|  |
| --- |
| Mechatronics Program |
| Faculty of Engineering |
| The British University in Egypt |





|  |  |
| --- | --- |
| Name | ID |
| Ghada Hany | 214516 |
| Lama Hamdi (Leader) | 209721 |
| Zahy Youssef | 209816 |



**Programming In C**

**21MTRN10C**

**Mechatronics & robotics engineering**

**year one**

**2021-2022**

**“Metro Ticket Vending machine”**

**Ghada Hany 214516**

**Lama Hamdi 209721  “group leader”**

**Zahy Youssef 209816**

Table of Contents

[Introduction 2](#_Toc102760586)

[Objectives 3](#_Toc102760587)

[Problem Specification 3](#_Toc102760588)

[Data 6](#_Toc102760589)

[Evaluation Criteria 6](#_Toc102760590)

[Approach 6](#_Toc102760591)

[Results and Analysis 7](#_Toc102760592)

[Detailed Flowchart 20](#_Toc102760593)

[Development 21](#_Toc102760594)

[Conclusions 21](#_Toc102760595)

[Appendix 22](#_Toc102760596)

[Acknowledgment 27](#_Toc102760597)

[References 28](#_Toc102760598)

# Introduction

A vending machine is a money operated machine that sells a variety of goods, including food, beverages, water, tickets, and other items. A metro ticket vending machine based on Cairo's metro line stations is designed in this project to assist users in getting their tickets in the quickest and most effective manner. Vending machines provide numerous advantages, such as requiring no human effort, saving time, etc.



Figure 1 Metro Ticket Vending Machine.

# Objectives

This program makes it easy for users to purchase metro tickets with less effort needed. This program is free from error and meet the user’s requirements.

# Problem Specification

This program is important these days since it reduces the amount of time spent interacting with people and reduces the spread of diseases such as Corona, which is something any programmer should consider nowadays when creating an application that will be used in public. Furthermore, this program saves time by reducing congestion, particularly in public transportation, also it has a lot of simple features that make it easy for the user to use it. This program consists of four libraries that make coding easier and more professional to use. In this application, our team focused on building simple code that any programmer could understand, and so that any user could handle easily.

This program contains multiple functions such as:

1- A function that asks the user for the station's name.

2- A function that calculates the ticket's value according to the specified station, after which the user is allowed to enter the number of people, or the quantity of tickets required from user.

3- A function that asks whether the user has a metro discount card; If the user has a discount card, they enter a specific number, then the program calculates the discount value and displays the total ticket price to the user.

4- If the user does not have any discounts, the program calculates the price of a single ticket and multiplies it by the quantity of tickets to provide the user with the total ticket price.

5- A function that requests the money from the user, then calculates the change by subtracting the money entered from the overall cost required.

7- Finally a function that prints the ticket color, and displays a bill with the destination selected, ticket quantity, overall cost, money paid and change (if exists).

Our group was able to avoid any error that could be entered by the user so as not to spoil the entire program. This program has been successfully executed. As conclusion, the vending sector has seen significant growth over the years, and it will continue to do so in the years to come.

To summaries, our team was able to avoid any potential user error, which would have spoiled the entire program, and it provided numerous functions to meet the user's requirements. This program was completed successfully.

  
Figure 2 Tickets Guide.

# Data

Input:

* Final station
* POD (determination) card
* Senior Citizen card
* Number of tickets
* Money
* Remaining money if needed

Output:

* Total money paid
* Change
* Ticket colour

# Evaluation Criteria

The system of this program is quite powerful because it already contains a lot of features that can handle any issue expected from the user. As a result, the evaluation of this program will be 10 / 10.

# Approach

Other than the syntax of the C programming language, there were no obvious errors that occurred when implementing the code. However, our group considered any unexpected error that could be added by the user in the choices given, and therefore built a short function to prevent these errors. If a user enters a letter instead of a number, for example, the program will instantly return to the previous step, allowing the user to re-enter the correct decision. Additionally, our team provided several specialized alternatives to make it easier for the user to choose without making any mistakes.

# 

# Results and Analysis

These lines (*#include*) are called preprocessor directive I.e., it is directed to the preprocessor or speaks to it as to make functions used in the program acquaint or known to the preprocessor. They contain the function prototypes  

*#include*<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<math.h>

1. *#include*<stdio.h>

The first preprocessor directive says to the preprocessor to include a file or library called stdio.h which is an abbreviation for standard input and output header file which contains or teaches the preprocessor two main functions:   
*scanf*() which takes characters from the users.   
*printf*() which displays characters to the sceen.

1. #include<string.h>

Secondly, this library allows the program to handle string functions. Strings are data type that have multiple characters like words or sentences. For example, %s which allows to *printf*() or *scanf*() in the format of a string  I.e., with the help of this function we were able to replace the arduous ” for loops” that were used to print() or *scanf*() character arrays.   
Another string function that was used in the program is strcmp() which compares two strings if the two strings are equal it yields zero value (0), else if they are not equal it prints 1.

