

**Assignment Cover Sheet**

|  |  |  |
| --- | --- | --- |
| **Qualification** | | **Module Number and Title** |
| HND in Computing/ Software Engineering | | SEC4101 -  Fundamentals in Computer Programming |
| **Student Name & No.** | | **Assessor** |
| Widana Gamage Avindi Premaratne – ST/HNDCOM/19/15 | | Mr. Deloosha Abeyasooriya |
| **Hand out date** | | **Submission Date** |
| 23rd September 2016 | | 21th October 2016 |
| **Assessment type** | **Duration/Length of**  **Assessment Type** | **Weighting of Assessment** |
| Coursework  Practical | Report – 3,000 words  Software Demonstration | 50%  40% |

|  |  |  |
| --- | --- | --- |
| **Learner declaration** | | |
|  | | |
| I, Widana Gamage Avindi Premaratne – ST/HNDCOM/19/15.  acknowledged. | <name of the student and registration number> | , |
| certify that the work submitted for this assignment is my own and research sources are fully |

|  |  |  |  |
| --- | --- | --- | --- |
| **Marks Awarded** |  | | |
| First assessor |  |  | |
| IV marks |  |  | |
| Agreed grade |  |  | |
| Signature of the assessor |  | Date |  |

**FEEDBACK FORM INTERNATIONAL COLLEGE OF BUSINESS & TECHNOLOGY**

**Module :** SEC4101

**Student :** Widana Gamage Avindi Premaratne

**Assessor :** Mr.Deloosha Abeyasooriya

**Assignment :** FCP

**Strong features of your work:**

**Areas for improvement:**

**Marks Awarded:**

**Course Work and Practical Assessment**

This assignment is worth 90% of the overall assessment for this module.

**Learning outcomes covered**

LO2. Design a basic structured computer program

LO3. Implement a modularized computer programme for a prepared design

LO4. Perform software testing and documentation

**Scenario and the Tasks**

Puff and Fresh is a fast growing local restaurant which offers only breakfast items. Currently they are maintaining orders and print the bill manually with the use of a cash book. They have taken a step forward to provide a valuable service to its customers by automating their breakfast billing procedure. Their main objective is to provide faster and reliable service by providing correct information resource, without tiring the customers when such service is required. Program can use an appropriate data storage mechanism when storing several inputs by users. **The program should do the following:**

* Show the customer the different breakfast items offered by the restaurant.
* Allow the customer to select more than one item from the menu.
* Calculate and print the bill.

Assume that the restaurant offers the following breakfast items (The price of each item is shown to the right of the item):

**===================================**

**\*\*\*Welcome to Puff and Fresh\*\*\***

**\*\*\*Breakfast Billing System\*\*\***

**===================================**

**Item No Menu Item Price**

* 1. String Hopers Rs.65.00
  2. Egg Sandwich Rs.45.00
  3. Thosai with Chutney Rs.28.50
  4. Parata Rs.20.50
  5. Sausages Bun Rs.40.00
  6. Fish Roll Rs.35.00
  7. Plain Tea Rs.10.00
  8. Coffee Rs.15.00
  9. Tea Rs.20.00
  10. Fruit Juice Rs.22.50

**Your program must contain at least the following functions:**

1. Function **getData**: This function loads the data into the array **menuList.**
2. Function **showMenu**: This function shows the different items offered by the restaurant and tells the user how to select the items
3. Function **printCheck**: This function calculates and prints the check. (Note that the billing amount should include a 5% tax.)

Note: Define a struct, **menuItemType**, with three components :itemNo,menuItem,price and use an array ,**menuList** , of the **struct menuItemType**

**Assumptions to be made:**

You may assume that the customer can select multiple items of a particular type.

Any other assumptions according to your system requirements can be made. (Should clearly stated)

**A sample output is**

**===================================**

**\*\*\*Welcome to Puff and Fresh\*\*\***

**\*\*\*Breakfast Billing System\*\*\***

**===================================**

**Amount Menu Item Price**

2 Egg Sandwich Rs.90.00

1 Thosai with Chutney Rs.28.50

1 Plain Tea Rs.10.00

**Tax Rs.6.425**

**Amount Due Rs.134.925**

Carefully investigate the given scenario and provide the proposed solution.

**TASKS**

**Task 01 - System Design (LO2) -Report**

Identify the requirements of the system to be implemented and **produce detailed requirements specification. Design flow charts** and **write pseudo codes** according to the

SRS. Use appropriate modularization to reduce the complexity of the design*. (20 marks)*

**Task 02 – System Implementation (LO3) - Practical submission and Demonstration** Implement a functional C++ program to meet the requirements given in the specification, by following the design created above. *(40 marks)*

**Task 03 - System Testing and Documentation (LO4) - Report**

Produce proper testing documentation including test plan, test cases and test results. Proper **standard documentation** need to be followed throughout the report and language need to be

used accordingly. ***(30 marks)***

**Submission Guidelines**

* Submission format Report
* Paper Size: A4
* Words: 3000 words
* Printing Margins: LHS; RHS: 1 Inch
* Header and Footer: 1 Inch
* Basic Font Size: 12
* Line Spacing: 1.5
* Font Style: Times New Roman
* **Referencing should be done strictly using Harvard system**

**Assessment Criteria**

**Task 01 -** System Design **(LO2)**

|  |  |  |
| --- | --- | --- |
| **This submission will be assessed as follows**  **Criteria** | **Total marks**  **Allocated** | **Marks obtained by the student for the**  **answer provided** |
| **Out of 20** |
| **Excellent** Design   * Excellent SRS given in detail * Highly **detailed** diagram * **Use of modularization** concepts clearly visible * **Excellent use of symbols** * **Clarity and Reduce complexity** of the design * Backed by relevant assumptions | 14 - 20 |  |
| **Good** Design   * Detail **SRS** including functional and nonfunctional requirements, data and file structure requirements * **Flow charts** following standard notations in flow charting **and pseudo codes** using proper structured English * **Accurate use** of selection repetition structures * **Logical** and continuous flow of instructions along the design | 12 -14 |  |
| **Satisfactory** Design   * Basic **SRS**  including functional requirements * **Clear identification** and **application** of symbols in flow charts * Average level design diagrams given | 8 -12 |  |
| **Poor** Design   * Evidence of lack of understanding systems requirement specification * Poor use of design tools and symbols * Design diagrams with invalid flows, incomplete diagrams with logical errors | 0-8 |  |

**Task 02** – System Implementation (LO3)

|  |  |  |
| --- | --- | --- |
| **This submission will be assessed as follows**  **Criteria** | **Total marks**  **Allocated** | **Marks obtained by**  **the student for the answer provided** |
| **Out of 40** |
| **Excellent** implementation   * Excellent use of **control structures** with improved coding efficiency * Use **file handling** techniques for storage and backup requirements * **Excellent Modularization** with **effective data passing** between developed modules. * Appropriate **arrays, structs(records)** used * Easy navigation between modules, **accuracy, creativity and completeness** of the system | 28-40 |  |
| **Good** implementation   * Use of **Comments** to improve code readability * good use of **control structures** with proper understanding * **Modularize** according to the given design. * Use of input validations, onscreen help options and User friendliness of the system | 24-28 |  |
| **Satisfactory** implementation   * **Operational system** according to the   requirements of the scenario   * Average use of **data types** and **operators** * Average use of **control structures** (selection and repetition) | 16-24 |  |
| **Poor** implementation   * Poor implementation with syntax errors * Lack of knowledge of the language basics used * Cannot fulfill basic system requirements | 0-16 |  |

**Task 03** - System Testing and Documentation (LO4)

|  |  |  |
| --- | --- | --- |
| **This submission will be assessed as follows**  **Criteria** | **Total marks**  **Allocated** | **Marks obtained by the student for the**  **answer provided** |
| **Out of 30** |
| **Excellent** Documentation   * Excellent Test documentation with detail **test plan** and **test cases** * **Testing conclusion** with critical review and future recommendations given * Appropriate use of language * Standard report format * Descriptive **user Manual** with screen shots of relevant interfaces * Proper use of Referencing | 21-30 |  |
| **Good** Documentation   * Detailed **Test Plan** * Appropriate **Test Cases** * Acceptance test with **User feedback** and test **conclusion** | 18-21 |  |
| **Satisfactory** Documentation   * Basic **Test Plan** * Average **Test Cases** * Average documentation | 12-18 |  |
| **Poor** Documentation   * Lack of test plan, poor test cases * No proper evidence of testing * Poor report formatting | 0-12 |  |

|  |  |  |
| --- | --- | --- |
| **Total Marks** | **90** |  |

# Acknowledgement

First of all I would like to express my deep gratitude to my parents for advising me to join to the Dual HND in Software Engineering + Computing and Systems Development offered by the ICBT Southern Campus Matara. They have always being guiding me to the correct path and success in my life to achieve my goal.

I also thank to Mr. Deloosha Abeysooriya, our Lecturer, for giving me guidance to prepare this assignment. His lectures have been very educative and interesting. The production of this assignment has given me opportunities to expand my knowledge of the world of computer systems as well as improve my skills in research, designing and communication. It would not have been possible for me to produce this assignment in the professional way in which I have done it without his guidance and advice.

In addition, I thank to the ICBT Campus for offering this course in Matara in the first place, which has enabled me as well as many others living in the Southern Province to join it.

Assignments based on computer-related subjects can never be prepared based on original information. Hence, I relied a lot on researching through the internet. As such, I acknowledge with thanks all the authors, experts and other professionals from whose publications I have used for referencing information for this assignment.

# Executive Summary

This assignment is about creating a breakfast billing system to Puff and Fresh Restaurant. Previously they have been a manual billing system and they request us to create a billing system to provide fast and reliable service to customers by providing correct information resource, without tiring the customers.

In the first task the learner have to identify the system requirements of this restaurant and produce a detailed System Requirement Specification. Also the learner should design the flowcharts and write pseudo codes according to the SRS and should explain the process of it.

In the second task the learner has to implement a functional C++ program for it to meet the requirements given in the specification according to the flow chart design created. The codes have to be attached and well explained how the program functions. The program should include control structures, arrays etc…

Finally in the task 3 the user have to provide proper testing documentation which includes test plan, user manual and user feedbacks and conclusions too.

# Table of Contents

Contents

[Acknowledgement i](#_Toc464830754)

[Executive Summary ii](#_Toc464830755)

[Table of Contents iii](#_Toc464830756)

[List of Figures iv](#_Toc464830757)

[1. Introduction 1](#_Toc464830758)

[What is a Programming Language? 1](#_Toc464830759)

[2. Task 1 – System Design 3](#_Toc464830760)

[2.1 Requirement Specification of the program. 3](#_Toc464830763)

[2.2 Flowcharts for the breakfast billing system 5](#_Toc464830764)

[3. Task 2 – System Implementation 21](#_Toc464830765)

[3.1 Codes for the Puff and Fresh Breakfast Billing System 35](#_Toc464830767)

[3.2 Modularization 57](#_Toc464830768)

[4. Task 3 – System Testing 58](#_Toc464830769)

[4.1 What is System Testing? 58](#_Toc464830771)

[4.2 Test Case 59](#_Toc464830772)

[4.3 User Manual 61](#_Toc464830773)

[4.5 User Feedbacks 69](#_Toc464830780)

[4.6 Testing Conclusion 76](#_Toc464830781)

[4.7 Recommendation 77](#_Toc464830782)

[4.8 References 78](#_Toc464830783)

# List of Figures

[Figure 1.0 - Flowchart of Log in to the system 5](file:///I:\Introduction%20C++%20-%20Copy.docx#_Toc464762909)

Figure 1.2 - Flowchart of displaying menu, order menu, search and exit……………………..7

Figure 1.3 – Flowchart of showing how to make an order, display and print the bill………...9

Figure 1.3 a – Flowchart of showing how to make an order, display and print the bill…… 10

Figure 1.3 b – Flowchart of showing how to make an order, display and print the bill…….11

Figure 1.3 c – Flowchart of showing how to make an order, display and print the bill……12

Figure 1.4 - Flowchart of Search menu ……………………………………………………...12

Figure 1.5 - Flowchart of Exit menu…………………………………………………………14

Figure 2.0 a – Main menu……………………………………………………………………16

Figure 2.1 b – Main menu……………………………………………………………………17

Figure 2.2 c - Main menu…………………………………………………………………….18

Figure 2.3 – Login Menu……………………………………………………………………..19

Figure 2.4 – Menu List……………………………………………………………………….20

Figure 2.5 a – Order Menu…………………………………………………………………...21

Figure 2.5 b – Order Menu…………………………………………………………………..22

Figure 2.5 c – Order Menu…………………………………………………………………..23

Figure 2.5 d – Order Menu…………………………………………………………………..23

Figure 2.5 e – Order Menu…………………………………………………………………. 24

Figure 2.5 f – Order Menu…………………………………………………………………..24

Figure 2.5 g – Order Menu………………………………………………………………… 24

Figure 2.6 – Display and print bill…………………………………………………………..25

Figure 2.7 –Search Menu……………………………………………………………………26

Figure 2.7 a –Search Menu ………………………………………………………………...26

Figure 2.7 b – Search Menu ………………………………………………………………..27

Figure 2.7 c – Search Menu………………………………………………………………...27

Figure 2.7 b – Search Menu…………………………………………………………………28

Figure 2.7 b – Search Menu…………………………………………………………………28

Figure 2.8 – Exit Menu………………………………………………………………………29

Figure 3.0 – User Manual…………………………………………………………………….57

Figure 3.1 – User Manual…………………………………………………………………….57

Figure 3.2 – User Manual…………………………………………………………………….58

Figure 3.3 – User Manual…………………………………………………………………….58

Figure 3.4 – User Manual…………………………………………………………………….59

Figure 3.5 – User Manual…………………………………………………………………….59

Figure 3.6 – User Manual…………………………………………………………………….60

Figure 3.7 – User Manual…………………………………………………………………….60

Figure 3.8 – User Manual…………………………………………………………………….61

Figure 3.9 – User Manual…………………………………………………………………….61

Figure 3.10 – User Manual…………………………………………………………………. 62

Figure 3.11 – User Manual…………………………………………………………………. 62

Figure 3.12 – User Manual………………………………………………………………….63

Figure 3.13 – User Manual………………………………………………………………….64

Figure 3.14 – User Manual………………………………………………………………….64

Figure 3.15 – User Manual………………………………………………………………….64

# Introduction

## What is a Programming Language?

Programming Languages are languages used by computers to give commands to solve problems by using a language understood by the machine. Also we can use programming languages to create programs. Programming languages can be categorized into three categories.

