24/11/2016

NAME: RAJ G GAMI

AD102069

Data Structures and Algorithm

PROJECT – RESTAURANT MENU AND BILLING SYSTEM

**TABLE OF CONTENTS**

[CHAPTER 1 - INTRODUCTION 3](#_Toc467737816)

[OVERVIEW 3](#_Toc467737817)

[CHAPTER 2 – BUSINESS LOGIC 4](#_Toc467737818)

[STATEMENT OF PROBLEM 4](#_Toc467737819)

[STATEMENT OF OBJECTIVES 4](#_Toc467737820)

[LIMITATIONS 4](#_Toc467737821)

[CHAPTER 3 – PLANNING AND ALGORITHM 5](#_Toc467737822)

[LIST OF VARIABLES 5](#_Toc467737823)

[LIST OF FUNCTIONS 7](#_Toc467737824)

[ALGORITHM FLOWCHART. 8](#_Toc467737825)

[CHAPTER 4 – DATA ANALYSIS AND INTEPRETATION 12](#_Toc467737826)

[CHAPTER 5 – CONCLUSIONS. 12](#_Toc467737827)

# CHAPTER 1 - INTRODUCTION

# OVERVIEW

Conventional methods of serving at restaurants such as employing large number of people for tasks such as order taking is one of the major drawbacks of various restaurant outlets and business. These businesses try to minimize their costs to a level possible in order to earn a profit, however due to increasing demand of the customer satisfaction and personal contact, competition has risen. This competition involved various restaurant and catering businesses striving to provide best customer satisfaction and quality personal assistance. This competition has led to increased costs of the businesses in this sector of the industry. This wipes out small scale business that strive for survival in high competition as they are not able to withstand high costs and drown into losses. Further the large scale business manage to withstand the high costs however, they do not manage to earn a profit, worth their effort.

In order to address this problem, the conventional method has to be put aside to allow effective and efficient solutions, provided by the technology we have today, to take charge. This project also involves building a program that is created in C language, which addresses the problem mentioned above. The program consists of the restaurant menu, and its various abilities from generating a user specific order, to generating a total amount payable by customer, keeping the process to be highly interactive hence allowing the customer to carry out this process by himself/herself.

# CHAPTER 2 – BUSINESS LOGIC

# STATEMENT OF PROBLEM

There is increase in demand for better customer satisfaction and personal attendance in restaurants. This has created a competition among various businesses in restaurant sector, increasing costs of business, which lead to closure of business for small scale restaurant owners, while the large scale restaurant owners hardly earn any profit.

# STATEMENT OF OBJECTIVES

To create a program that contains a menu of the restaurant from which the customer can select his/her meal.

To create a program that generates customer’s order before final checkout

To create a program that allows customer to carry out changes such as addition of new item or deletion of an existing item from his/her order.

To create a program that generates a total amount payable by customer at the end of procedure.

To create a program that is interactive, user friendly, and able to meet customers’ demands of satisfaction and personal attendance.

# LIMITATION

As this project leads to an evolution of operation routine for restaurant businesses, it still is a prototype hence contains some limitations. Among the minor limitations that would be developed in time, the major one is that the program is not complex enough to address the money input by the customer and output their change along with a token that a customer would use to obtain his/her meal when ready. Rather a customer would have to move towards a counter where he will be provided further assistance and he/she should pay at.

# CHAPTER 3 – PLANNING AND ALGORITHM

# LIST OF VARIABLES

There are generally two types of variables in a C program, namely:

1. Global Variables – Global variables are variables that are defined under main function.
2. Local Variable – Local variables are variable that are defined specifically under a particular function.

Generally this variables also contain a data type, which defines the type of data that will be assigned to the particular variable. For example, a variable such as number would be assigned a data type such as integer to allow only integer values to be assigned to the variable

Below is list of all the variables, including their data types and initial assignment that have been defined in the project. They will categorized by determining the function under which they were declared.

1. Main Function

* *float price;* //this refers to price of each item
* *char choice;* //this refer to a character that is assigned to open a sub-menu, among others, under the main menu.
* *int a[16];* //this is an array of 16 elements. This elements are the total cost of each item that the customer has selected. Each element is a multiplication result of its respective price and the chosen quantity.
* *int i* //this is variable that are used to carry out loops and other activities related to an array
* *sum=0;* //this refers to the total amount payable by the customer
* *char b[16][40] = { ""*

*"Grilled Chicken Sandwich",*

*"Cheese Burger",*

*"Veg Fajita",*

*"Bacon Ranch Salad",*

*"Buttermilk Crispy Chicken Salad",*

*"Side Salad",*

*"French Fries",*

*"Low Fat Strawberry Yoghurt",*

*"Fruit and Yoghurt Parfait",*

*"Chocolate Shake",*

*"Vanilla Shake",*

*"Strawberry Shake",*

*"Coke",*

*"Sprite",*

*"Fanta" };*

//this a 2-D array of the items offered at the restaurant.

