**ASANSOL ENGINEERING COLLEGE**

Abstract

Topic :- Restaurant Management System

Presented By:-

NAME:- Bishal Bhattacharjee

ROLL NO:- 10800221097

SUBJECT:-Data Structure and Algorithm

SUBJECT CODE:- PCC CS-301

REGISTRATION NO:-211080100210112

YEAR:-Second

SEMESTER:- 3rd

In the guidance of – >

Mr. Sudip Kr.De (IT Dept.)

RMS that is, Restaurant Management Systems are the crucial technologies that enables a single outlet or enterprise to better serve its customers and aid employees with food and beverage transactions and controls. Restaurant management System is database program that keeps record of all transaction carried out in the restaurant on daily bases. Ultimately, restaurant management is concerned with guiding a restaurant to long-term success .The Restaurant Management System helps the restaurant management to keep adequate record of all transactions carried out and does that will still be carried out by the restaurant and maintain the database of the restaurant.

Introduction

Restaurant management system is the system for manage the restaurant business. Restaurant management can vary across multiple management styles, however, there is always one common denominator when it comes to setting goals: maximizing a restaurant’s profitability.

This is the simple model of a RMS created using C Programming Language. In

this typical implementation of RMS there is two data structures involed:

i. Array Data-Structure

ii. Single Linked List

iii. Structure

iv.Double Linked-List

V.DMA

Structure are a way to group several related variables into one place. Each variable in the structure is known as a member of the structure. Unlike an array, a structure can contain many different data types.

DMA is a process in which we allocate or deallocate a block of memory during the run-time of a program.

Array Data Structure is used in the form of Matrix to store the Customer IDfor the given Slot and Table. Further it is used to help in the displaying of theseat matrix or the availibilty of seats.

Linked List is used to create a dynamic list of the details of the customers.Restuarant Management System can be used for small to large scale restuarant business, Food trucks or in the canteen of the workplace.The benefits associated with the use of restaurant management systems arerelated to automation and speeding up processes to improve the customerexperience. Moderncustomers expect fast, frictionless service and restaurantmanagement software play a key role in meeting these expectations.

Problem Definition

Using the concept of Object-Oriented Programming design, a “RestaurantManagement System”. Suppose in the Restaurant there are 7 Tables; Eachtable can be booked for a maximum 1Hour; the Opening hour is at 7:00 andclosing time is at 23:00.You have to maintain the customers booking details(ID, Name, Phone, OrderDetails, Date (DDMMYYYY), Order amount, Tablenumber (1/2/3…/7), Time slot (11:00-12:00, 12:00-13:00, ……., 22:00-23:00).Your system should be able to provide followings:

 Showing Menu of Foods

 Showing Occupancy View

 Total collected Amount from all the customers

 Find that customer who spent maximum for providing Gift for that day

 Printing Customers details

Methadology

In this implementation of Restuarant Management System We do not useDBMS for storing of data rather a linked list is maintained to store the data.This implementation is lightweight and has the bare minimum requirementsfor the RMS. It is good in a way if you dont want to keep the data of the usersfor a longer period of time.Now coming to the implementation we have created a User Defined data-typecalled nodewhich is required to store the customer details such as name,contact number,the reservation timing along with the table which he has booked, this is alsoused to store the total bill amount of the customer.

There are 10 method excluding MAIN method, these are :

i. void Selectmenu(); -> This is one of the helper method for booking as well as

used to get the complete menu of the restuarant. Thisis called at the time of booking for the aid customer toselect his/her desired delicacy.

ii. float order(); -> This is also a helper method for booking which helpsin taking the details of the orders and summarizing thetotal bill amount. This method is created in order tomaintain the modularity in the program which hjelps todebug the program easily.

iii. char \*Timing(int n); -> This method is independent to booking it justserves the purpose of displaying the slot and it’s respective timings.

iv. void SeatChecking(int mat[][]); -> This method is used for the purpose ofchecking seat availiblity along with thet iming slot and table number. SeatListfunction is aided by the times function fordisplaying the slot number and respectivetime in a grid similar to booking userinterface of movie/transport system seatselector.