1. Machine Language
2. Assembly Language
3. High Level Language

Machine Language is the only language that can be directly understood by the computer. This language consists of binary values 1 and 0. But this programming language is very hard to understand for humans to code them. Programs written in this language runs fast on computers, so this language is very close to the machines and its machine dependent.

Because machine language are hard to code, there came Assembly Level Languages which is also called Symbolic Language because it uses symbols in the program. This language is quite close to human language. Also it uses a translator name Assembler to translate the codes to machine language to the machine. But Assembly Language is machine dependent like machine languages.

After Assembly Languages there came High Level Languages into being. These languages are close to humans and based on human languages like English and there aren’t machine dependent. Because of these properties it is easy to write and understand codes. Also it is easy to debug, machine independent. But like Assembly Language, High Level Languages too need a translator. Compiler and Interpreter are some examples for High Level Language translators. Because of High level languages many programs and software are created to solve problems.

# Task 1 – System Design



## Requirement Specification of the program.

In requirement specification we are going to discuss about the description of the program the functional and non- functional requirements of the program and the inputs and outputs of the program.

So this program is an automated breakfast billing system created to Puff and Fresh Restaurant. Currently this restaurant is maintaining orders and print the bill manually with the use of a cash book. So they need to have a breakfast billing system to provide fast and reliable service by providing correct information resource. The requirements that they need to be included in this program are:

* Show the customer the different breakfast items offered by the restaurant
* Allow the customer to select more than one item from the menu
* Calculate and print the bill

So I have created a program for this billing system using C++ language, a high level programming language. So I’ll explain in every detail how the program functions.

First when the program runs, the user will see the login screen with username and password. So the user have to enter the required username and password. If the login details are correct the user can login into the system.

After the user logins to the system, he or she will see the menu list which displays the breakfast items offered by the restaurant. Then the user has to first input the number of items that he or she is going to order. Next, the user has to enter the item number of the relevant breakfast item that he’s going to order. After that he has to enter the quantity of that item. Then the user will see the item name, and the total price of it.

The final part is the printing of the bill. After the user has ordered the relevant items, the items will be displayed in the invoice. In there, the total amount will be calculated and the 5% tax will be included to the bill. Finally the bill can be printed and the user can exit the program.

In this program I have used the file stream to view the menu list, variable types like integer, string etc... , switch case to build up this program.

## Flowcharts for the breakfast billing system

Figure .0 - Flowchart of Log in to the system

Start

If username = admin AND password = 123

Input username and password

Yes

Enter into the system

No

Display “User Invalid”

The above figure is a flowchart of user login into the system. So in here, the user have to input the username and password as “Admin” and “123”. If the username and password is valid only, the user can enter into the system. Else, the system would display an error message “User Invalid” and again comes back to the same position till he enters correct username and password.

**Pseudo Code**

BEGIN

INPUT username and password

If username = admin AND password = 123 Then,

Display “User Valid”

Else

Display “User Invalid”

END

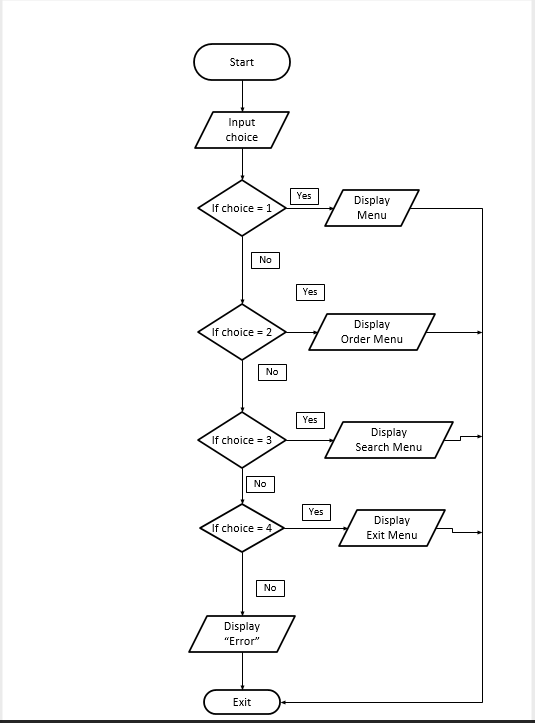


Figure 1.2 - Flowchart of displaying menu, order menu, search and exit

In this figure of the flowchart shows how the menu list is displayed to the user. First the system must get the user’s inputs. The user has to input a number to “choice” variable. If the user has input “1”, then the system has to display the menu list to the user. Else, if the user has input “2”, then the system has to display the order list to the user. Else, if the user has input “3”, then the system has to display the search menu. Else if the user has input “4”, then system has to display Exit menu. Else, the system has to display “Error” for wrong input and end it

**Pseudo Code**

BEGIN

INPUT choice

If choice = 1 Them,

Display Menu

Else If choice = 2 Then,

Display Order list

Else if choice = 3 Then,

Display Search

Else if choice = 4

Display “Error”

END

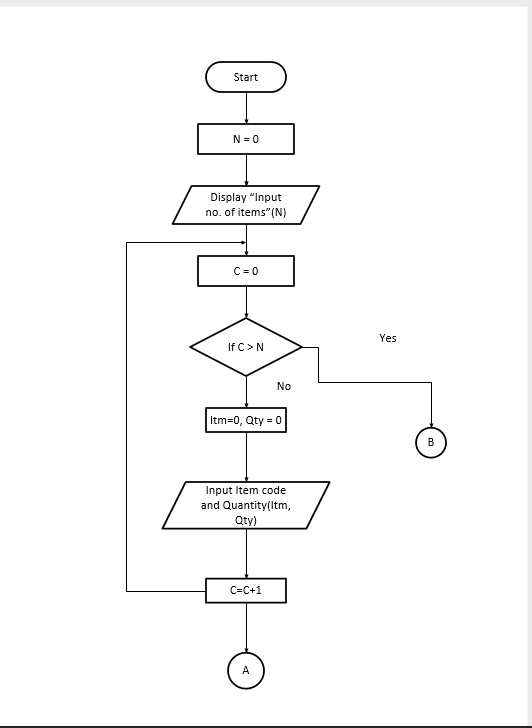


Figure 1.3 – Flowchart of showing how to make an order, display and print the bill

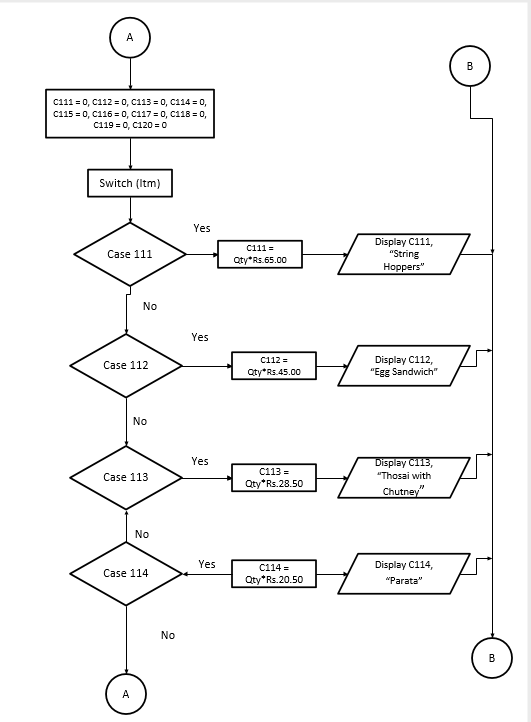
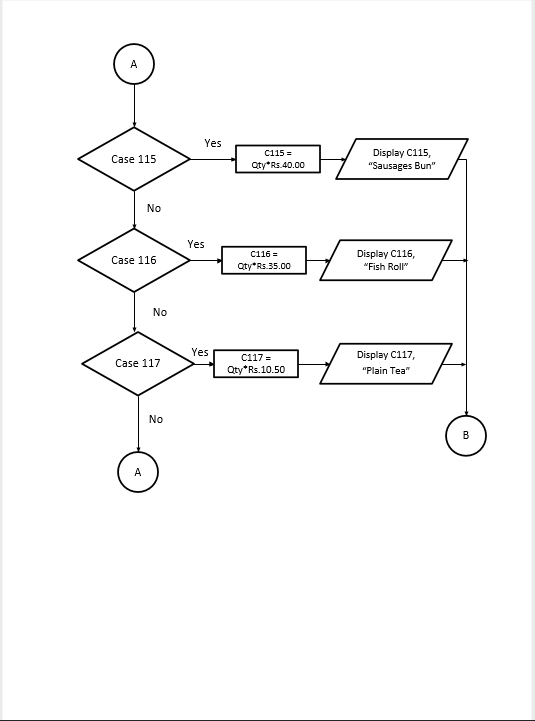
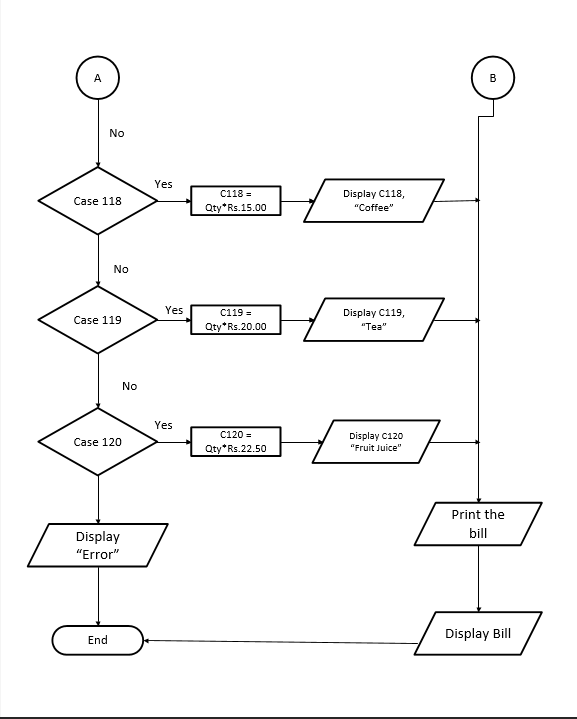


Figure 1.3 a - Flowchart of showing how to make an order, display and print the bill

Figure 1.3 b - Flowchart of showing how to make an order, display and print the bill

Figure 1.3 c - Flowchart of showing how to make an order, display and print the bill

This flowchart shows how the items are ordered and printed to the bill. In Figure 1.3 it shows the start of ordering items. As shown in Figure 1.3 first the user have to input the No.of items(N). Then I have used another variable named C for counter variable. I have used a for loop to repeat of entering Item code and quantity. The number of times the for loop will repeat is based on the number of items the user has given to input before.

In Figure 1.3 a, ten variables are declared to store the calculations of each menu item. After the user has ordered items, the switch case then checks the item code (Itm) and checks whether its equal to the case number. For example we order “111” item. Then the switch checks whether the item is equal to its’s case (case 111) . If it’s true it does the calculation and prints the bill and order. Else it would check the othr cases.

BEGIN

Itm = 0

INPUT No. of items(Itm)

For C=0 to Itm Step 1

I=0, Qty = 0

Display “Input Item code and quantity” (I, Qty)

C=C + 1

C111 = 0, C112 = 0, C113 = 0, C114 = 0, C115 = 0,

C116 = 0, C115 = 0, C118 = 0, C119 = 0, C120 = 0

Switch (Itm)

Case 111:

{

C111 = Qty\*Rs.65.00

Display “String Hoppers”, C111

}

break;

Case 112:

{

C112 = Qty\*Rs.45.00

Display “Egg Sandwich”, C112

}

break;

Case 113:

{

C113 = Qty\*Rs.28.50

Display “Thosai and Chutney”, C113

}

break;

Case 114:

{

C114 = Qty\*Rs.20.50

Display “Parata”, C114

}

break;

Case 115:

{

C115 = Qty\*Rs.40.00

Display “Sausage Bun”, C115

}

break;

Case 116:

{

C116 = Qty\*Rs.35.00

Display “Fish Roll”, C116

}

break;

Case 117:

{

C111 = Qty\*Rs.10.50.