1. #include<stdlib.h>

This library contains the function prototype of abs(), which makes the value of whole integer number absolute I.e., not negative.

1. #include<math.h>

Contrastingly, the function prototype of fabs(), is contained in this preprocessor directive which returns the absolute value of a number containing decimal like float or double.

float Quantity (float price);

this line is a function prototype which declares the Quantity function used later on in the program. It takes the following form:   
return data type – name of function – (type of arguments or inputs passed).    
Thus, this function takes in a variable named price of float data type, and returns an integer value.

float monchang(float Total\_price, char colour[6]);

Likewise, this function prototype declares a function named monchang which is used later on in the program to compute the change in money. This function has two arguments Total\_price which is of data type float and colour which is of data type character array that reserves 20 consecutive memory locations.

Note: both of these function prototypes declaration could have been neglected if we had written the functions with their body directly in the first place in the program, before the main function.

int main()   
This is the main function, which contains all other sub functions. Therefore, this function doesn’t receive any arguments and returns 0 as a sign that the program is successfully executed while other non-zero values indicate failure.

char s1[5], s2[5] = "back", s3[2] = "2";

a variable named s1 and s2 and s3 are declared as character array data type representing three strings, and s3 is initialized with a constant value of ‘2’ as will be used to return back later in the program. Even though it is initialized with “2”, we declared it as character array since variables are required to be of string format when compared together later in the program using strcmp().

Note: a character hold 1 byte in memory, so s1 & s2 array holds

6 ×1=6 bytes6 ×1=6 bytes

, while s3 occupies 2 bytes.

    int n, discount;

two variables n & discount are declared as integers specifying the station number, and type of discount respectively.

Note: an integer occupies a space of 4 bytes in the memory.

    float price, Total\_price;

two variables are declared or identified as float data types, as both of these variables will hold money that may contain decimal places.

Note: a float occupies a space of 4 bytes in the memory.

    char colour[6];

A variable of data type character array of size 6 characters, this variable holds the ticket’s color, depending on the station’s category.   
Note: a character holds 1 byte in memory, so this array holds

6 ×1=6 bytes6 ×1=6 bytes

.

Do

This is a do while loop which is a post-test loop. Therefore, the code contained within the loop is executed at least once before the condition is tested and whether the condition is fulfilled or not. The do while loop was used to allow the user to go back to previous step if he wants to change his choice.

{

This curled bracket begins the do while loop.

printf("\nWelcome to Cairo's Green Metro Line! \n");

   printf("\nYou are in Adly Mansour station \n");

   printf("\n 1- Haykstep                   2- Omar Ibn El Khatab                     3- Qubaa \n 4- Hisham Barakat             5- El Nozha                               6- El Shams Club \n 7- Alf Masken                 8- Heliopolis \n");

These printf (s) statements displays welcoming message for the user and lists the Cairo’s metro line stations.

   printf("\nEnter the equivalent No. of your final station : ");

This printf (s) statements ask the user to enter the No. of his final station.

   scanf("%d", &n);

This scanf statement takes from the user the No. of his final station.

   printf(" Your equivalent station's No. is : '%d' \n", n);

Displays to the user his station’s No.

   printf("\npress '1' to enter POD(Determination) card, '2' for Senior Citizen Card, else enter any No. to continue : ");

Ask the user if he has a people of determination card to enter, or senior citizen card in order to apply the discount effects upon the ticket prices.

Note: there should be file handling functions in this step that scans through external file and verifies the user’s card code.

   scanf("%d", &discount);

Takes from the user his type of discount: 1 for POD & 2 for senior citizens.

}while(strcmp(s1,s2) == 0 || strcmp(s1,s3) == 0 ||  n > 8 || n < 1 || discount > 64);

This do-while loop repeats the following code:   
nested if-else if statement. A nested if- else if statement is an if nested inside another if statement. In other words, an external if statement that contains another internal if -else if statement. As can be illustrated in this example:

if(n>=1&&n<=3){

This is an outer or **outer if** statement containing if – else-if – else statements inside it.   
Note:  that the curly brackets ‘{‘ were placed considering the multiple statements inside it (compound statement).

       if(discount == 1)

       price=0.50;

This is an **inner If** statement.

Note: No need for curled brackets since it contains only one statement.

       else if(discount == 2)

       price=((float)5/2);

This is an **inner else-if** statement.

Note: No need for curled brackets since it contains only one statement.

Note: (float) was used in order to get the result of 5/2 as decimal I.e., ‘2.5’ EGP other wise the operation would give an integer value of ‘2’

       else

       price = 5;

This is an **inner else** statement.

Note: No need for curled brackets since it contains only one statement.

 char colour[20] = "Purple";

The character array named variable is assigned “purple” which represents the ticket color for the stations from 1 to 3.

