\n\n----------- Current stock ---------\n\n");
    printf("| Water => %d |", resources[0]);
    printf("| Milk  => %d |", resources[1]);
    printf("| Sugar => %d |", resources[2]);

    printf("\n\n--------- Current profit ---------\n\n");
    printf("|     %.2f     |", *moneyBox);
}
```

ABOUT CODE

This is the main function, responsible for executing and coordinating the sub-functions, as mentioned in previous pages. The program utilizes a `while` loop to redirect the user to the home screen in case of any error messages or after the successful completion of their order.



```c
int main()
{
    int on = 1;
    int resources[] = {500, 400, 300}; // water,milk,coffee
    float money_box = 0;
    int choice;
    struct MenuItem menu[3];

    fillMenue(menu);

    while (on)
    { // show the menu to the customer
        showMenu(menu);

        printf("\nPlease choose a number to select drink =: ");
        scanf("%d", &choice);
        choice = choice - 1;
        // if machine repair occur owner can press specail code which is 1000 and
        // turn off the machine
        if (choice == 999)
        {
            printf("Machine is going to sleep");
            break;
        }
        else if (choice == 1999)
        {
            giveReport(&money_box, resources);
        }
        else
        {
            int itemAvailability = findDrink(menu, choice);
            // if user choose invalid item go to begining of the program
            if (itemAvailability == 0)
            {
                continue;
            }

            int canMake = checkResource(menu, choice, resources);

            if (canMake)
            {
                int cashEnhough = cashier(menu, choice, &money_box, resources);

                if (cashEnhough == 0)
                {
                    continue;
                }
            }
            else
            {
                continue;
            }
        }
    }
```