Display “Plain Tea”, C112

}

break;

Case 118:

{

C118= Qty\*Rs.15.00

Display “Coffee”, C118

}

break;

Case 119:

{

C119= Qty\*Rs.20.00

Display “Tea”, C119

}

break;

Case 120:

{

C120= Qty\*Rs.22.50

Display “Coffee”, C120

}

break;

Display the order

Write to the bill

Display the bill

END

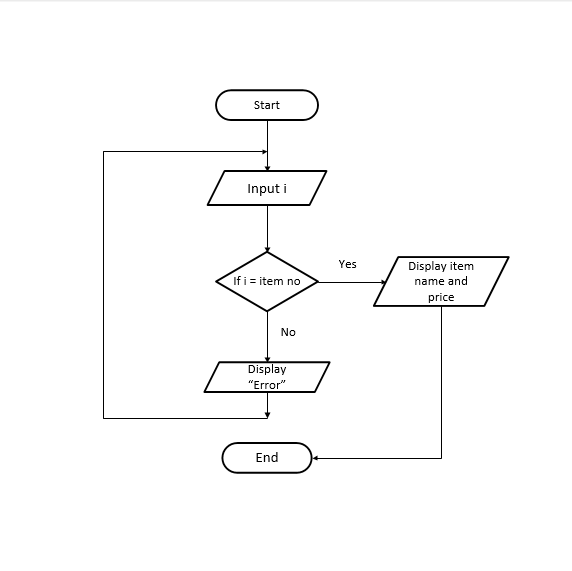


Figure 1.3 - Flowchart of Search menu

This figure of flowchart show how the search menu works in the system. After the user has selected the search option, he would get a message to input the item code to search. Then, the user has to input the item code. After that the system checks whether the number is equal to the item no. given in the program. If it’s true then system will display the item name and price. Else it will display error and comes back to the beginning of the search menu program.

**Pseudo Code**

BEGIN

INPUT i

If I = item no Then,

Display item name and price

Else,

Display “Error”

Loop

END

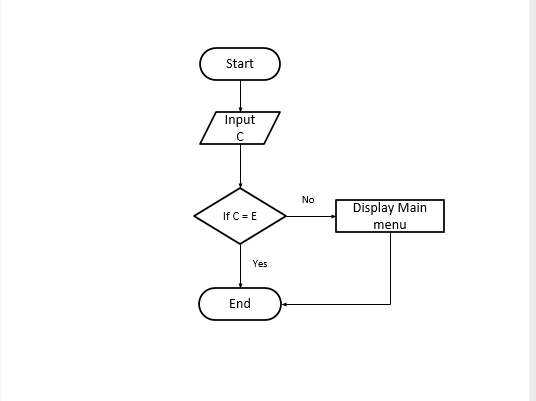


Figure 1.4 - Flowchart of Exit menu

In this flowchart it shows how the exit option works. After the user has selected the 4th option, then the system displays a message asking whether the user really needs to exit the program. If so, the user has to enter “E” to exit. If the user has enter this, then the program will end. Else it would come back to the beginning of the whole program it means to the main menu

**Pseudo Code**

BEGIN

INPUT C

If C = E Then,

EXIT the program

Else,

Display Main menu

END

# Task 2 – System Implementation



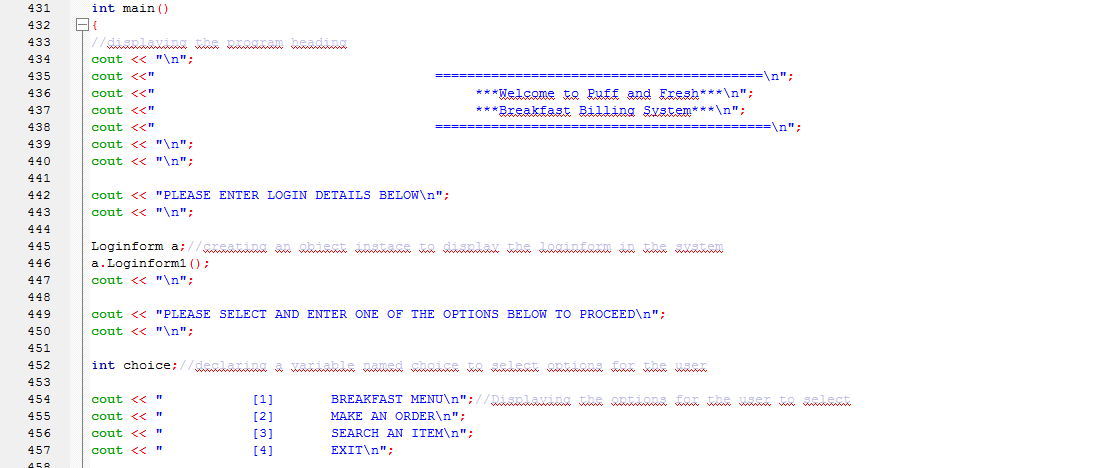
So in this section we have to develop the program according to the flowcharts drawn in Task 1. To develop this program I used Code:Blocks IDE using C++ Language. Let’s see how the system is developed.

Figure 2.0 a – Main menu

This is the main method of the Puff and Fresh program. So in this section first of all, I have created the heading of the program to display. Then after that the program displays the logii details message(“Please enter login details below”). Then to show the login menu I have used a function named “Loginform1” in the class named “Loginform” separately in the program. So to display that, I have created an object instance in the main to show the Loginform class and Loginform1 function. After creating object instance it displays the login form. After the user has input correct login details, then the user will see a navigation pane to select some options to proceed. The user will see the for options and the user ha to selec a one. I have declared a variable named ”choice” to assign the user’s input to select an option.

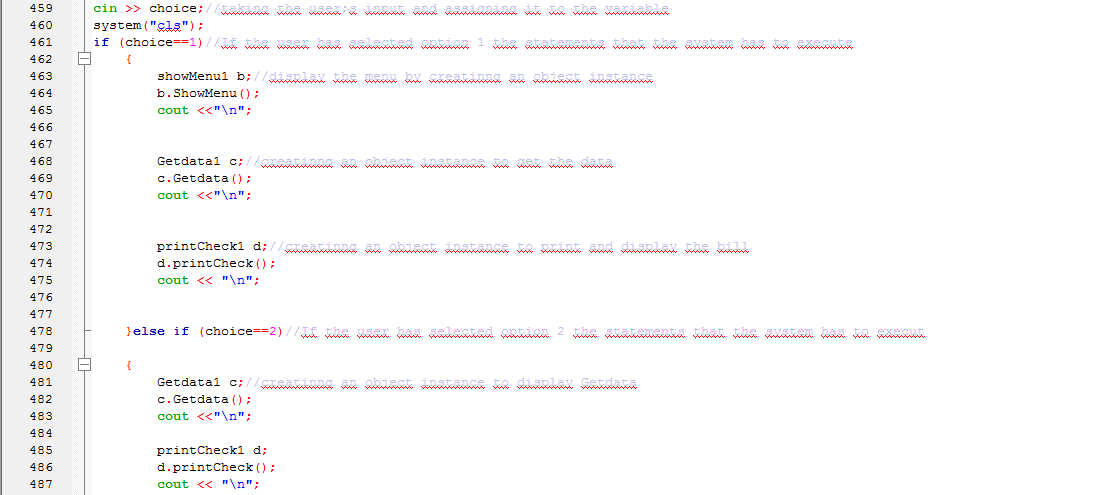


Figure 2.1 b – Main menu

After the user has input a number to select the option the program will run based on the user’s selected option to proceed. So I have used a control structure to run the program according to user’s choice. I n here I have used the IF function. So, if the user has input “1” as the choice, then the program will display the menu first. To display the menu like the **Loginform** I have used a function named **showMenu** which includes the menu list and a class named showMenu1 separately. Like the **Loginform** I created an object instance in the main to show the menu. After the user has input the relevant inputs, then program will run the **Getdata** function which includes order menu. In here too I have used a function and a class for **Getdata** and create an object instance to display it. Next the program will print and show the bill. For that task I have created **“printCheck”** function. Finally the program will end up.

If the user has input “2” as the option, then the program will run the **Getdata** and **printCheck** functions only. It means it will show the order menu and bill only.

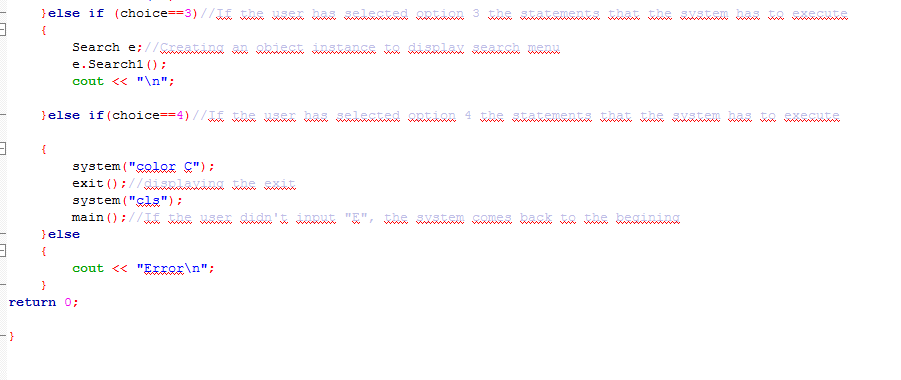


Figure 2.2 c - Main menu

Or if the user has select “3” as the option, the program will run the **Search1** function in the **Search** class which includes the searching of menu items in the menu list. Else if the user has input “4” as the option, it will show up the exit function which includes the exit menu. If the user inputs “E”, the program will close. Else, it will show up the main menu if the user gave any input.

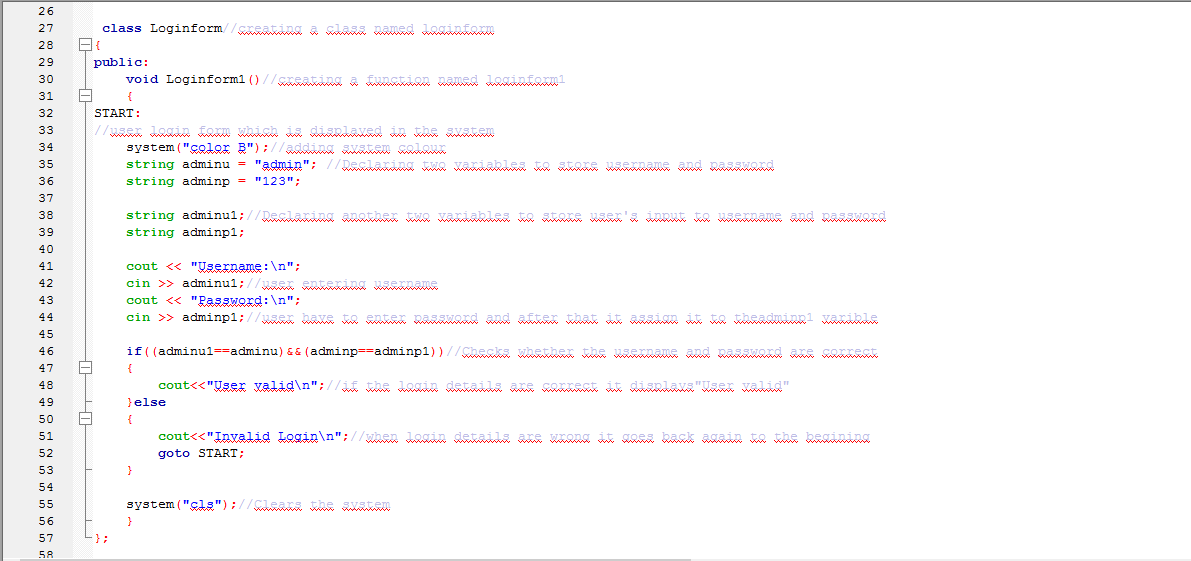


Figure 2.3 – Login Menu

This figure shows the **Loginform** class and function of the program. I have put the system colour as”Cyan” for the program. Then I have declared two string variables namely “adminu” and “adminp” to store the username and password of administrator. Then I have declared another two string variables namely “adminu1” and “adminp1” to assign user’s username and password inputs. After that the program will display to enter username and password. Then the two string variables will assign user’s inputs. I have used If condition to check whether the user has entered correct login details. If the user has entered correct login details the program will proceed to next level. Else it would display “User invalid” and again asks the user to input correct username and password.

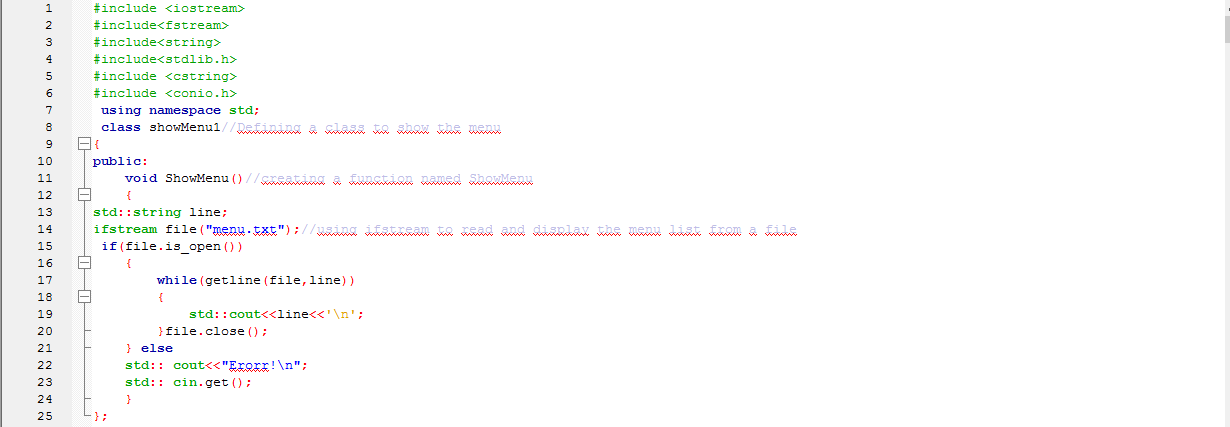


Figure 2.4 – Menu List

This figure shows the **showMenu** function and class which includes of displaying the menu list. First of all, I have named a class and function named showMenu1 and showMenu. I had already typed the menu list of the restaurant in text file and I need to open and display that in the program. So I used **Ifstream** file class to open and read the contents in that file and display it in the program. So in this **showMenu** function it includes the ifstream class of opening and reading the contents from “menu.txt” file.