 }

This curled bracket ends the previous outer if statement.

      else if(n>=4&&n<=6){

This is an external or **outer else-if**  statement containing if – else-if – else statements inside it.   
Note:  that the curly brackets ‘{‘ were placed considering the multiple statements inside it (compound statement).

          if(discount == 1)                                        /\*\*\*\*\*\*Nested Else - If-Else-If\*\*\*\*\*\*/

       price=0.50;

This is an **inner If** statement.

Note: No need for curled brackets since it contains only one statement.

       else if(discount == 2)

       price=((float)7/2);

This is an **inner else-if** statement.

Note: No need for curled brackets since it contains only one statement.

Note: (float) was used in order to get the result of 7/2 as decimal I.e., ‘3.5’ EGP otherwise the operation would give an integer value of ‘3’

        else

       price = 7;

This is an **inner else** statement.

Note: No need for curled brackets since it contains only one statement.

         char colour[20] = "Blue";

The character array named variable is assigned “Blue” which represents the ticket color for the stations from 4 to 6.

      }

This curled bracket ends the previous outer else-if statement.

   else if(n>=7&&n<=8){

  Outer Else-If .

   if(discount == 2)

  price=((float)10/2);

This is an inner If.

Note: (float) was not essential here since the result of 10/2 is always an integer, but it was added for the uniformity of the code to match the previous sequence maintained.

       else if(discount == 1)

       price=0.50;

This is an inner Else-If.

        else

       price = 10;

This is an inner Else.

         char colour[20] = "Green";

the character array named variable is assigned “Green” which represents the ticket color for the stations from 7 to 8.

      }

This curled bracket ends the previous outer else-if statement.

else

This is an outer else statement.

printf("your station is not found \n");

If the user enters a station No. greater than 8 or less than 1, a message “your station is not found” will be displayed.

printf(" Ticket price = %2.2f EGP \n", price);

prints the ticket price.

   printf("type 'back' or press '2' to go back, else enter any key to continue: ");

Displays message for the user to return back or continue.

   scanf("%s",  s1);

scans from the user s1, which will determine if the user wants to go back or continue.

}while(strcmp(s1,s2) == 0 || strcmp(s1,s3) == 0 ||  n > 8 || n < 1 || discount > 64);

   Total\_price = Quantity(price);

   monchang(Total\_price, colour);

This is a do while loop which is a post-test loop. Therefore, the code contained within the loop is executed at least once before the condition is tested and whether the condition is fulfilled or not. The do while loop was used to allow the user to go back to previous step if he wants to change his choice.   
The condition of the loop is fulfilled if any one of following five conditions is true:

1. strcmp(s1,s2) == 0

if the user types “back” when the program scanf for s1, this condition compares it to s2 if both are equal that is s1 = s2 = back, and the result is 0 then the condition is fulfilled, and the loop is repeated.

1. strcmp(s1,s3) == 0

if the user types “2” when the program scanf for s1, this condition compares it to s3 if both are equal that is s1 = s2 = 2, and the result is 0 then the condition is fulfilled, and the loop is repeated.

1. n > 8
2. n < 1

If the user enters a station’s No. greater than 8 or less than 1 which does not exist the loop is repeated after a message “your station is not found” is displayed.

1. discount > 64);

If the user enters a No. greater than 64 in the discount variable, the loop begins again. This is because discounts have only two options either ‘1’ for POD(Determination) card or '2' for Senior Citizen Card, else the user can enter any other No.

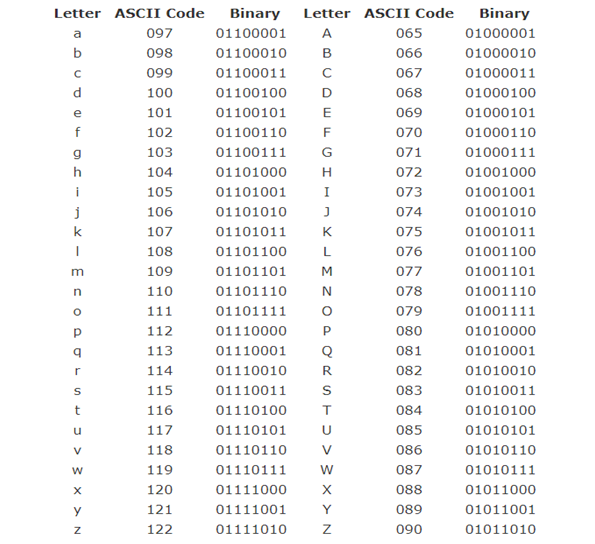
**Important note: we discovered that the program gives nonsense values if the user entered an alphabet instead of a No. in this variable. Therefore, we made 64 as the end limit for this variable this is due to the fact that the smallest letter value equals ‘65’ which represents an upper-case A.**

Note:

== is a relation operator used for comparison like in conditions. Unlike, just a single = which is an assignment operator used for variables to assign them.

< & > is a relational operator used for comparison denoting lesser than and greater than.

|| is a logical operator means || which is true if either condition is fulfilled.



float Quantity (float price)                                  /\*\*\*\*\*\*Quantity function\*\*\*\*\*\*/

Return data-type – function name – (argument or input type).