v. void Checking();-> Keeping MAIN method aside, this is one of the mostimportant methods as it ties all the other function ina way that they co-ordinate and work together. Italso helps to determine the total collection of theday as well as highest bill amount generated. Thisway we do not need to search for the entire list togenerate the total sum and the highest bill amountwhich makes the implementation a bit faster.

vi. node \*Removed();-> Once the customer pays the bill and leaves therestuarant his reserved table should be makeavailable for other people. So in order to achieve this we use this function. Also this can be used toempty all the bookings at the time of closing the restuarant.

vii. void display();-> This method is used to display the list of the customers

for the purpose of serving the orders generated by thecustomer requirements.

viii. void finalCustomer(); -> This method is used for displaying the totalnumber of customers dinned today.

ix. void totalEarning(); -> This method is used for diplaying the totalamount of revenue earned as of today.

x. void highestOrder(); -> This method displays the Highest Bill generatedas of that day in order to give the customer a giftfrom the restuarant side.

Source Code

#include <stdlib.h>

#include <stdio.h>

struct node

{

char name[25], contactno[13];

char \*date;

int id, slot, table;

float bill;

struct node \*next;

};

typedef struct node node;

node \*head, \*tail, \*hCust;

int mat[16][7] = {0};

int c = 0;

float totalSum = 0, hBill = 0;

struct noxd

{

char foodname[50];

int quantity;

float price;

int data;

struct noxd \*prev;

struct noxd \*next;

};

struct noxd \*headc = NULL,\*newnoxd,\*tailc = NULL;

struct noxd \*heada = NULL, \*taila = NULL;

struct noxd \*head\_s;

struct noxd\* createadmin(struct noxd \*head,int data, char foodname[25], float price)

{

newnoxd = (struct noxd\*)malloc(sizeof(struct noxd));

newnoxd->data = data;

newnoxd->price = price;

newnoxd-> quantity = 0;

strcpy(newnoxd->foodname,foodname);

newnoxd->next = NULL;

newnoxd->prev = NULL;

struct node \*temp = head;

if(temp==NULL)

heada = taila = newnoxd;

else

{

while(temp->next!=NULL)

temp=temp->next;

temp->next=newnoxd;

newnoxd->prev = taila;

taila = newnoxd;

}

return heada;

}

void menu()

{

printf("----ME N U-------\n");

printf(" 1>Panner Chilly \n");

printf("2> Panner Butter Masala\n");

printf(" 3>Chicken Haka Noodles\n");

printf("4> MOMO\n");

printf("5>Lemonade\n");

printf("6>Chicken Tikka\n");

printf("7>Sharbat\n");

printf("8>Cofee\n");

printf("9>nan\n");

printf("10>kulcha\n");

printf("11>tarka\n");

printf("12>tea\n");

}

struct node\* createcustomer(struct noxd \*head,int data,int quantity)

{

newnoxd = (struct noxd\*)malloc(sizeof(struct noxd));

struct noxd \*temp1 = heada;

int flag = 0;

while(temp1!=NULL)

{

if(temp1->data==data)

{

flag = 1;

break;

}

temp1 = temp1->next;

}

if(flag==1)

{

newnoxd->data = data;

newnoxd->price = quantity\*(temp1->price);

newnoxd-> quantity = quantity;

strcpy(newnoxd->foodname,temp1->foodname);

newnoxd->next = NULL;

newnoxd->prev = NULL;

struct noxd \*temp = head;

if(temp==NULL)

headc = tailc = newnoxd;

else

{

while(temp->next!=NULL)

temp=temp->next;

temp->next=newnoxd;

newnoxd->prev = tailc;

tailc = newnoxd;

}

}

else

{

printf("\n\t\t\t\t\t\t\tThis item is not present in the menu!\n");

}

return headc;

}

float order()

{

float rate, sum = 0;

while (1)

{

int ch, plates;

printf("Choice : ");

scanf("%d", &ch);

printf("Plates : ");

scanf("%d", &plates);

if (ch == 1)

{

rate = 200 \* plates;

}

else if (ch == 2)

{

rate = 350 \* plates;

}

else if (ch == 3)

{

rate = 130 \* plates;

}

else if (ch == 4)

{

rate = 399 \* plates;

}

else if (ch == 5)

{

rate = 249 \* plates;

}

else if (ch == 6)