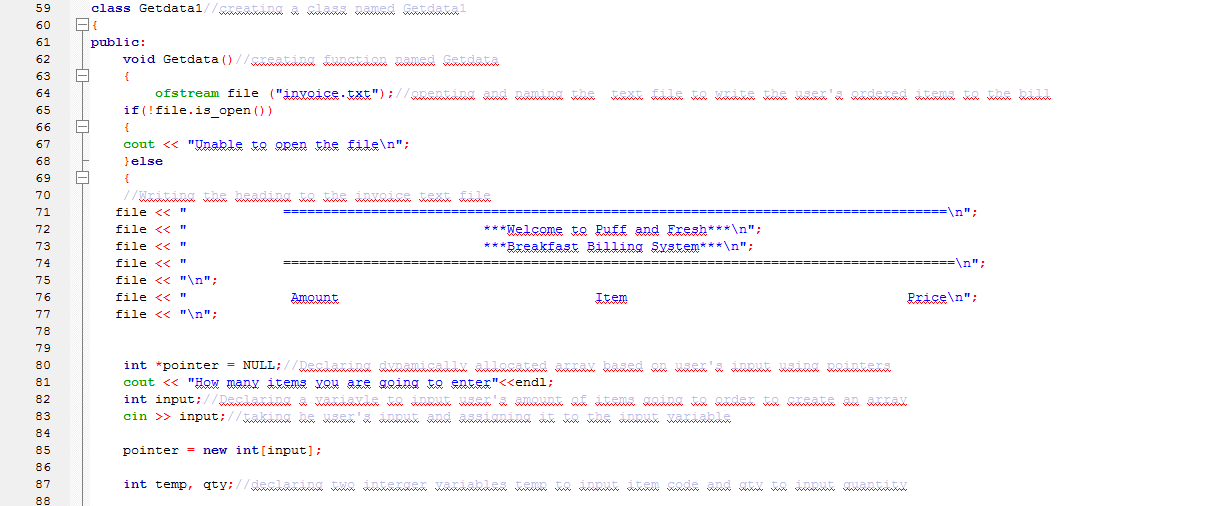


Figure 2.5 a – Order Menu

This figure shows the **getData** function and class. This section includes of ordering the menu items, calculating and writing to the bill. So first of all I have declared a class and a function namely getData1 and **getData**. Then I have used “Ofstream” file class to open a text file named”invoice.txt” and write the ordered items and calculations to it as the bill. If the file is available first it would write the things to the file. Else, it would display unable to open file. So in next I have used “file” to write all the items to the bill. I have written the heading of the bill to the text file now. Next is the ordering of items. I have dynamically allocated arrays using pointers based on the number of items that he user needs to order. I have declared an integer type variable to get the number of items that the user is going to order. Also I have declared another two integer type variables namely “temp” and “qty” to assign the item code that the user has input and the quantity.

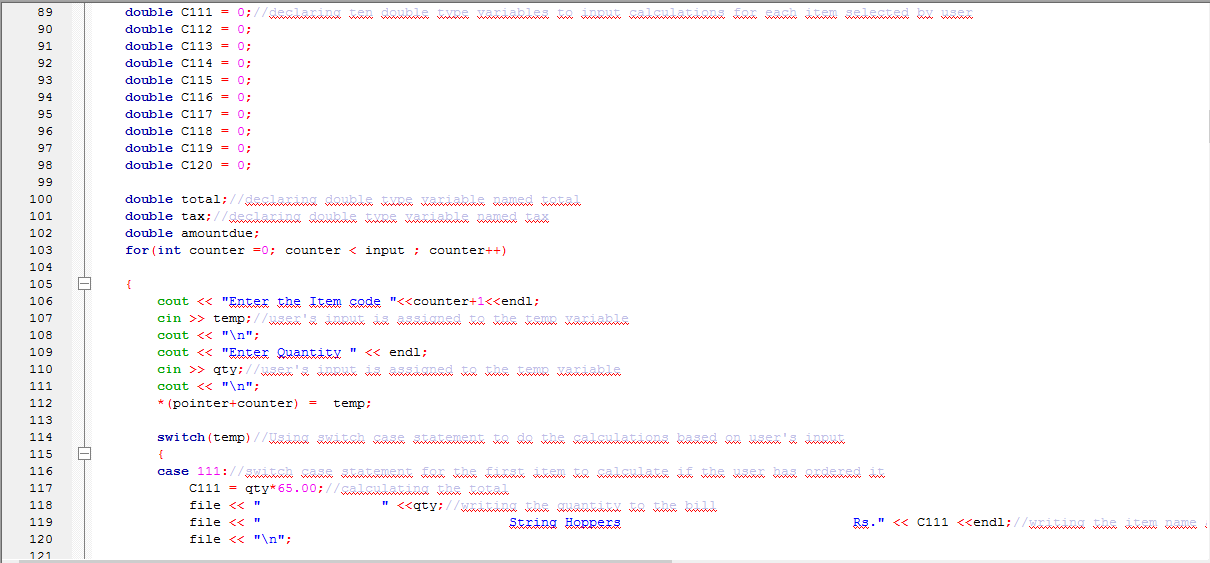


Figure 2.5 b – Order Menu

Then I have declared ten double type variables to store the calculations of each menu item of the restaurant and another three double type variables to input total, tax and amount due. Next the system will ask the item code that the user has ordered and the quantity of each item. So I have used a for loop to repeat the process to enter item code and quantity till the counter gets the value more than number of items ordered by the user. The “temp” variable stores the item code entered by the user. So I have used switch case statement here to do the calculations if the user has entered the relevant code. For example if the user has entered “112” as the code and quantity as “2”, the switch case will check the temp variable where the user’s input code is stored and executes the statement relevant to the item code “112” only. So in this case the system will calculate the amount relevant to “case 112” and display “Egg Sandwich Amount is Rs.90” and it will write to the bill. Likewise the switch case will execute the relevant statements based on the item code input by user. Below are the rest of the switch case statements.

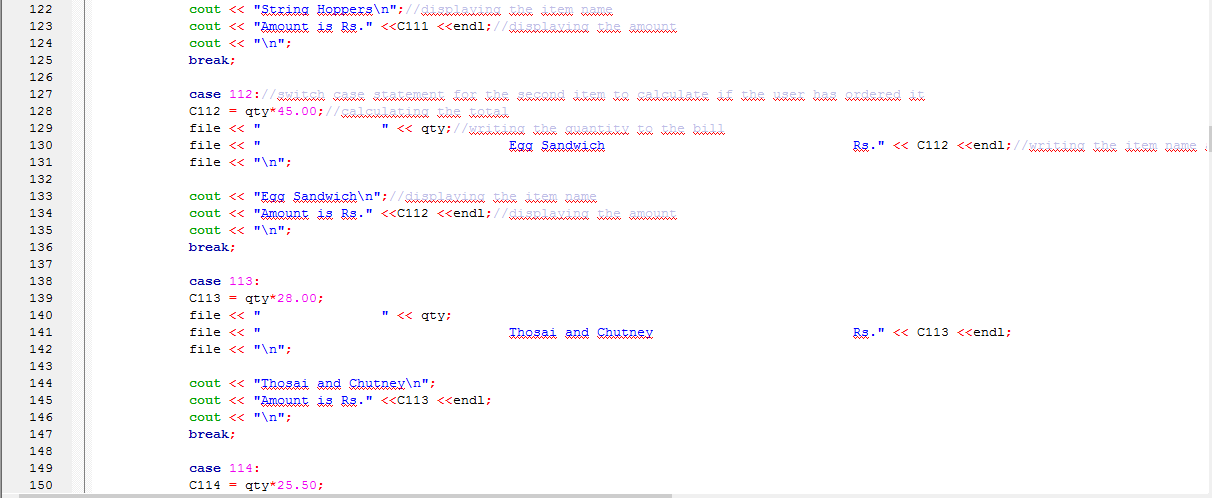


Figure 2.5 c – Order Menu

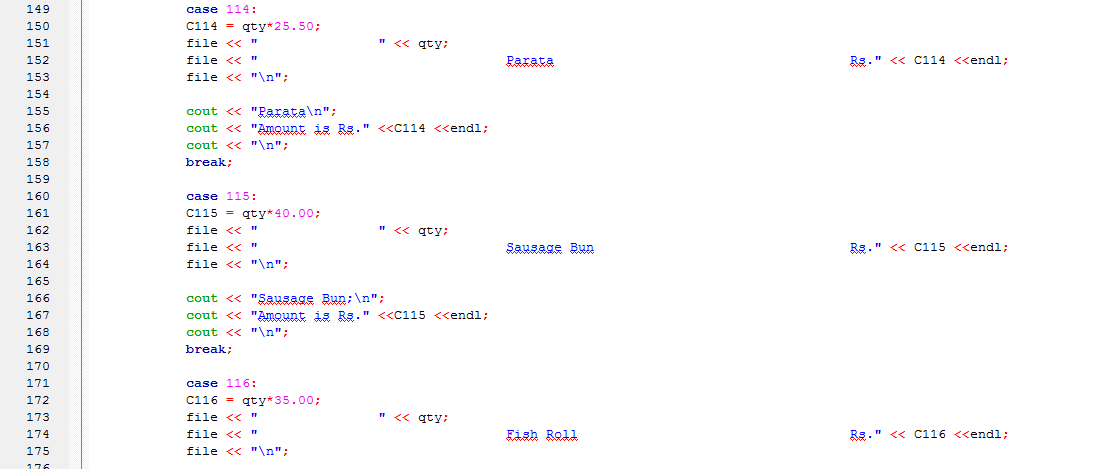


Figure 2.5 d – Order Menu

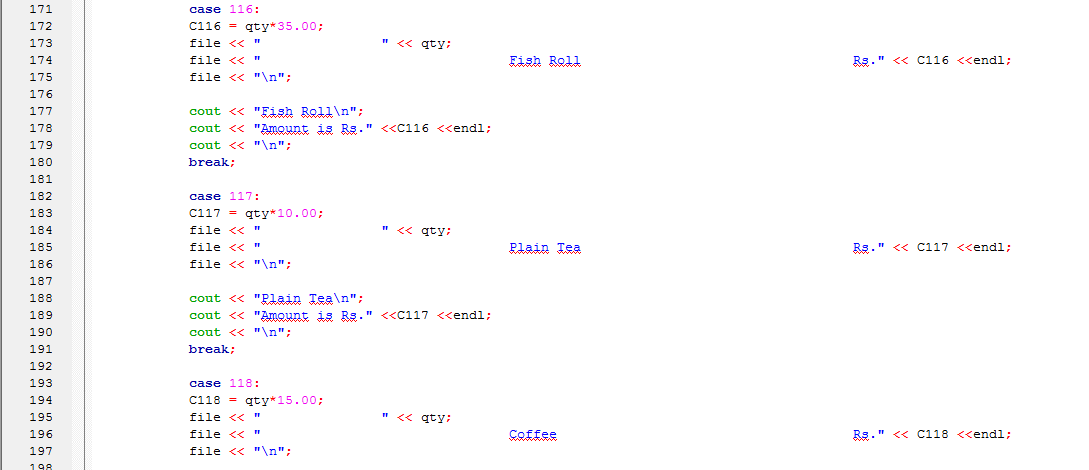
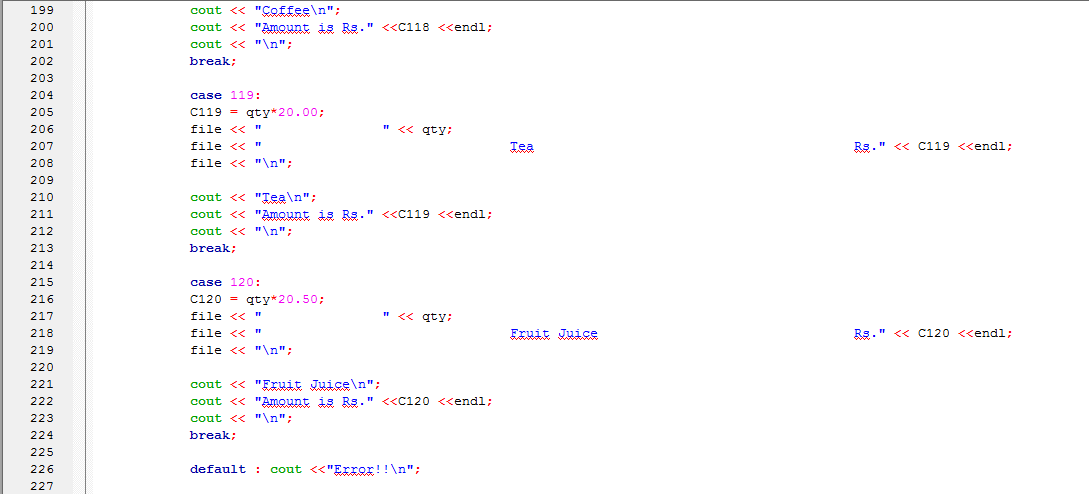


Figure 2.5 e – Order Menu

Figure 2.5 f – Order Menu

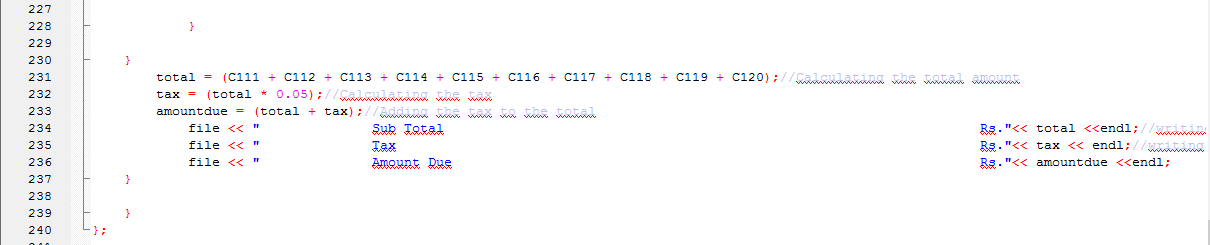


Figure 2.5 g – Order Menu

So in this figure shows the total bill calculation. After the user has ordered the items the sum of all the items will be calculated in total variable, tax will be calculated in tax variable and the amount due will be calculated in amount due variable. After that all the outputs are written to the file.