This is a new function called Quantity, that takes a float argument ‘price’ from the main function and returns float value.

The purpose of this function is to calculate the total price after multiplying the quantity taken from the user by the unit price of ticket.

{

This curled bracket begins the new function ‘Quantity’.

int quan;

Declaration of a local variable named ‘quan’ as integer.

float Total\_price;

Declaration of a local variable named ‘Total\_price’ as float.

char s4[8], s5[8] = "confirm", s6[2] = "1";

Declaration of three local variables as character array named ‘s1’, ‘s5’ & ‘s6’. Additionally, s5 is initialized as “confirm” and s6 is initialized as ‘1’.

Note: that since a char occupies a space that equals 1 byte, s4 & s5 will have 8 bytes of memory space, while s6 will have 2 bytes only.

 do{                            /\*\*\*\*\*\*Do While Loop\*\*\*\*\*\*/

This is a do while loop which is a post-test loop. Therefore, the code contained within the loop is executed at least once before the condition is tested and whether the condition is fulfilled or not. The do while loop was used to allow the user to go back to previous step if he wants to change his choice.

   printf("\nPlease enter No. of tickets  : ");

Displays message for the user to enter No. of tickets.

   scanf("%d", &quan);

Takes from the user the quantity of tickets.

    printf(" Quantity = '%d'  \n", abs(quan));

abs() is a function contained within the <stdlib.h> library, which makes the value of whole integer number absolute I.e., not negative. Since the variable ‘quan’ which represents the quantity taken from the user should always be whole number.

Total\_price = price \* quan;

This operation assigns ‘Total-price’ to ‘price’ multiplied by ‘quan’, which multiplies the price of one ticket with the quantity the user entered to get the total price.

   printf("type 'confirm' or press '1' to continue, else enter any key: ");

Displays message asking the user whether he wants to continue then press ‘1’ or type ‘confirm’, otherwise, press any other key.

Note: this could have been done the opposite way around, i.e., press ‘2’ or type ‘back’ to go back, as was done in the main function. However, we liked to have a variety in the coding as this time the while condition will contain && rather than || as in the do-while of the main function.

   scanf("%s",  s4 );

This line takes from the user his choice whether he wants to go previous step or continue.

To continue a user should press ‘1’ or type ‘confirm’.

}while(strcmp(s4,s5) != 0 && strcmp(s4,s6) != 0 || quan>64);\

The condition of the loop is fulfilled in any one of following two conditions:

1. strcmp(s4,s5) != 0 && strcmp(s4,s6) != 0

if the user didn’t enter either ‘1’ nor typed ‘confirm’, then the loop will repeat itself since s4 didn’t equal neither s5 or s6, and the result of strcmp() didn’t equal 0.

1. quan>64)

If the user enters a No. greater than 64 in the ‘quan’ variable, the loop begins again.

**Important Note: we discovered that the program gives nonsense values if the user entered an alphabet instead of a No. in this variable. Therefore, we made 64 as the end limit for this variable this is due to the fact that the smallest letter value equals ‘65’ which represents an upper-case A. \**



    return fabs(Total\_price);

by the end of the Quantity function, the function returns the value of ‘Total\_price’ variable to the main function which is a float value.

Important Note: this variable is of data type float, to make it absolute since the price should always be positive, the fabs() function was used as the abs() function only works for the integer variables.   
The fabs() function is declared in the <math.h> library.

}

This ends the Quantity function.

float monchang(float Total\_price,  char colour[10])                /\*\*\*\*\*\*Money Change function\*\*\*\*\*\*/

Return data-type – function name – (argument or input type)

This is a new function called monchange, that takes a float argument ‘Total\_price’, and ‘colour’ of data type character array of size [10] from the main function. Consequently, this function returns float value.

The purpose of this function is to calculate the change in money if the user paid higher than what the ticket costs.

{

This curled bracket starts the new function ‘monchang’.

      float money\_entered1, change, tot\_mon\_payed=0, money\_entered;

Declares the following variables as float since they hold money that may have decimal places.

     printf("\nplease enter the total price = %2.2f EGP : ",Total\_price);

Displays to the user the total amount of money he has to pay.

   scanf(" %f", & money\_entered1);

Takes from the user the amount of money he will pay.

  if (fabs(money\_entered1) >= Total\_price){

An If statement that is executed if the condition is fulfilled. The condition of its fulfillment is that the money entered by the user should be greater than or equal than the total price he has to pay.

Important Note: since this variable is of data type float, to make it absolute since the total price should always be positive, the fabs() function was used as the abs() function only works for the integer variables.   
The fabs() function is declared in the <math.h> library.

 change = fabs(money\_entered1) - Total\_price;

If the condition is true, this operation will be done to calculate the change which will subtract the total price from the money entered by the user.

The change should always be positive. Similarly, since it is of float type we use fabs() to make it absolute.

 printf("change = %2.2f \nTicket Color '%s'\n\nThank you for using Cairo's Green Metro Line", change, colour);

A message is displayed showing the change, giving the user his appropriate ticket (purple, green or blue), and thanks him for using the metro.