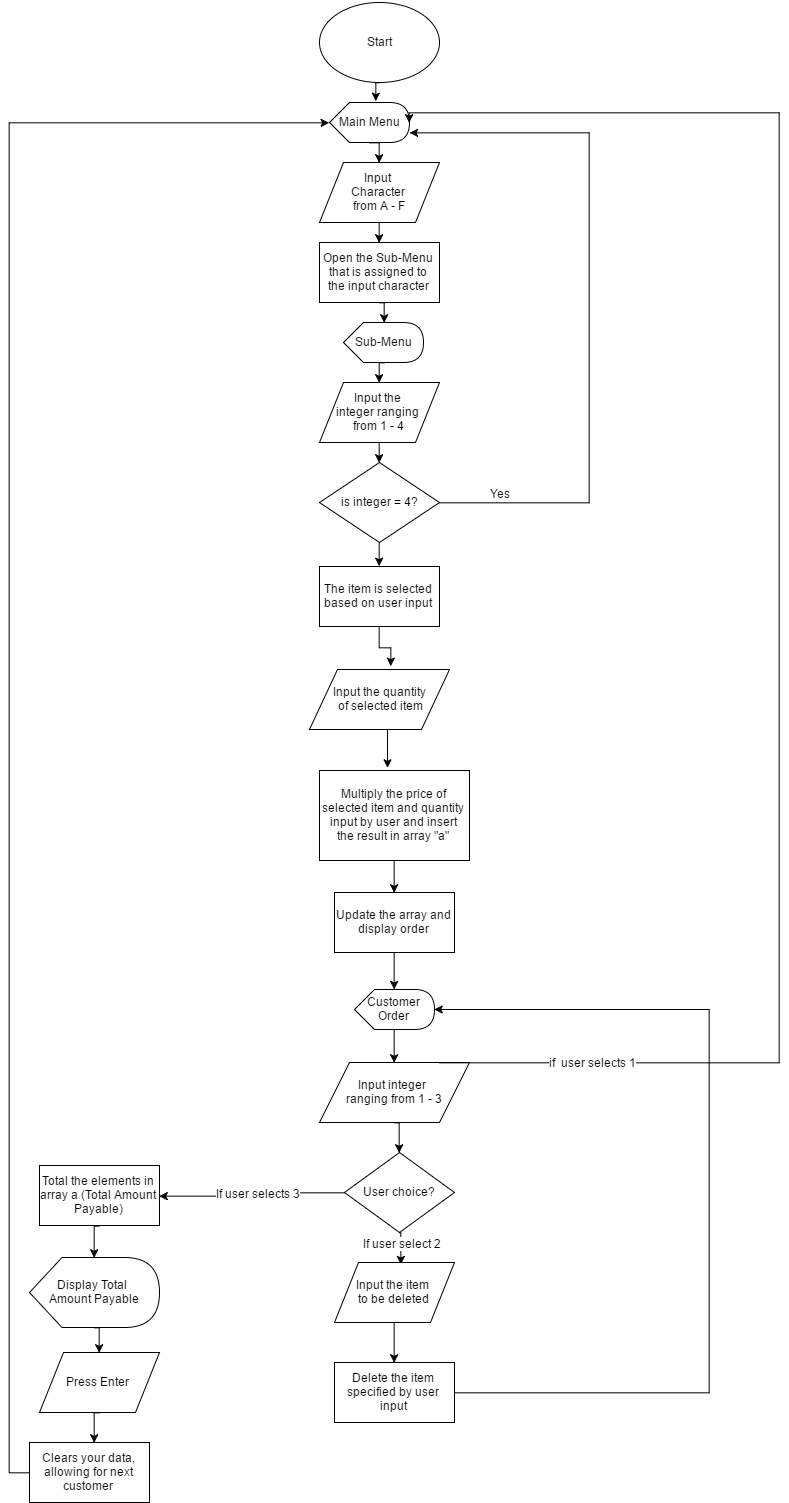
1. Sub-menus are triggered using a Nested If statement. Under each of this sub-menu, local variables are:
   * *int choice = 0;* // the variable int *choice* is assigned numbers that point to a particular meal in a sub-menu. An initial assignment “= 0” means that by when a user loads a sub-menu, choice 0 is loaded which is null.
   * *int quantity = 0;* //this variable defines the quantity of each item that the user inputs, after selecting the respective item.
2. Order Function
   * *int position;* //this variable is used in deleting item in customer order. The integer is input by customer and the respective item is deleted from order after pressing “Enter”.
   * *int again;* //this variable is used in decision making structure for user to decide whether to add another item, delete an item or checkout. The integer input is assigned to *int again;* and the instruction under respective integer is carried out.
   * *sum = 0;* initial amount payable is always equal to zero, until customer adds items to his/her order. This is not a declaration but an assignment.