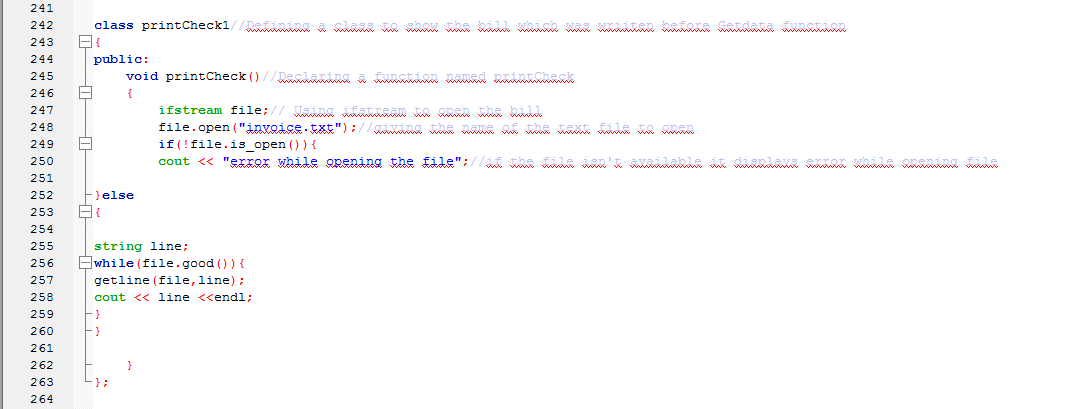


Figure 2.6 – Display and print bill

In this figure it shows the **printCheck** function and class. In here the printing and displaying of bill is done here. I have used “**Ifstream**” file class again to show up the written items of the bill to the program.

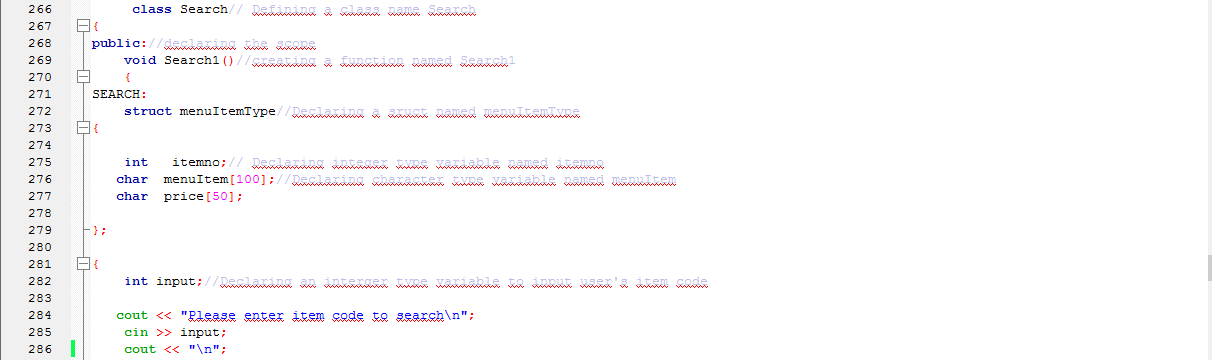


Figure 2.7 –Search Menu

This is the Search function and class. In here it search and gives the relavent menu item and type based on what the user has input for item code. So, for that, I have used a struct named ”menuItemType” to display the information of menu items. Then I have declared one integer type variable to store item number and two character type variables to store item name and price. After that I declare another integer type variable to assign user’s input (item code) to it. Next the system will ask the user to input the item code to search.

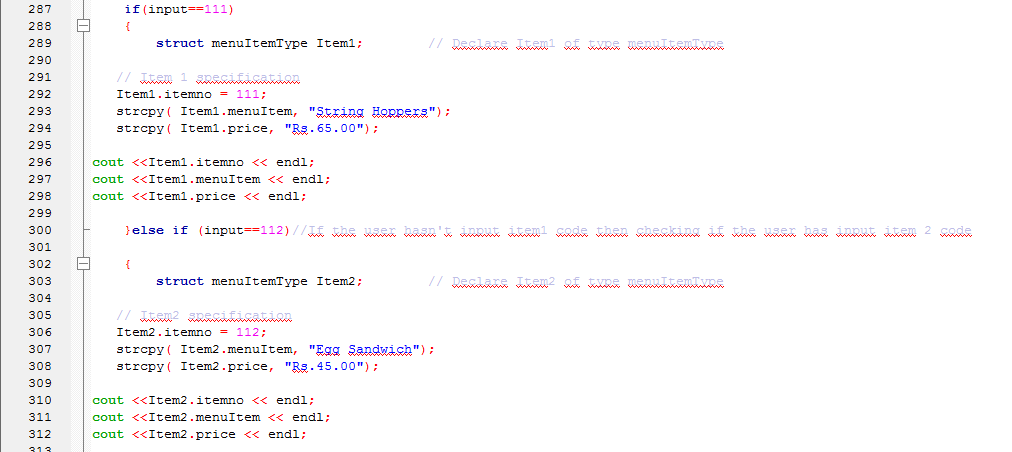


Figure 2.7 a – Search Menu

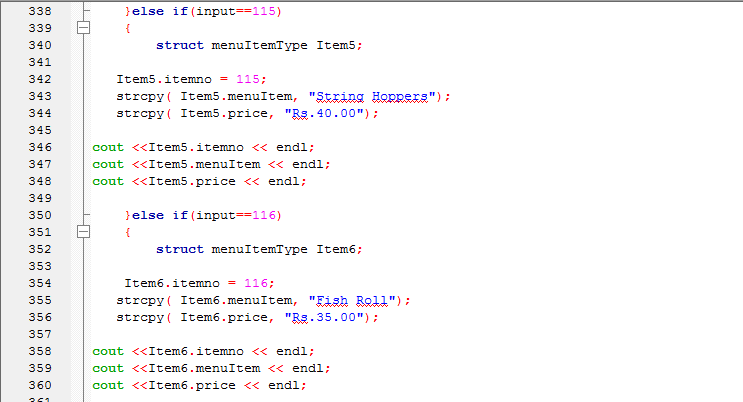
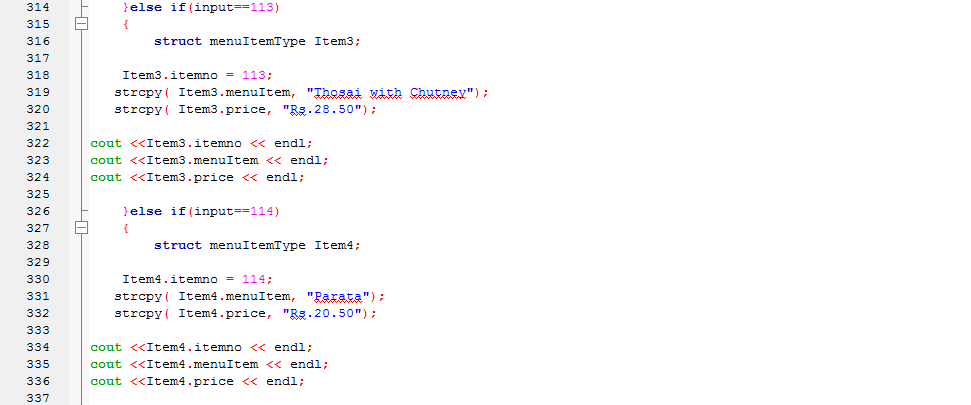
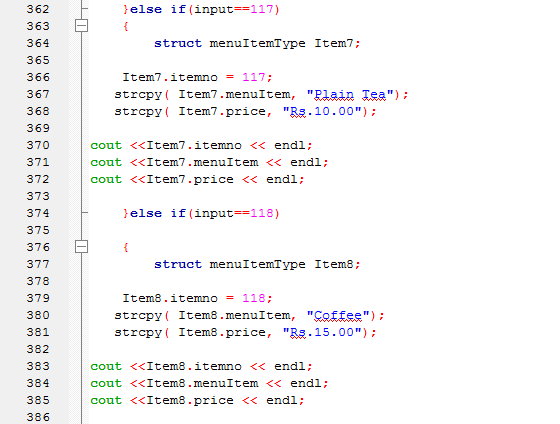
So I have used If condition again to display the relevant item details according to the user’s input. For example if the user has input the code as “111”, then under that I have declared “Item1” of the type menuItemType and assign the values for the three variable itemno, menuItem and price as “111”, “String Hoppers” and “Rs.65.00” and then finally displays them. Likewise in below figures I have written the conditions for the item code 112,113, 114,115, 116,117,118,119 and 120.

Figure 2.7 c – Search Menu

Figure 2.7 b – Search Menu



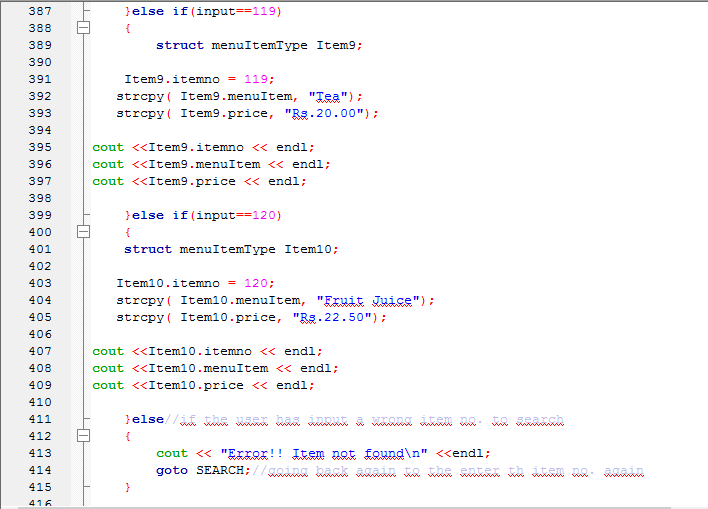


Figure 2.7 e – Search Menu

Figure 2.7 d – Search Menu

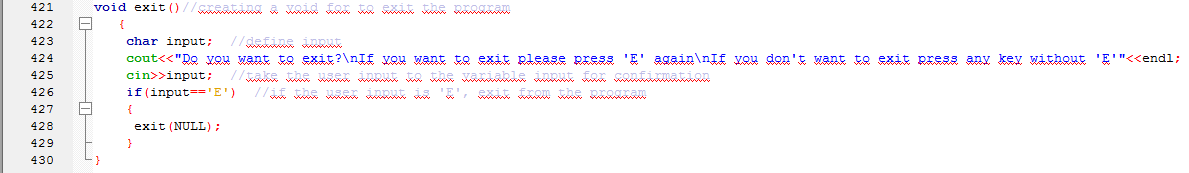


Figure 2.8 – Exit Menu

This figure shows the exit function. In here I have declared a character type variable named ”input” for the user to input letter “E” to close the program. The program asks whether the user want to input letter “E” to exit or else enter any letter to go back. In here I have used if condition to check the user’s input. If the user has input “E” it exits the program. Else it would go back to the main menu.

After developing these functions the program will be complete and it’s ready to use.