  }

Ends the if statement

  else

If the previous If statement was false, the program excutes the following algorithms or steps.

That is, if the user entered amount of money less than the total required money, the function proceeds to this path.

{

This curly bracket begins the else statement.

     change = fabs(money\_entered1) - Total\_price;

Operation that subtracts total price from the money entered firstly from the user and assigns it to change.

     tot\_mon\_payed = fabs(money\_entered1);

Assigns the money entered firstly from the previous step to the total money payed.

  do{

Begins the do-while loop

   printf("insufficient money, please enter '%2.2f' EGP : ", fabs(change));

Prints to the user the remaining amount of money left for him to pay

   scanf("%f",& (money\_entered));

Takes from the user again the remaining amount of money still to be paid.

   tot\_mon\_payed =  tot\_mon\_payed + fabs(money\_entered);

Then equates the total money paid by the user by total money paid previously + the money inserted later on.

   printf("Total money payed: %2.2f \n", tot\_mon\_payed);

Displays to the user the total money he paid in the whole transaction process.

   change = tot\_mon\_payed - Total\_price;

Calculates the change if the user’s total money paid is greater than the required total price.

   }while(tot\_mon\_payed<Total\_price);

This repeats the previous algorithmic process if the total money paid by the user is still less than the total price that is required. On the other hand, it ends when the total money paid is equal to or greater than the total price.

printf(" change = %2.2f EGP \n\nTicket Color '%s'\n\nThank you for using Cairo's Green Metro Line", change, colour);

A message is displayed showing the change, giving the user his appropriate ticket (purple, green or blue), and thanks him for using the metro.

}

This curly bracket ends the previous else statement.

}

This curly bracket ends the function monchang, as it has finished its task.

# Detailed Flowchart

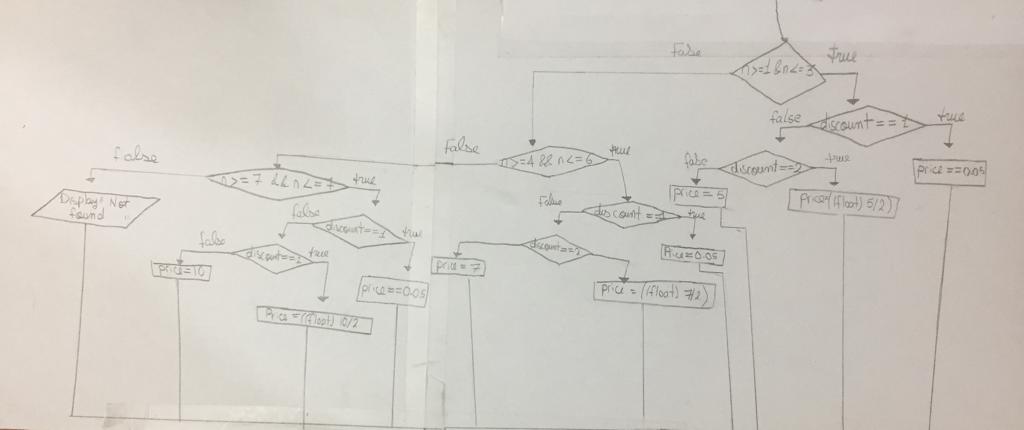
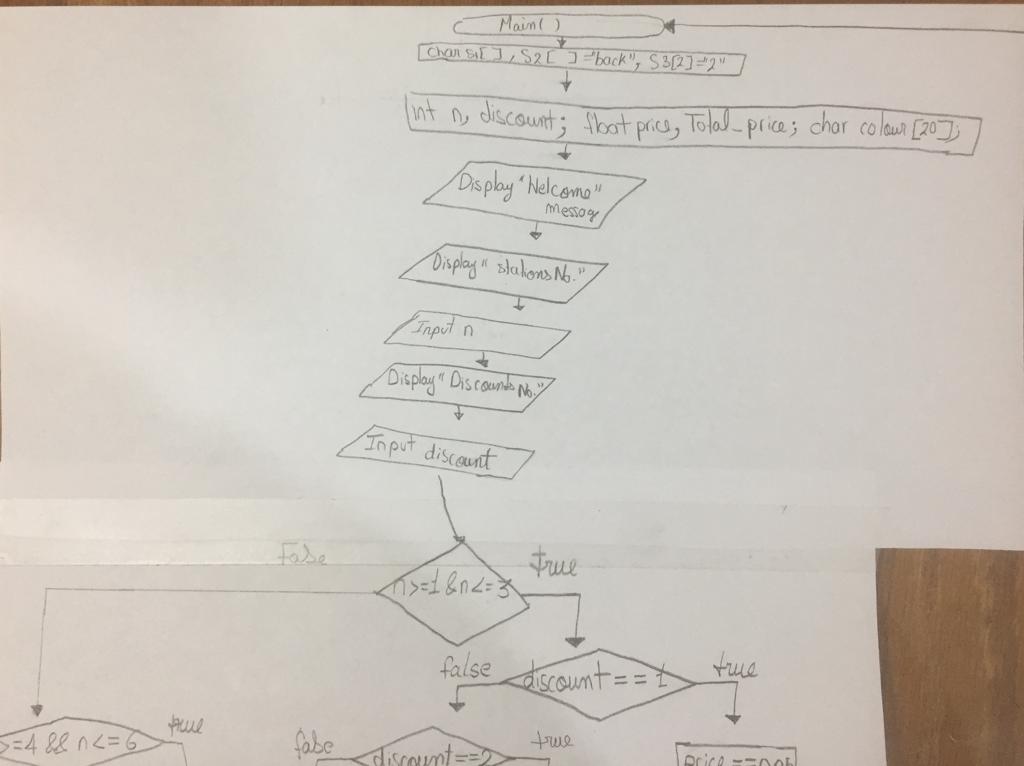
Main function flowchart:  


