

ALGORITHM AND PROBLEM SOLVING

PROJECT REPORT

ON

“BUS TICKET BOOKING SYSTEM”



Submitted By-

Mridnal Garg (9919102032)
Pranati Tiwari (9919102056)
Pranjal Shukla (9919102060)

Submitted To-

Dr. Swati Gupta

TABLE OF CONTENTS

1. ABSTRACT

2. INTRODUCTION

3. OVERVIEW AND PLANNING

- 2.1 Proposed Work
- 2.2 Hardware Requirements
- 2.3 Software Requirements

4. LITERATURE SURVEY

- 3.1 Literature Summary

5. METHODOLOGY

- 4.1 Method Used
- 4.2 Applications

6. SYSTEM IMPLEMENTATIONS

- 5.1 Code
- 5.2 Results and discussion

7. CONCLUSION

8. REFERENCES

ABSTRACT

The program is based Linked – List and is completely made on C/C++ language. The program allows its user to add and alter data in the program. The menu driven access makes it easier for any user to use the program sufficiently.

A reservation system saves the details of all the Buses and can be manipulate them. This program will help a person to manage data of each bus efficiently. The program allows a person to add new bus in the list, with its additional information like name of Driver of the bus, the place from where bus starts its journey, the place where Bus have to reach, time of departure, time of arrival, cancellation of any bus and most importantly, reservation of passengers. Ticket booking, ticket cancellation, viewing details of each bus, and such tasks can be performed.

At any particular moment, total earnings can be seen, number of tickets booked in a particular bus can be known.

INTRODUCTION

The menu driven program asks for the task which is to be performed in the first page. A person can book a ticket by entering '6' (Reservation of seats) and then the program will ask to enter bus number in which one wishes to book ticket, then the program asks desired seats, and then the name of person after each detail is filled, a confirmations message shows up that ticket is booked, if there is any clash with seat number or something else, the user is notified immediately about the same. After the confirmation message, details are stored in the program.

If someone wants to cancel the reservation, they can simply do it by pressing '7' in the menu driven program. Daily income can be seen under '8', and similarly for other commands.

But before booking any seats, there need to be a bus for it, which can be done by entering '1' in the command window. After pressing '1' in the command window, the program asks to enter bus code, Driver's name, Arrival time (in format HH:MM) , Departure time, Destination, starting place and Price of one ticket. After receiving all the details, the program saves the information for further use like reservation.

Status of buses can be known by pressing '3' or '5'. One shows details of all the buses while one shows details of particularly one bus only.

PROPOSED DESIGN

1. Fill bus details
2. Check bus details
3. Booking a ticket
4. Cancellation of ticket
5. Cancellation of bus

Hardware requirements

There is no specific system requirement. The program can run on any device having C/C++ compiler.

Software requirements

1. Code Blocks (Or any other C compiler (recommended - GNUCC))

3.1 Literature summary

Structures that are defined in the program are

Time: To take input of time in proper format

Node: To store Bus information like bus number, bus code, driver's name and other things.

The functions defined are –

- int cunt(struct node *q)
- void line()
- void srch(int num)
- void Delete(int num)
- void insert(int x)
- void Print(struct node *q)
- void reservation(struct node *q)
- void cancel()
- void daily_profit()

Methods Used

- Structures

Structures are used to store data of same bus under same name / position.

- Linked lists

Linked lists are used to go through one bus details to other bus details.

- Multi-Dimensional matrix

Multi-Dimensional matrix is used to store information of seats (reservation) of any particular bus.

Application

Bus reservation system has application in government or private transport offices where there is a need of storing data like these, this program can be very useful.

Code Snippet

```
#include<iostream>
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
using namespace std;

char i=':.';
char chEmpty[]="Empty";
struct time{
    int iHrs,iMins;
};

static int Num=0;
struct node{
    int bus_no;
    int iBusNum;
    int iBusCode;
    char chSeat[8][4][20];
    char chDrivers_Name[20];
    struct time stDeparture_Time, stArrival_Time;
    char chGoesTo[20];
    char chGoesFrom[20];
    float fFare;
    int iTicketSold;
    struct node *next;
}*p;

int cunt(struct node *q);
void line();
void srch(int num);
```

```
void Delete(int num);  
void insert(int x);  
void Print(struct node *q);  
void reservation(struct node *q);  
void cancel();  
void daily_profit();
```

```
int main(){  
    int A;  
    while(1){  
        line();  
        cout << "\n1.ADD NEW BUS";  
        cout << "\n2.REMOVE A BUS";  
        cout << "\n3.SHOW ALL THE BUSES";  
        cout << "\n4.TOTAL NUMBER OF BUSES";  
        cout << "\n5.SEARCH BUS";  
        cout << "\n6.Reservation of seats";  
        cout << "\n7.Cancel a reservation";  
        cout << "\n8.Daily profit";  
        cout << "\n0.EXIT";  
        cout << "\n\nEnter your choice here : ";  
  
        cin >> A;  
        switch(A){  
            case 1:{  
                int x,n,i;  
                line();  
                cout << "Enter bus code : ";  
                cin >> x;  
                insert(x);  
                break;
```

```

    }
case 2:{
    line();
    cout << "\n Enter Bus code : ";
    int num;
    cin >> num;
    Delete(num);
    Print(p);
}
case 3:{
    line();
    Print(p);
    break;
}
case 4:{
    line();
    cout << "Total number of Buses are \n" << cunt(p);
    break;
}
case 5:{
    int num;
    line();
    cout << "Enter Bus number : ";
    cin >> num;
    if(num>cunt(p)||num<1){
        cout << "Invalid number !! \n";
        main();
    }
    srch(num);
    break;
}
case 6:{