## Codes for the Puff and Fresh Breakfast Billing System

#include <iostream>

#include<fstream>

#include<string>

#include<stdlib.h>

#include <cstring>

#include <conio.h>

using namespace std;

class showMenu1//Defining a class to show the menu

{

public:

void ShowMenu()//creating a function named ShowMenu

{

std::string line;

ifstream file("menu.txt");//using ifstream to read and display the menu list from a file

if(file.is\_open())

{

while(getline(file,line))

{

std::cout<<line<<'\n';

}file.close();

} else

std:: cout<<"Erorr!\n";

std:: cin.get();

}

};

class Loginform//creating a class named loginform

{

public:

void Loginform1()//creating a function named loginform1

{

START:

//user login form which is displayed in the system

system("color B");//adding system colour

string adminu = "admin"; //Declaring two variables to store username and password

string adminp = "123";

string adminu1;//Declaring another two variables to store user's input to username and password

string adminp1;

cout << "Username:\n";

cin >> adminu1;//user entering username

cout << "Password:\n";

cin >> adminp1;//user have to enter password and after that it assign it to theadminp1 varible

if((adminu1==adminu)&&(adminp==adminp1))//Checks whether the username and password are correct

{

cout<<"User valid\n";//if the login details are correct it displays"User valid"

}else

{

cout<<"Invalid Login\n";//when login details are wrong it goes back again to the begining

goto START;

}

system("cls");//Clears the system

}

};

class Getdata1//creating a class named Getdata1

{

public:

void Getdata()//creating function named Getdata

{

ofstream file ("invoice.txt");//openting and naming the text file to write the user's ordered items to the bill

if(!file.is\_open())

{

cout << "Unable to open the file\n";

}else

{

//Writing the heading to the invoice text file

file << " ===================================================================================\n";

file << " \*\*\*Welcome to Puff and Fresh\*\*\*\n";

file << " \*\*\*Breakfast Billing System\*\*\*\n";

file << " ====================================================================================\n";

file << "\n";

file << " Amount Item Price\n";

file << "\n";

int \*pointer = NULL;//Declaring dynamically allocated array based on user's input using pointers

cout << "How many items you are going to enter"<<endl;

int input;//Declaring a variavle to input user's amount of items going to order to create an array

cin >> input;//taking he user's input and assigning it to the input variable

pointer = new int[input];

int temp, qty;//declaring two interger variables temp to input item code and qty to input quantity

double C111 = 0;//declaring ten double type variables to input calculations for each item selected by user

double C112 = 0;

double C113 = 0;

double C114 = 0;

double C115 = 0;

double C116 = 0;

double C117 = 0;

double C118 = 0;

double C119 = 0;

double C120 = 0;

double total;//declaring double type variable named total

double tax;//declaring double type variable named tax

double amountdue;

for(int counter =0; counter < input ; counter++)

{

cout << "Enter the Item code "<<counter+1<<endl;

cin >> temp;//user's input is assigned to the temp variable

cout << "\n";

cout << "Enter Quantity " << endl;

cin >> qty;//user's input is assigned to the temp variable

cout << "\n";

\*(pointer+counter) = temp;

switch(temp)//Using switch case statement to do the calculations based on user's input

{

case 111://switch case statement for the first item to calculate if the user has ordered it

C111 = qty\*65.00;//calculating the total

file << " " <<qty;//writing the quantity to the bill

file << " String Hoppers Rs." << C111 <<endl;//writing the item name and price after calculating to the file

file << "\n";

cout << "String Hoppers\n";//displaying the item name

cout << "Amount is Rs." <<C111 <<endl;//displaying the amount

cout << "\n";

break;

case 112://switch case statement for the second item to calculate if the user has ordered it

C112 = qty\*45.00;//calculating the total

file << " " << qty;//writing the quantity to the bill

file << " Egg Sandwich Rs." << C112 <<endl;//writing the item name and price after calculating to the file

file << "\n";

cout << "Egg Sandwich\n";//displaying the item name

cout << "Amount is Rs." <<C112 <<endl;//displaying the amount

cout << "\n";

break;

case 113:

C113 = qty\*28.00;

file << " " << qty;

file << " Thosai and Chutney Rs." << C113 <<endl;

file << "\n";

cout << "Thosai and Chutney\n";

cout << "Amount is Rs." <<C113 <<endl;

cout << "\n";

break;

case 114:

C114 = qty\*25.50;

file << " " << qty;

file << " Parata Rs." << C114 <<endl;

file << "\n";

cout << "Parata\n";

cout << "Amount is Rs." <<C114 <<endl;

cout << "\n";

break;

case 115:

C115 = qty\*40.00;

file << " " << qty;

file << " Sausage Bun Rs." << C115 <<endl;

file << "\n";

cout << "Sausage Bun;\n";

cout << "Amount is Rs." <<C115 <<endl;

cout << "\n";

break;

case 116:

C116 = qty\*35.00;

file << " " << qty;

file << " Fish Roll Rs." << C116 <<endl;

file << "\n";

cout << "Fish Roll\n";

cout << "Amount is Rs." <<C116 <<endl;

cout << "\n";

break;

case 117:

C117 = qty\*10.00;

file << " " << qty;

file << " Plain Tea Rs." << C117 <<endl;

file << "\n";

cout << "Plain Tea\n";

cout << "Amount is Rs." <<C117 <<endl;

cout << "\n";

break;

case 118:

C118 = qty\*15.00;

file << " " << qty;

file << " Coffee Rs." << C118 <<endl;

file << "\n";

cout << "Coffee\n";

cout << "Amount is Rs." <<C118 <<endl;

cout << "\n";

break;

case 119:

C119 = qty\*20.00;

file << " " << qty;

file << " Tea Rs." << C119 <<endl;

file << "\n";

cout << "Tea\n";

cout << "Amount is Rs." <<C119 <<endl;

cout << "\n";

break;

case 120:

C120 = qty\*20.50;

file << " " << qty;

file << " Fruit Juice Rs." << C120 <<endl;

file << "\n";

cout << "Fruit Juice\n";

cout << "Amount is Rs." <<C120 <<endl;

cout << "\n";

break;

default : cout <<"Error!!\n";

}

}

total = (C111 + C112 + C113 + C114 + C115 + C116 + C117 + C118 + C119 + C120);//Calculating the total amount

tax = (total \* 0.05);//Calculating the tax

amountdue = (total + tax);//Adding the tax to the total

file << " Sub Total Rs."<< total <<endl;//writing the total amount to the text file as the bill

file << " Tax Rs."<< tax << endl;//writing the tax to the text file as the bill

file << " Amount Due Rs."<< amountdue <<endl;

}

}

};

class printCheck1//Defining a class to show the bill which was wriiten before Getdata function

{

public:

void printCheck()//Declaring a function named printCheck

{

ifstream file;// Using ifstream to open the bill

file.open("invoice.txt");//giving the name of the text file to open

if(!file.is\_open()){

cout << "error while opening the file";//if the file isn't available it displays error while opening file

}else

{

string line;

while(file.good()){

getline(file,line);

cout << line <<endl;

}

}

}

};

class Search// Defining a class name Search

{

public://declaring the scope

void Search1()//creating a function named Search1

{

SEARCH:

struct menuItemType//Declaring a sruct named menuItemType

{

int itemno;// Declaring integer type variable named itemno

char menuItem[100];//Declaring character type variable named menuItem

char price[50];

};

{

int input;//Declaring an interger type variable to input user's item code

cout << "Please enter item code to search\n";

cin >> input;

cout << "\n";

if(input==111)

{

struct menuItemType Item1; // Declare Item1 of type menuItemType

// Item 1 specification

Item1.itemno = 111;

strcpy( Item1.menuItem, "String Hoppers");

strcpy( Item1.price, "Rs.65.00");

cout <<Item1.itemno << endl;

cout <<Item1.menuItem << endl;

cout <<Item1.price << endl;

}else if (input==112)//If the user hasn't input item1 code then checking if the user has input item 2 code

{

struct menuItemType Item2; // Declare Item2 of type menuItemType

// Item2 specification

Item2.itemno = 112;

strcpy( Item2.menuItem, "Egg Sandwich");

strcpy( Item2.price, "Rs.45.00");

cout <<Item2.itemno << endl;

cout <<Item2.menuItem << endl;

cout <<Item2.price << endl;

}else if(input==113)

{

struct menuItemType Item3;

Item3.itemno = 113;

strcpy( Item3.menuItem, "Thosai with Chutney");

strcpy( Item3.price, "Rs.28.50");

cout <<Item3.itemno << endl;

cout <<Item3.menuItem << endl;

cout <<Item3.price << endl;

}else if(input==114)

{

struct menuItemType Item4;

Item4.itemno = 114;

strcpy( Item4.menuItem, "Parata");

strcpy( Item4.price, "Rs.20.50");

cout <<Item4.itemno << endl;

cout <<Item4.menuItem << endl;

cout <<Item4.price << endl;

}else if(input==115)

{

struct menuItemType Item5;

Item5.itemno = 115;

strcpy( Item5.menuItem, "String Hoppers");

strcpy( Item5.price, "Rs.40.00");

cout <<Item5.itemno << endl;

cout <<Item5.menuItem << endl;

cout <<Item5.price << endl;

}else if(input==116)

{

struct menuItemType Item6;

Item6.itemno = 116;

strcpy( Item6.menuItem, "Fish Roll");

strcpy( Item6.price, "Rs.35.00");

cout <<Item6.itemno << endl;

cout <<Item6.menuItem << endl;

cout <<Item6.price << endl;

}else if(input==117)

{

struct menuItemType Item7;

Item7.itemno = 117;

strcpy( Item7.menuItem, "Plain Tea");

strcpy( Item7.price, "Rs.10.00");

cout <<Item7.itemno << endl;

cout <<Item7.menuItem << endl;

cout <<Item7.price << endl;

}else if(input==118)

{

struct menuItemType Item8;

Item8.itemno = 118;

strcpy( Item8.menuItem, "Coffee");

strcpy( Item8.price, "Rs.15.00");

cout <<Item8.itemno << endl;

cout <<Item8.menuItem << endl;

cout <<Item8.price << endl;

}else if(input==119)

{

struct menuItemType Item9;

Item9.itemno = 119;

strcpy( Item9.menuItem, "Tea");

strcpy( Item9.price, "Rs.20.00");

cout <<Item9.itemno << endl;

cout <<Item9.menuItem << endl;

cout <<Item9.price << endl;

}else if(input==120)

{

struct menuItemType Item10;

Item10.itemno = 120;

strcpy( Item10.menuItem, "Fruit Juice");

strcpy( Item10.price, "Rs.22.50");

cout <<Item10.itemno << endl;

cout <<Item10.menuItem << endl;

cout <<Item10.price << endl;

}else//if the user has input a wrong item no. to search

{

cout << "Error!! Item not found\n" <<endl;

goto SEARCH;//going back again to the enter th item no. again

}

}

};

};

void exit()//creating a void for to exit the program

{

char input; //define input

cout<<"Do you want to exit?\nIf you want to exit please press 'E' again\nIf you don't want to exit press any key without 'E'"<<endl;

cin>>input; //take the user input to the variable input for confirmation

if(input=='E') //if the user input is 'E', exit from the program

{

exit(NULL);

}

}

int main()

{

//displaying the program heading

cout << "\n";

cout <<" =========================================\n";

cout <<" \*\*\*Welcome to Puff and Fresh\*\*\*\n";

cout <<" \*\*\*Breakfast Billing System\*\*\*\n";

cout <<" ==========================================\n";

cout << "\n";

cout << "\n";

cout << "PLEASE ENTER LOGIN DETAILS BELOW\n";

cout << "\n";

Loginform a;//creating an object instace to display the loginform in the system

a.Loginform1();

cout << "\n";

cout << "PLEASE SELECT AND ENTER ONE OF THE OPTIONS BELOW TO PROCEED\n";

cout << "\n";

int choice;//declaring a variable named choice to select options for the user

cout << " [1] BREAKFAST MENU\n";//Displaying the options for the user to select

cout << " [2] MAKE AN ORDER\n";

cout << " [3] SEARCH AN ITEM\n";

cout << " [4] EXIT\n";

cin >> choice;//taking the user;s input and assigning it to the variable

system("cls");

if (choice==1)//If the user has selected option 1 the statements that the system has to execute

{

showMenu1 b;//display the menu by creatinng an object instance

b.ShowMenu();

cout <<"\n";

Getdata1 c;//creatinng an object instance to get the data

c.Getdata();

cout <<"\n";

printCheck1 d;//creatinng an object instance to print and display the bill

d.printCheck();

cout << "\n";

}else if (choice==2)//If the user has selected option 2 the statements that the system has to execut

{

Getdata1 c;//creatinng an object instance to display Getdata

c.Getdata();

cout <<"\n";

printCheck1 d;

d.printCheck();

cout << "\n";

}else if (choice==3)//If the user has selected option 3 the statements that the system has to execute

{

Search e;//Creating an object instance to display search menu

e.Search1();

cout << "\n";

}else if(choice==4)//If the user has selected option 4 the statements that the system has to execute

{

system("color C");

exit();//displaying the exit

system("cls");

main();//If the user didn't input "E", the system comes back to the begining

}else

{

cout << "Error\n";

}

return 0;

}

## Modularization

This section includes the functions we have used for this system to design. When we use functions it’s easy to read the program and reduces the complexity of it. Below are the functions that I have used for the program.

* main()
* Loginform1()
* showMenu()
* getData()
* printCheck()
* Search1()
* exit()

So in previous section, I have mentioned about these functions and discussed their process before.

# Task 3 – System Testing



## What is System Testing?

After developing a system, it should be tested properly to check if there are any errors or not. Sometimes there can be coding errors, designing errors or requirement errors in the system. So it’s very important to identify these errors and correct them. Every type of input in the system should be tested whether the expected output can be gained properly from this system. Also we should check whether the non- functional requirements are fulfilled or not. There three steps in testing.

* Unit Testing – Testing the components of the system separately
* Integration Testing – Ensuring that whether the components of the system are integrated and the tasks are done properly giving the proper output.
* Accepting testing – testing the whole system to check whether the expected output can be received

Also, there are two types of testing

* White box testing - Test the program for any syntax, logical or any other errors in the program.
* Black box testing- Test the program with the inputs, outputs etc.