Figure 4 Main function flowchart

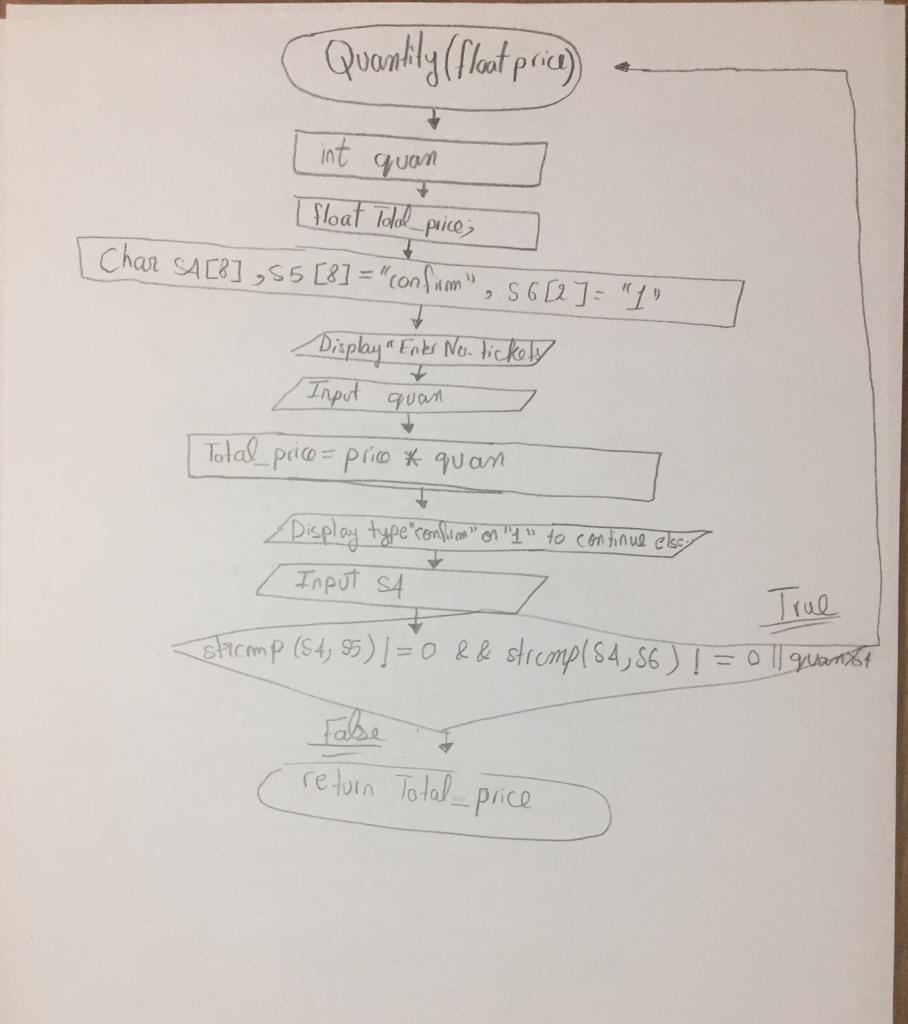
Quantity function flowchart:  