{

rate = 359 \* plates;

}

else if (ch == 7)

{

rate = 110 \* plates;

}

else if (ch == 8)

{

rate = 269 \* plates;

}

else if (ch == 9)

{

rate = 210 \* plates;

}

else if (ch == 10)

{

rate = 170 \* plates;

}

else if (ch == 11)

{

rate = 89 \* plates;

}

else if (ch == 12)

{

rate = 59 \* plates;

}

sum = sum + rate;

printf("\nWant More :\n1.YES\n2.NO\n>>>");

scanf("%d", &ch);

if (ch == 2)

break;

}

return sum;

}

char \*times(int n)

{

switch (n)

{

case 1:

return "7:00 - 8:00 a.m";

case 2:

return "8:00 - 9:00 a.m";

case 3:

return "9:00 -10:00 a.m";

case 4:

return "10:00 - 11:00 a.m";

case 5:

return "11:00- 12:00 p.m";

case 6:

return "12:00 - 1:00 p.m";

case 7:

return "1:00 - 2:00 p.m";

case 8:

return "2:00 - 3:00 p.m";

case 9:

return "3:00 - 4:00 p.m";

case 10:

return "4:00-5:00 p.m";

case 11:

return "5:00 - 6:00 p.m";

case 12:

return "6:00 - 7:00 p.m";

case 13:

return "6:00 -7:00 p.m";

case 14:

return "7:00 - 8:00 p.m";

case 15:

return "8:00 - p.m";

case 16:

return "9:00 - p.m";

case 17:

return "10:00 - p.m";

default:

return "";

}

}

void seatList(int mat[16][7])

{

printf("\n Tables No are:-\n");

for (int k = 0; k < 7; k++)

{

printf("%d\t\n", k + 1);

}

for (int i = 0; i < 16; i++)

{

for (int j = 0; j < 8; j++)

{

if (mat[i][j] == 0)

{

if (j == 0)

{

printf("%s\t\t", times(i + 1));

}

printf("\nAvailable\n");

}

else

{

printf(" \nNotAvailable\n");

}

}

printf("\n");

}

}

void booking()

{

int tab, slot;

printf("\n\nSeats Availability:\n\n");

seatList(mat);

printf("\nSelect Table Number : ");

scanf("%d", &tab);

printf("Select Slot Number : ");

scanf("%d", &slot);

fflush(stdin);

if (mat[slot - 1][tab - 1] != 0)

{

printf("\nWrong Choice: Already Booked !!!!\n");

return;

}

if (tab - 1 < 0 || tab - 1 > 6 || slot - 1 < 0 || slot - 1 > 15)

{

printf("\nWrong Choice: Either slot or table !!!!\n");

return;

}

mat[slot - 1][tab - 1] = ++c;

node \*temp = (node \*)malloc(sizeof(node));

printf("Name : ");

scanf(" %[^\n]%\*c", temp->name);

printf("Contact Number : ");

scanf(" %[^\n]%\*c", temp->contactno);

menu();

temp->id = c;

temp->date = \_\_DATE\_\_;

temp->table = tab;

temp->slot = slot;

temp->bill = order();

temp->next = NULL;

totalSum = totalSum + temp->bill;

if (temp->bill > hBill)

{

hBill = temp->bill;

hCust = temp;

}

if (head == NULL)

{

head = temp;

tail = temp;

return;

}

tail->next = temp;

tail = temp;

}

node \*disbursed()

{

if (head == NULL)

{

printf("\nAll Tables are Empty.\n");

return NULL;

}

node \*temp = head;

head = head->next;

mat[temp->slot - 1][temp->table - 1] = 0;

printf("\n\nTable %d Cleared of %d Slot of %s\n\n", temp->table, temp->slot, times(temp->slot));

free(temp);

}

void disp()

{

if (head == NULL)

{

printf("\nNo Customer Bookings!!!\n");

}

node \*mover = head;

while (mover != NULL)

{

printf("\nName : %s\n", mover->name);

printf("Contact Number : %s\n", mover->contactno);

printf("Customer ID : %d\n", mover->id);

printf("Timing : %s\n", times(mover->slot));

printf("Table : %d\n", mover->table);

printf("Date : %s\n", mover->date);

printf("Total Bill : %.2f\n\n", mover->bill);

mover = mover->next;