## Test Case

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test name | Changes made | Expected result | Actual result | Pass/Fail |
| Testing the login menu in the system with correct username and password. | Give the correct username and password. | Log on to the system. | Log on to the system. | Pass |
| Testing the login menu in the system with incorrect username and password | Gave an invalid username and password to the system. | Display error message and came back to the login. | Display error messages and coming back to the login menu. | Pass |
| Get the breakfast menu | Give the command to the system as ‘1’. | Get the breakfast menu. | Get the breakfast menu. | Pass |
| Test the system to make an order after entering to the breakfast menu. | Input some items to order | Get the available item with the price. | Get the available item with the price | Pass |
| Test the system whether the bill is displayed and print when option “1” is selected | After the input some items checks whether the bill is displayed and print | The bill will be displayed and printed. | Bill is displayed and printed | Pass |
| Test the system to make an order after entering invalid item codes | Input some invalid codes to order | Displayed “Error” | Displays “Error” | Pass |
| Get the order menu list when option “2” is selected | Give the command to the system as ‘2’. | Display the order menu | Displays the order menu | Pass |
| Test the system whether the bill is displayed and print when option “2” is selected | After the input some items checks whether the bill is displayed and print | The bill will be displayed and printed. | Bill is displayed and printed | Pass |
| Search an item with an available item. | Give the command as ‘3’ and give an available item. | Displays the relevant item with the price | Displays the relevant item with the price | Pass |
| Search an item without an available item. | Give the command as ‘3’ and give an unavailable item. | Displayed “Error” and came back to search menu | Displayed “Error” and comes back to search menu | Pass |
| Test the system to exit the program. | Give the command as ‘4’, and enter “E” to exit | Exit the program | Exit the program. | Pass |
| Test the system if the user has input another value after selecting option “4” | Give the command as ‘4’, and entered any letter to exit | Came back to the main menu | Comes back to the main menu | Pass |

## User Manual

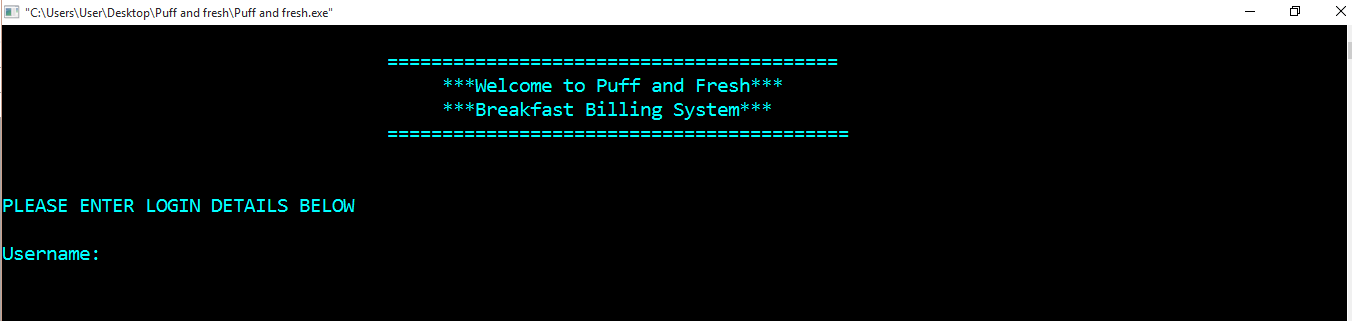
1. ** Log in to the system**

Figure 3.0

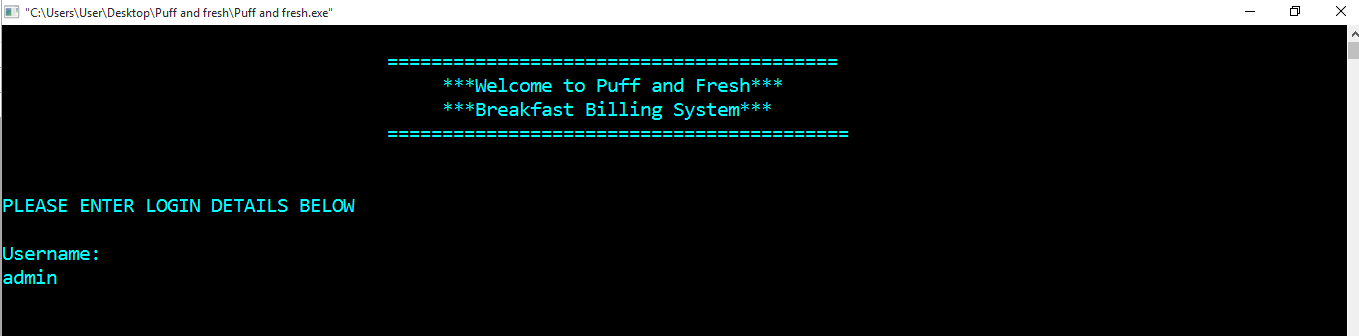
In here you will see the main menu of the billing system.

Figure 3.1

So to enter to the system first, you have to give the username

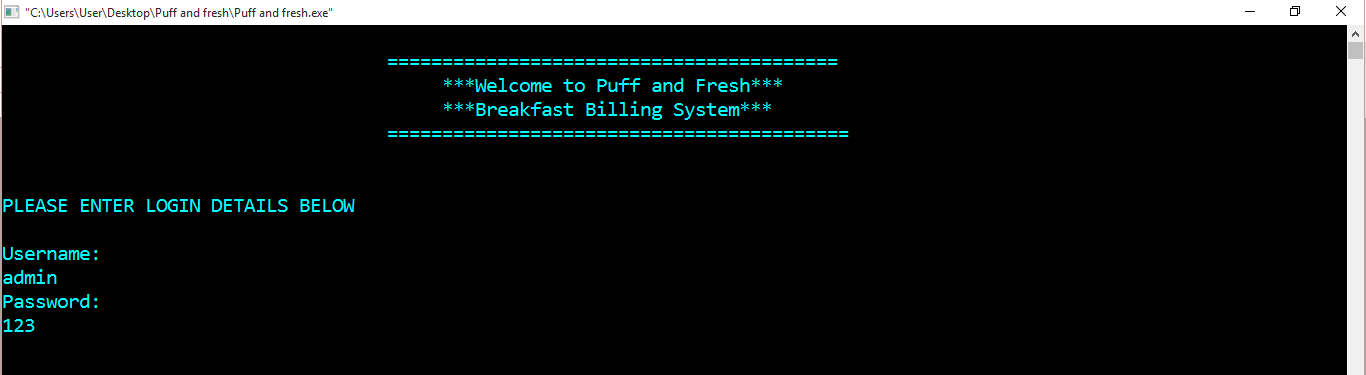


Figure 3.2

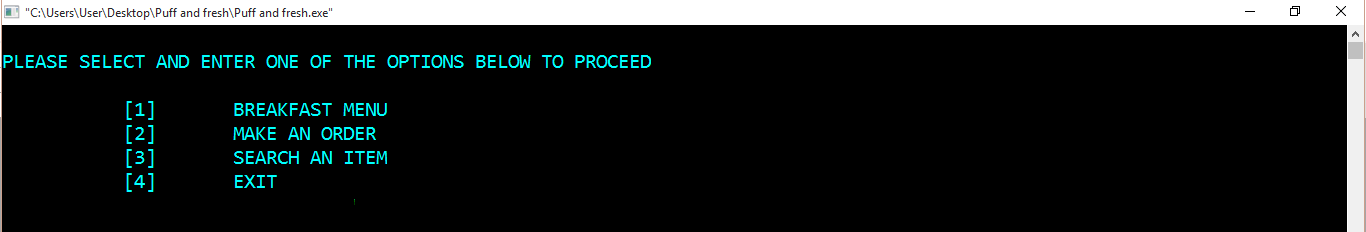
After that you have to enter the password as “123”

Figure 3.3

This is the screen after you have logged in correctly to the system.

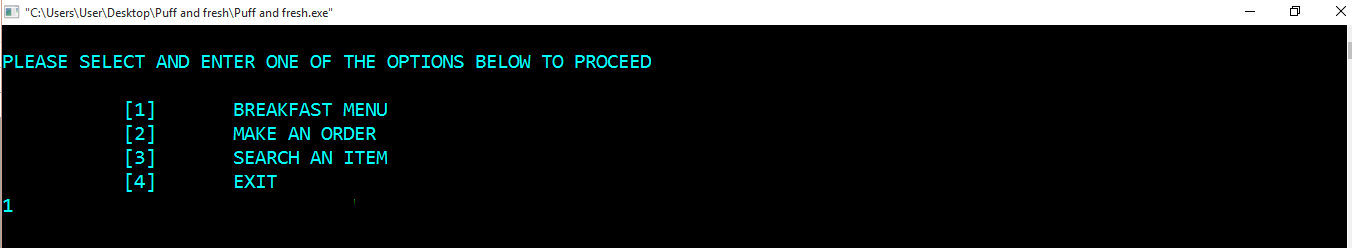
1. **How to get into the breakfast menu in to the system**

Figure 3.4

First you have to select option “1” to go to breakfast menu.

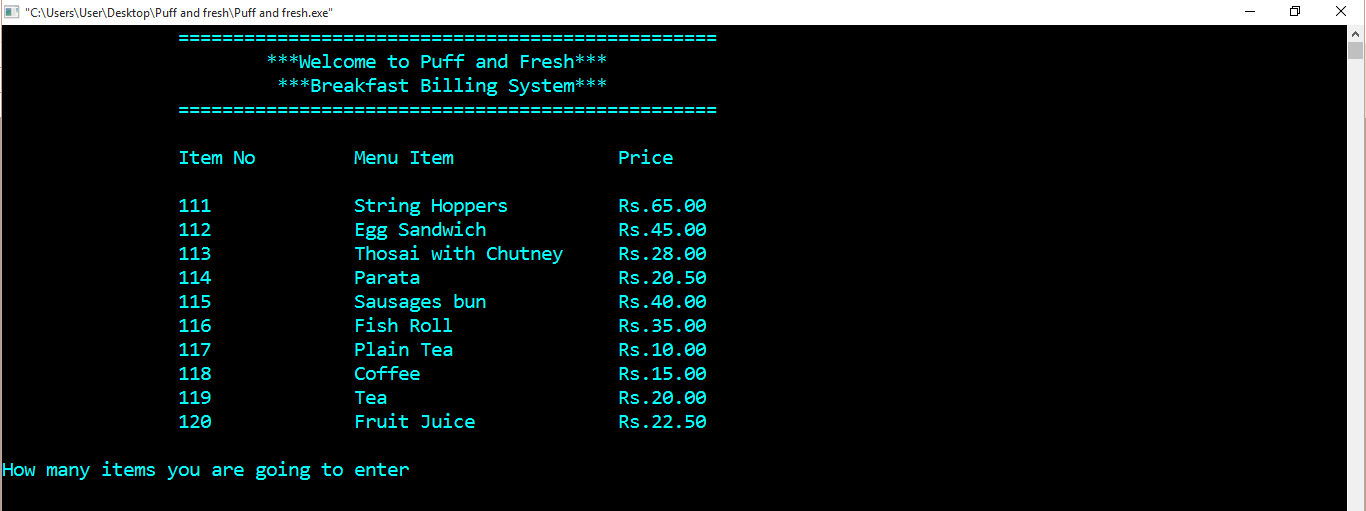


Figure 3.5

The menu list will be displayed and then you have to enter the number of items to order.



Figure 3.6

So in here, I have entered the number of items to order as three.

Figure 3.7

After that you have to input the relevant item codes to order as displayed above.