Figure 5 Quantity function flowchart

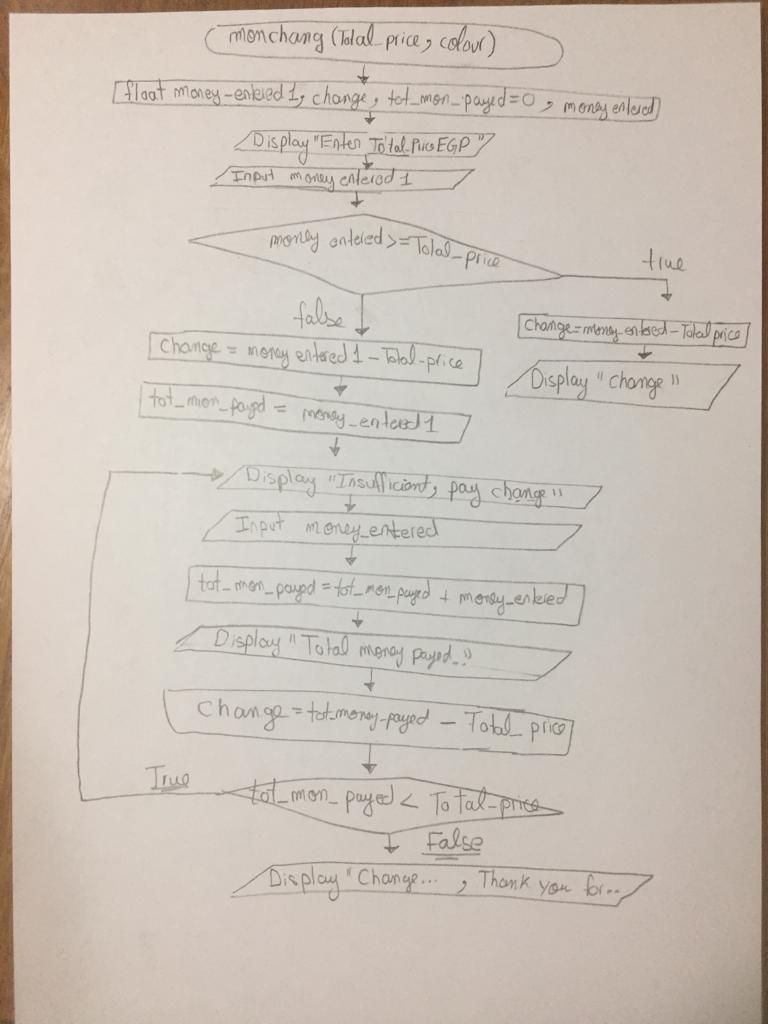
monchang function flowchart:  


Figure 6 monchang function flowchart

# Development

This program could be improved by adding features that provide the user with more benefits and improvements, such as the ability to reserve the seat number and category that they desire, as well as a function to compute the cost of these extra services.

# Conclusions

In conclusion, the vending industry has experienced substantial expansion throughout the years and will continue to do so in the future. The vending machine must now be improved to provide greater support and benefits to users. However, this program provides a solution to the issues raised while remaining true to the design objectives. The program was created with the use of a GDB compiler, which is useful for creating digital systems using a high-level description language. We achieve this without adding to the problem's complexity. Some problems occurred during the initial implementation; however, these issues were minor and simply served to mislead. Although there is still work to be done to improve this application, however, it can still stand on its own.

# Appendix

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Cairo's Green Metro Line 3

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

#include<math.h>

float Quantity (float price);                                     /\*\*\*\*\*\*Function should be decleared before they are used\*\*\*\*\*\*/

void monchang(float Total\_price, char colour[20]);              /\*\*\*\*\*\*Function should be decleared before they are used\*\*\*\*\*\*/

float main()

{char s1[5], s2[5] = "back", s3[2] = "2";

    int n, discount;

    float price, Total\_price;

    char colour[20];

do{                                                                        /\*\*\*\*\*\*Do While loop\*\*\*\*\*\*/

   printf("\nWelcome to Cairo's Green Metro Line! \n");

   printf("\nYou are in Adly Mansour station \n");

   printf("\n 1- Haykstep                   2- Omar Ibn El Khatab                     3- Qubaa \n 4- Hisham Barakat             5- El Nozha                               6- El Shams Club \n 7- Alf Masken                 8- Heliopolis \n");

   printf("\nEnter the equivalent No. of your final station : ");

   scanf("%d", &n);

   printf(" Your equivalent station's No. is : '%d' \n", n);

   printf("\npress '1' to enter POD(Determination) card, '2' for Senior Citizen Card, else enter any No. to continue : ");

   scanf("%d", &discount);

                                                                         /\*\*\*\*\*\*Nested If-Else-If\*\*\*\*\*\*/

   if(n>=1&&n<=3){                                                      /\*\*\*\*\*\*outer If\*\*\*\*\*\*/

       if(discount == 1)                                              /\*\*\*\*\*\*inner If\*\*\*\*\*\*/

       price=0.50;

       else if(discount == 2)                                     /\*\*\*\*\*\*inner else If\*\*\*\*\*\*/

       price=((float)5/2);                                       /\*\*\*\*\*\*float() was used to give a decimal result '2.5', otherwise it would have given '2' only\*\*\*\*\*\*/

       else                                                   /\*\*\*\*\*\*inner else\*\*\*\*\*\*/

       price = 5;

 char colour[20] = "Purple";