}

}

void totalCustomer()

{

if (c != 0)

printf("\n\nTotal Customers Today : %d\n\n", c);

else

printf("\n\nNo Bookings yet.!!!\n\n");

}

void totalCollection()

{

if (totalSum != 0)

printf("\n\nTotal Collection of the Day is : Rs.%.2f\n\n", totalSum);

else

printf("\n\nNo Bookings yet.!!!\n\n");

}

void highestBill()

{

if (hBill != 0)

{

printf("\n\nTodys's Highest Bill : Rs.%.2f of ", hBill);

printf("\nName : %s\nContact Number : %s\n\n", hCust->name, hCust->contactno);

}

else

printf("\n\nNo Bookings yet.!!!\n\n");

}

void main(int argc, char const \*argv[])

{

printf("WELCOME TO THE RESTUARANT BOOKING\n");

while (1)

{

int ch;

printf("\nEnter\n1.Booking\n2.Seats Availiblity\n3.Slot Timing\n4.Menu\n5.List Of Customers\

n6.Make table availble\n");

printf("7.Total Customer of the Day\n8.Total Collection of the Day\n9.Highest Bill of the Day\

n10.Close\n\n");

printf(">>>");

scanf("%d", &ch);

switch (ch)

{

case 1:

booking();

break;

case 2:

seatList(mat);

break;

case 3:

printf("\nSlot\t: Timings-\n");

for (int i = 1; i <= 16; i++)

{

printf("%d\t: %s\n", i, times(i));

}

break;

case 4:

menu();

break;

case 5:

disp();

break;

case 6:

disbursed();

break;

case 7:

totalCustomer();

break;

case 8:

totalCollection();

break;

case 9:

highestBill();

break;

case 10:

printf("\*\*\*\*\*\*Restuarant is Closing\*\*\*\*\*\*\n");

exit(0);

default:

break;

}

}

}

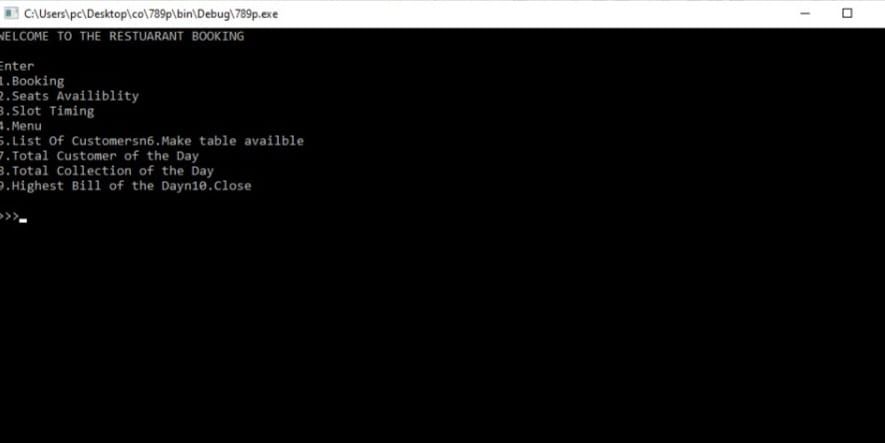


Fig 1: Using Menu to enter the choices

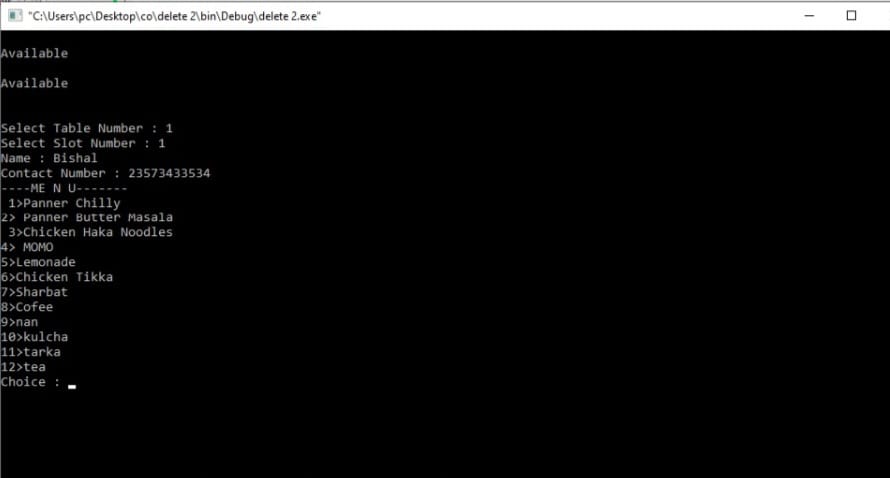


Fig 2: Using Booking for selecting seats and Ordering dishes. 