# LIST OF FUNCTIONS

Along with variables, functions are also declared in a program. Below is a list of the functions in the program as well as a brief description of their use.

1. *void m\_m ( );* //this function exhibits the main menu and consists of Nested If statements that guides a user throughout to the sub-menu and allowing addition of meals to his/her order
2. *void complete( );* //this function is a courtesy function that thanks the customer and prompts to come again soon. Its primary use is to increase interactivity.
3. *void order ( );* //this function handles all the items that a customer has selected to purchase. They are added to an array and displayed here. This function allows further options to add more items to the order, delete an item from the order and finally checkout, which will display the amount payable.

# ALGORITHM FLOWCHART.

****

**CHAPTER 3 – STRUCTURE OF PROGRAM**

**OPERATIONS AND TOOLS USED**

* Data Types & Variables
* Header files
* Operators
* Decision Making
* Loops
* Functions
* Arrays
* Input/ Output Functions

1. Data Types & Variables

In this program, there are several uses of data types ranging from the basic data types consisting of integer types and floating-point types, to the type void. Examples are *void main, void m\_m, void order, void complete, float price, char choice,* etc.

1. Header files

Header files are files with extension “.h”. In this program, the header files are included by using a C preprocessing directive “#include”. It contains C function declarations and macro definitions to be shared between several source files. Examples are *#include <stdio.h>, #include <stdlib.h>, #include <ctype.h>, #include <windows.h>, #include <time.h>, #include <conio.h>*

1. Operators

Under the large category of operators, several types of operators were used to create the program. This types are:

1. Arithmetic Operators

This operators include:

\* - Multiplies both operands Example *a[1] = 500 \* quantity;*

++ - Increment operator decreases the integer value by one. Example: Refer to loops.

*Line 401 for( i = 1 ; i < 16 ; ++i ), Line 416 for (i=1; i<16; ++i).*

1. Relational Operators

This include:

= = - Checks if value of two operands are equal or not. If yes, then the condition becomes true.

# CHAPTER 4 – DATA ANALYSIS AND INTEPRETATION

**Comparison between the conventional methods and the float house**

**DATA ANALYSIS AND INTERPRETATION**

As indicated in the chart above:

* The number of disease outbreak reduces since human waste disposal can be controlled.
* Number of deaths due to flash floods reduces drastically since families can protect themselves against floods.
* A family can withstand several days of floods if food is available

# CHAPTER 5 – CONCLUSIONS.

**BACK GROUND**

Several people lose their lives during flash floods. The government has always made efforts to rescue flood victims though sometimes it come too late.

The Kenya Red cross on its part has flown to areas not accessible by any other means to airlift victims to safer grounds as well as providing them with the necessary amenities.

This however be improved by using the float houses so that by the time rescue operations reach on site, every other person is safe.

**PROBLEMS WERE TO BE SOLVED**

Our project was addressing:

* Disease outbreaks
* Poor sanitation during floods
* Loss of lives among others

**ACHIEVEMENT OF OBJECTIVES**

As indicated in the chart:

* Lives can now be saved during flash floods
* Disease outbreaks can be minimized
* Sanitation can be improved
* The government can save millions

**PARAMETERS/ ADJUSTMENTS**

* Density of the float house
* Weight of the float house
* Volume of the float house

|  |  |  |
| --- | --- | --- |
| **Parameter** | **Effect of change/ Alteration** | |
| Density of the float house | Increase | As the value of density of object approaches the value of density of water, buoyancy decreases and house will sink |
| Decrease | Increase in buoyancy and house will float |
| Weight of float house | Increase | As value of weight of object increases, buoyancy decreases and house will sink |
| Decrease | Increase in buoyancy and house will float |
| Volume of float house | Increase | As the value of volume of object increases, buoyancy decreases and house will sink |
| Less | Increase in buoyancy and house will float |

Figure 2 – Parameters

1. **EMERGING ISSUES**

In the year 2012, Kenya experienced some of the worst flash floods. Several people lost their lives despite the government and Red Cross efforts to mitigate the situation. If all families from the flood prone areas have float houses, losing lives during floods will be history.

1. **RECOMMENDATIONS**

That every family from flood prone areas build a flash flood float house and keep it safely for any eventuality.

That besides flood mitigation measures being put in place through the Ministry of Special Programs, the government should also build standby float houses and keep them in camps within flood prone areas to help those who can’t afford to build such houses.

Float shops can also be set up.

**CONCLUSION**

The country still lags behind in the improvement of lives of all Kenyans. Saving every live therefore on its own means stability.

This will go a long way to sustain the economic growth since apart from farming; life will go on as usual.

Disaster management skills will also improve.

**REFERENCES**

* Mango Thomas, (2003). The Issue of Floods in Budalang’i division. General Printers, Kenya
* Dunnes Thomas, (1974). Suspended sediment data for the rivers of Kenya, university press, University of Washington.

Fratkin, E. (1997). Governance and flood mathematics, publisher place, Atlanta.