 }

                                                                  /\*\*\*\*\*\*Nested Else-If\*\*\*\*\*\*/

      else if(n>=4&&n<=6){                                       /\*\*\*\*\*\*outer Else-If\*\*\*\*\*\*/

          if(discount == 1)                                    /\*\*\*\*\*\*inner If\*\*\*\*\*\*/

       price=0.50;

       else if(discount == 2)                               /\*\*\*\*\*\*inner Else-If\*\*\*\*\*\*/

       price=((float)7/2);

        else                                             /\*\*\*\*\*\*inner Else\*\*\*\*\*\*/

       price = 7;

         char colour[20] = "Blue";

      }

                                                                 /\*\*\*\*\*\*Nested Else - If-Else-If\*\*\*\*\*\*/

   else if(n>=7&&n<=8){                                         /\*\*\*\*\*\*outer Else-If\*\*\*\*\*\*/

   if(discount == 2)                                          /\*\*\*\*\*\*inner If\*\*\*\*\*\*/

       price=((float)10/2);

       else if(discount == 1)                              /\*\*\*\*\*\*inner Else-If\*\*\*\*\*\*/

       price=0.50;

        else                                            /\*\*\*\*\*\*inner Else\*\*\*\*\*\*/

       price = 10;

         char colour[20] = "Green";

      }

   else                                               /\*\*\*\*\*\*outer Else\*\*\*\*\*\*/

   printf("your station is not found \n");

 printf(" Ticket price = %2.2f EGP \n", price);

   printf("type 'back' or press '2' to go back, else enter any key to continue: ");

   scanf("%s",  s1);

}while(strcmp(s1,s2) == 0 || strcmp(s1,s3) == 0 ||  n > 8 || n < 1 || discount > 64);         /\*\*\*\*\*\*Do While Condition\*\*\*\*\*\*//\*\*\*\*\*\*strcmp compares the strings together\*\*\*\*\*\*/

   Total\_price = Quantity(price);                                                            /\*\*\*\*\*\*فاطط ناطط to Quantity function\*\*\*\*\*\*\*/

   monchang(Total\_price, colour);                                                           /\*\*\*\*\*\*فاطط ناطط to Money Change function\*\*\*\*\*\*\*/

       return 0;

}

         float Quantity (float price)                                                 /\*\*\*\*\*\*Quantity function\*\*\*\*\*\*/

{     int quan;

float Total\_price;

char s4[8], s5[8] = "confirm", s6[2] = "1";

 do{                                                                               /\*\*\*\*\*\*Do While Loop\*\*\*\*\*\*/

   printf("\nPlease enter No. of tickets  : ");

   scanf("%d", &quan);

    printf(" Quantity = '%d'  \n", abs(quan));                              /\*\*\*\*\*\*abs() returns the absolute value of integer\*\*\*\*\*\*/

   Total\_price = price \* quan;

   printf("type 'confirm' or press '1' to continue, else enter any key: ");

   scanf("%s",  s4 );

}

while(strcmp(s4,s5) != 0 && strcmp(s4,s6) != 0 || quan>64);

    return fabs(Total\_price);                                       /\*\*\*\*\*\*fabs() returns the absolute value of float or double\*\*\*\*\*\*/

}

void monchang(float Total\_price,  char colour[10])                /\*\*\*\*\*\*Money Change function\*\*\*\*\*\*/

{

      float money\_entered1, change, tot\_mon\_payed=0, money\_entered;

     printf("\nplease enter the total price = %2.2f EGP : ",Total\_price);

   scanf(" %f", & money\_entered1);

  if (fabs(money\_entered1) >= Total\_price){

 change = fabs(money\_entered1) - Total\_price;

 printf("change = %2.2f \nTicket Color '%s'\n\nThank you for using Cairo's Green Metro Line", change, colour);

  }

  else{

     change = fabs(money\_entered1) - Total\_price;

     tot\_mon\_payed = fabs(money\_entered1);

  do{                                                           /\*\*\*\*\*\*Do While Loop contained within else statement\*\*\*\*\*\*/

   printf("insufficient money, please enter '%2.2f' EGP : ", fabs(change));

   scanf("%f",& (money\_entered));

   tot\_mon\_payed =  tot\_mon\_payed + fabs(money\_entered);

   printf("Total money payed: %2.2f \n", tot\_mon\_payed);

   change = tot\_mon\_payed - Total\_price;

   }while(tot\_mon\_payed<Total\_price);

printf(" change = %2.2f EGP \n\nTicket Color '%s'\n\nThank you for using Cairo's Green Metro Line", change, colour);

}

}

# Acknowledgment

We'd like to express our gratitude to module leader Dr. Nourhan Zayed for an interesting semester and all the time and effort she put into us, whether through lectures or individual feedbacks. Most importantly, for teaching us how think (BUE motto) logically and critically to find solutions to problems.

Lastly, but not least,

Thanks to Eng. Mostafa Abdelaziz for his assistance and providing us with experience and understanding of the course through the tutorials.

# References

* Chopra, R. (2017). *C Programming: A Self-Teaching Introduction*. Mercury Learning and Information.
* Gaddis, T. (2018). Starting Out with Programming Logic and Design. Pearson.
* Kelley, A., & Pohl, I. (1998). *A Book on C*. Addison-Wesley.