Fig 3: Checking the available seat for the required time

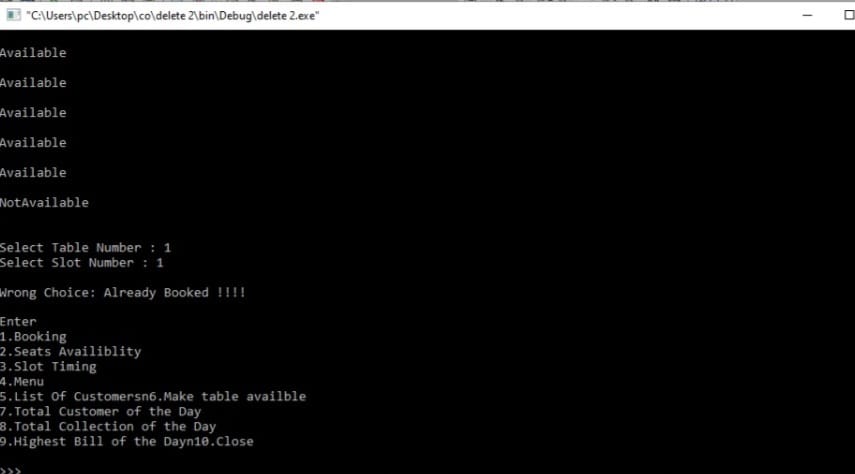


Fig 4:Checking if the slot or table chooose is correct or no

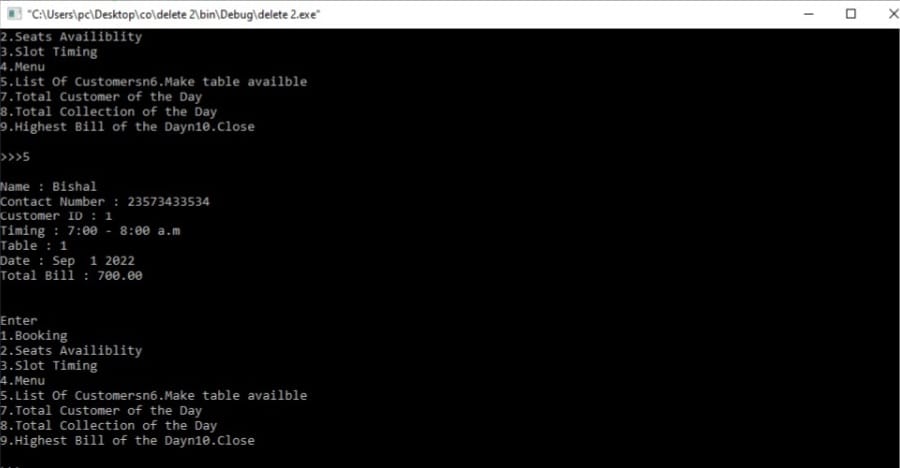


Fig 5: List of the customer in that particular time

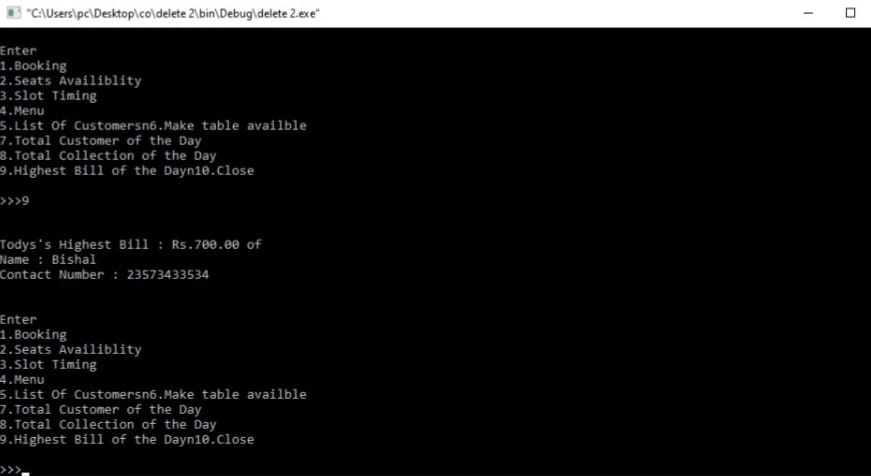


Fig 6: Highest Bill of the day and the user who ordered

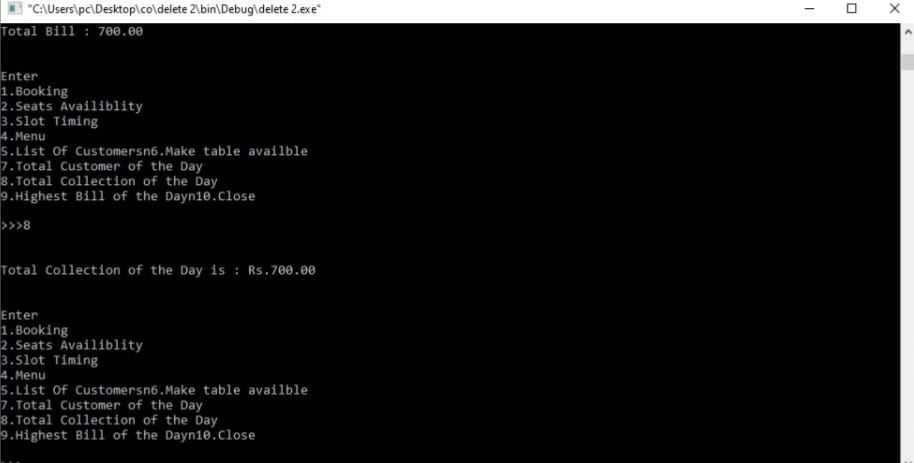


Fig :7 Total Collection of the day and the amount



Fig :8 Ordering Food and estimating the quantity

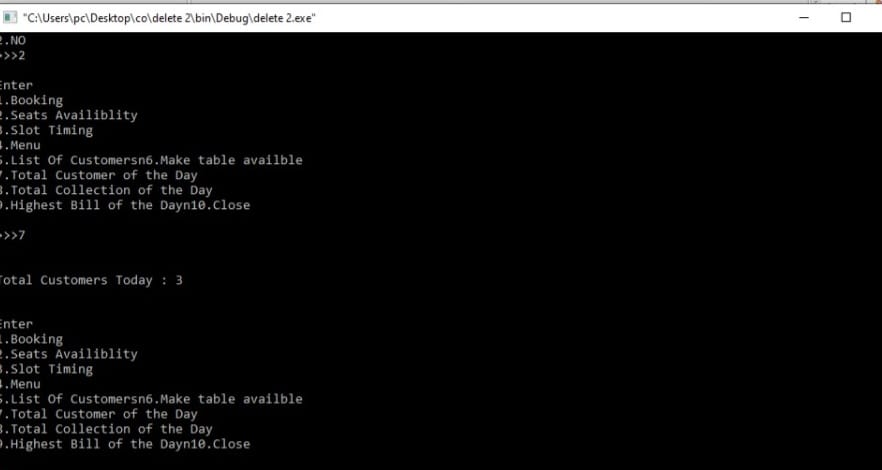


Fig :9 Total Customer on that day

Conclusion and Futurework

We were able to create a computerized system for Silk Route to maintainbilling & Restaurant records .This system is able to store data and retrieve the records whenever needed easily.Data entering ofcustomers and employees are also included in this system along with theorder and the billing process.Customers,restaurant records and employeesare interconnected in order to maintain the accuracy of this system .This system can also be further improved adding many other features and including the other systems as well. Finally we believe that we were ableto launch an effective computerized system to the restaurant causing therestaurant to perform well in the future regarding the billing and restaurantrecords.