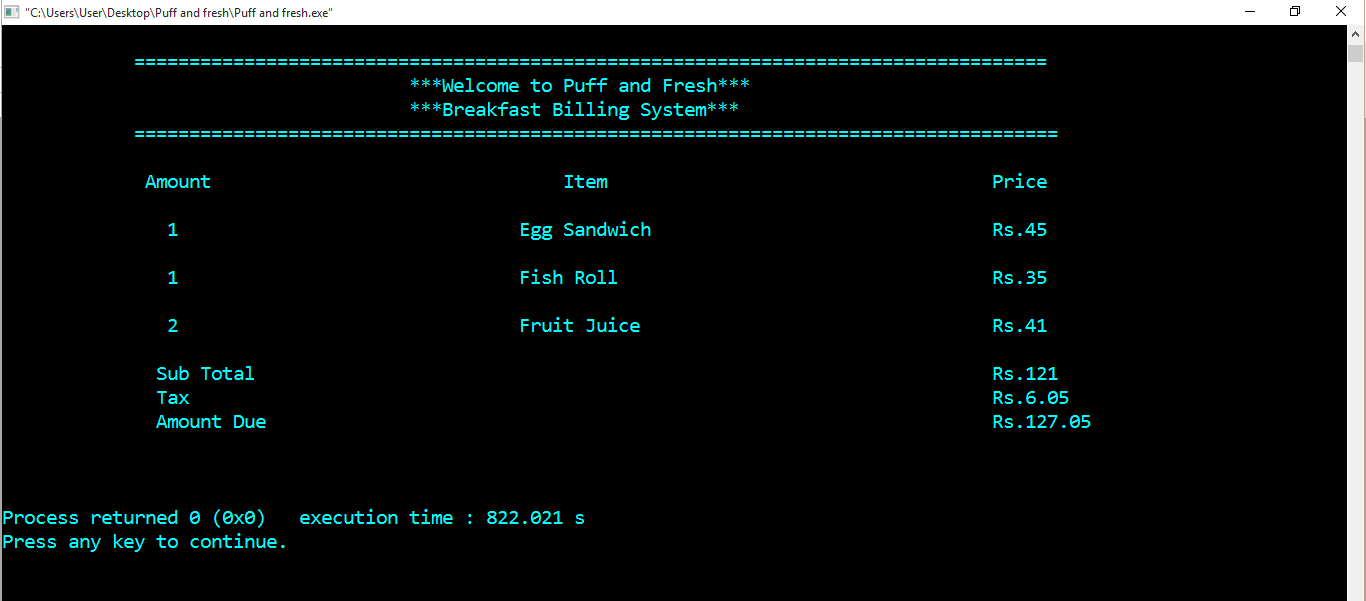


Figure 3.8

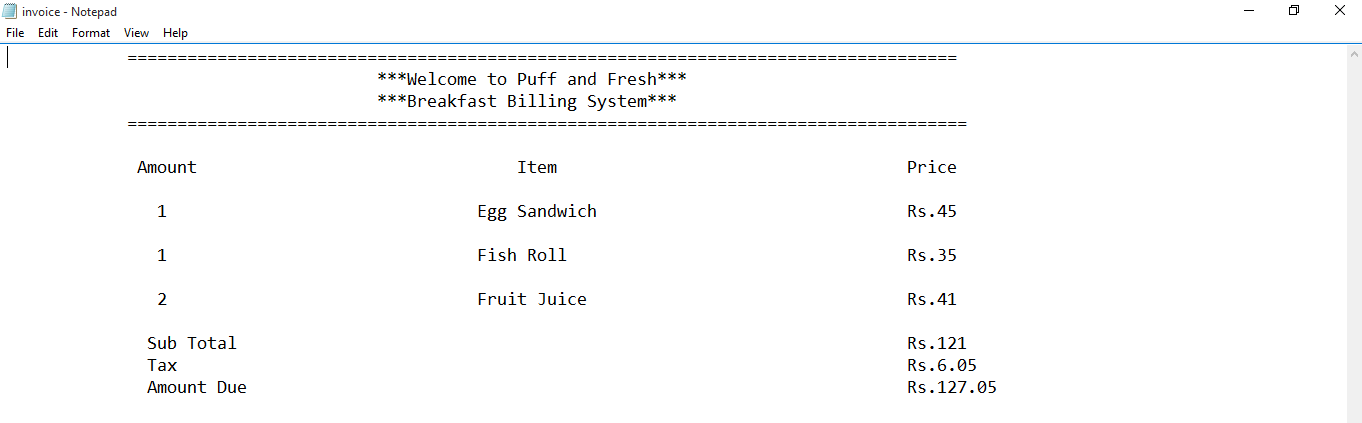
After you have ordered the items, then the ordered items will be printed in the bill and displayed in the system.

Figure 3.9

This is the bill that will be printed in the text file.

1. **How order items directly without showing the menu**

You have to enter “3” in the option menu to proceed

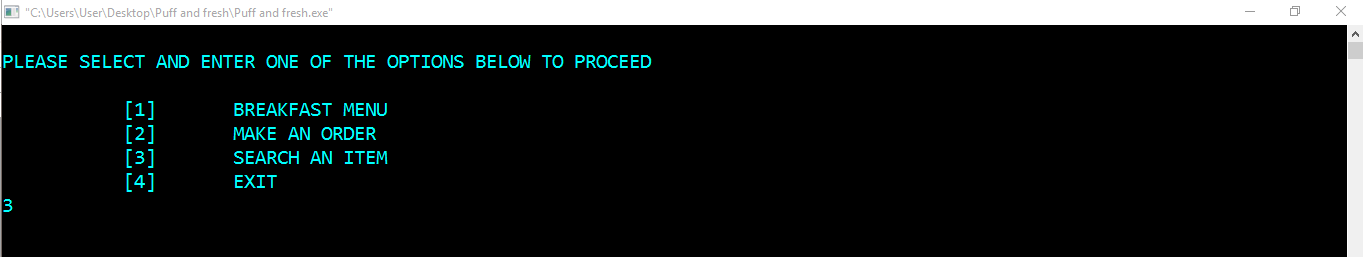
1. **How to search an item?**

Figure 3.10

To get into search menu, you have to select option “3” first.



Figure 3.11

Then input an item code to search.

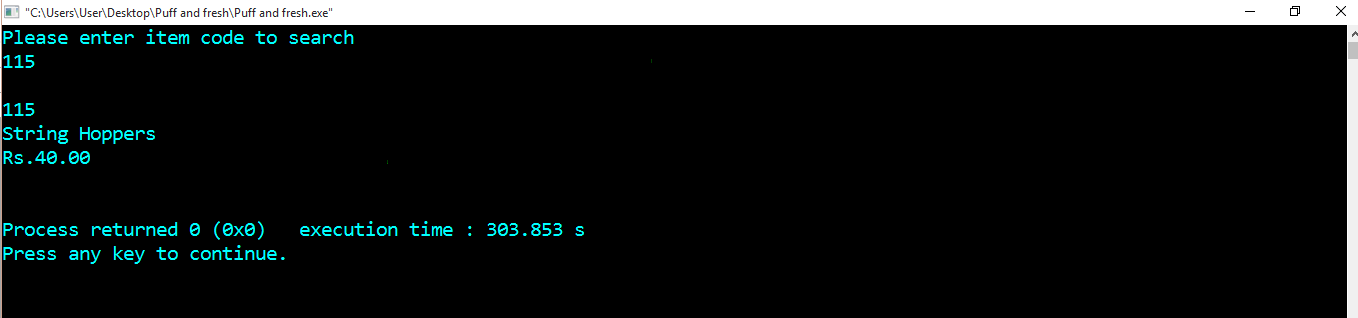


Figure 3.12

After that the system will display the information.

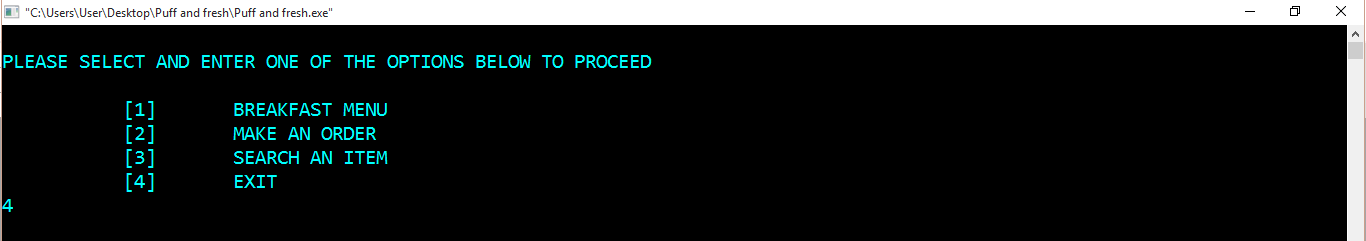
1. **How to exit from the system**

Figure 3.13

First you have to select option “4” to proceed.

Figure 3.14

Enter “E” to exit.

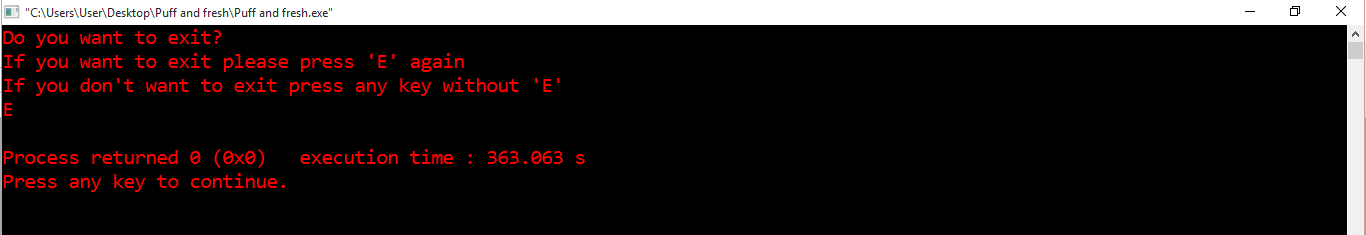


Figure 3.15

The program has exit.



## User Feedbacks

1. Is the program easy to use?

Yes No

1. Are you satisfied with the flow of the program?

Yes No

1. Is the program easy to navigate?

Yes No

1. Is the program user - friendly?

Yes No

1. Are you satisfied with the responding time of the program?

Yes No

1. Are you satisfied with the program?

Yes No

1. Are you satisfied with the interface of the program?

Yes No

**Comments**

The above figure displays the user feedback questionnaire to check the feedbacks of the users of this program. I selected four people to gather feedbacks about this program. Below are their feedbacks about the program

**Nuwan**

1. Is the program easy to use?

Yes No

1. Are you satisfied with the flow of the program?

Yes No

1. Is the program easy to navigate?

Yes No

1. Is the program user - friendly?

Yes No

1. Are you satisfied with the responding time of the program?

Yes No

1. Are you satisfied with the program?

Yes No

1. Are you satisfied with the interface of the program?

Yes No

**Comments**

* Should have a good interface.
* Not much user friendly.
* If the user inputs a wrong item code in order menu, he has to close the program and reopen it.

**Amaya**

1. Is the program easy to use?

Yes No

1. Are you satisfied with the flow of the program?

Yes No

1. Is the program easy to navigate?

Yes No

1. Is the program user - friendly?

Yes No

1. Are you satisfied with the responding time of the program?

Yes No

1. Are you satisfied with the program?

Yes No

1. Are you satisfied with the interface of the program?

Yes No

**Comments**

* Should be more user friendly.
* Should have more options

**Senali**

1. Is the program easy to use?

Yes No

1. Are you satisfied with the flow of the program?

Yes No

1. Is the program easy to navigate?

Yes No

1. Is the program user - friendly?

Yes No

1. Are you satisfied with the responding time of the program?

Yes No

1. Are you satisfied with the program?

Yes No

1. Are you satisfied with the interface of the program?

Yes No

**Comments**

* It’s better if the program would develop more with more features.

1. Is the program easy to use?

Yes No

1. Are you satisfied with the flow of the program?

Yes No

1. Is the program easy to navigate?

Yes No

1. Is the program user - friendly?

Yes No

1. Are you satisfied with the responding time of the program?

Yes No

1. Are you satisfied with the program?

Yes No

1. Are you satisfied with the interface of the program?

Yes No

**Comments**

* Should be more user friendly.
* Should have more options

After gathering feedbacks, we are now going to analyze them by using a table.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Question | Nuwan | Amaya | Senali |  |
| Is the program easy to use |  |  |  |  |
| Are you satisfied with the flow of the program? |  |  |  |  |
| Is the program easy to navigate? |  |  |  |  |
| Is the program user – friendly? |  |  |  |  |
| Are you satisfied with responding time of the program |  |  |  |  |
| Are you satisfied with the program? |  |  |  |  |
| Are you satisfied with the interface of the program? |  |  |  |  |

After analyzing their feedbacks to a table we can come to a conclusion that the program is good enough to use. As we analyze the feedbacks we can see that many of the people say that the interface isn’t much good. So we should improve the interface of our program too. Also when we analyze the user comments, many of them say that it’s much better if the program should have more options and features. One user has said that the program doesn’t have options like “Go Back”, “OK” options. So in that case we should develop our program with more options to navigate and features to make it more user friendly.

## Testing Conclusion

We build up a good system for Puff and Fresh Breakfast Billing System. So we can conclude that we have built up a good system for them due to following reasons and after analyzing those user feedbacks. We had a good plan so at the first stage we designed flowcharts to the system on how the system should flow and we developed the system by following those designed flowcharts before. So because of that we could finish up making the system so easily. Also our program is very easy to use. There isn’t much complexity in our program anyone can understand and handle it. The program is user – friendly and very easy to navigate because we add some navigation options to it too. The calculations in this system are done accurate. Also the responding time of our program is good. It was a requirement of the restaurant to have a fast and reliable service from this system to the customers.

Also we had a test plan for this system whether the system runs successfully meeting the user requirements. We had a test case and tested each and every components of the system and expected results were all passed in each components. Also the screenshots in the user manual also proves that the system functions smoothly without any errors. As I said previously at the beginning we got good responses from the users about this system. Many of them said that the system is very easy to use, user – friendly and easy to navigate and has a good flow. They have suggested to add up more features more options to the system.

So we can come to a conclusion that we have built up a good system for Puff and Fresh Restaurant.

## Recommendation

So finally we have built up a good new breakfast billing system for Puff and Fresh Restaurant successfully without any errors. So in this section I would like to discuss some recommendations for this system to improve its quality. Previously this restaurant has been using a manual billing system. So when we are implementing this system for the first time in the restaurant the users will be unfamiliar with system. So its better if we have a system training for the users teaching on how the system is functioning, how can users use it and what does the system do. Also we can provide the user manual to the users which includes on how to use this system.

We can leave this system after implementing this thinking we have finished the job. We have to maintain this system regularly to ensure the quality if this system. So I suggest some future recommendations for this system. In the user feedbacks some users mentioned that the interface should improve. It’s because this system runs in Command Prompt so we can’t add graphics and images to it. So I recommend that its good if this system is developed in a software like Visual Studio to solve that problem. Also, we can link a database to this software to add more users and more menu items. Currently the system includes only one user, which is the administrator and to add a new menu item in this system we should go to the source code of the program and add the menu item, price and the relevant calculations for it. By following these recommendations I think we can improve the quality of this system.

## References

* Tutorialspoint (2016) *Test case*. Available at: https://www.tutorialspoint.com/software\_testing\_dictionary/test\_case.htm (Accessed: 21 October 2016).
* Tutorialspoint (2016) *System testing*. Available at: https://www.tutorialspoint.com/software\_testing\_dictionary/system\_testing.htm (Accessed: 21 October 2016).