```



```

        line();
        reservation(p);
        break;
    }
case 7:{
    line();
    cancel();
    break;
}
case 8:{
    line();
    daily_profit();
    break;
}
case 0:{
    line();
    exit(1);
    break;
}
default:{
    cout << "\n\nWrong choice...";
    break;
}
}
}
}

```

//counting total number of buses

```

int cunt(struct node *q){
    struct node *temp;
    temp=q;
    int tot=0;

```

```

while(temp!=NULL) {
    tot++;
    temp=temp->next;
}
return (tot);
}

```

```

void line(){
    int i;
    cout << endl;
    for(i=0;i<60;i++){
        cout << "*";
    }
    cout << endl;
}

```

//Searching a Bus its bus number

```

void srch(int num){
    struct node *r;
    r=p;
    if(num==1) {
        cout << "\nBus code -> " << r->bus_no;
        cout << "\tDriver's name -> " << r->chDrivers_Name;
        cout << "\nArrival time -> " << r->stArrival_Time.iHrs << i << r->stArrival_Time.iMins;
        cout << "\tDeparture time -> " << r->stDeparture_Time.iHrs << i << r->stDeparture_Time.iMins;
        cout << "\nFrom -> " << r->chGoesFrom;
        cout << "\t To -> " << r->chGoesTo;
        cout << "\nFare -> " << r->fFare;
        cout << "\t Tickets sold -> %d" << r->iTicketSold;

        int index=0;
        for(int i=0;i<8;i++){

```

```

        cout << endl;
        for(int j=0;j<4;j++){
            index++;
            cout << index << "." << r->chSeat[i][j];
        }
    }
}
else if(num <= cunt(p)){
    int i;
    for(i=1;i<num;i++){
        r=r->next;
    }
    cout << "\nBus code -> " << r->bus_no;
    cout << "\tDriver's name -> " << r->chDrivers_Name;
    cout << "\nArrival time -> " << r->stArrival_Time.iHrs << i << r->stArrival_Time.iMins;
    cout << "\tDeparture time -> %d%c%d" << r->stDeparture_Time.iHrs << i << r->stDeparture_Time.iMins;
    cout << "\nFrom -> " << r->chGoesFrom;
    cout << "\t To -> " << r->chGoesTo;
    cout << "\nFare -> " << r->fFare;
    cout << "\t Tickets sold -> " << r->iTicketSold;
    int index=0;

    for(int i=0;i<8;i++){
        cout << endl;
        for(int j=0;j<4;j++){
            index++;
            cout << index << "." << r->chSeat[i][j];
        }
    }
}
else{
    cout << "\n Wrong Bus number input (Please Enter Bus Number only!)\n";

```

```

    }
}

```

//deleting a bus entry by its Bus code by its value

```

void Delete(int num){
    struct node *temp,*r;
    r=p;
    while(r!=NULL){
        if(r->bus_no==num){
            if(r==p){
                p=r->next;
                free(r);
                return;
            }
            else{
                temp->next=r->next;
                free(r);
                return;
            }
        }
        else{
            temp=r;
            r=r->next;
        }
    }
    cout << "No such Bus code found.\n";
}

```

```

void insert(int x){
    struct node *temp,*m;
    m = p;
    temp =(struct node *)malloc(sizeof(struct node));

```

```
temp->bus_no=x;
Num++;
temp->iBusNum=Num;
```

dname:

```
cout << "Enter driver's name : ";
fflush(stdin);
gets(temp->chDrivers_Name);
if(strlen(temp->chDrivers_Name)>20){
cout << "Maximum 20 characters are allowed";
goto dname;
}
```

aTime:

```
printf("Enter arrival time : ");
fflush(stdin);
scanf("%d%c%d",&temp->stArrival_Time.iHrs,&i,&temp->stArrival_Time.iMins);
if(temp->stArrival_Time.iHrs == 0){
cout << "\nInvalid time.";
goto aTime;
}
if((i!=':')||((temp->stArrival_Time.iHrs>=24)||((temp->stArrival_Time.iHrs<0)
||((temp->stArrival_Time.iMins>=60)||((temp->stArrival_Time.iMins<0))){
cout << "\nInvalid time !\n\tOnly format HH:MM accepted\n";
goto aTime;
}
```

dTime:

```
cout << "Enter departure time : ";
fflush(stdin);
scanf("%d%c%d",&temp->stDeparture_Time.iHrs,&i,&temp->stDeparture_Time.iMins);
if(temp->stDeparture_Time.iHrs==0){
```

```

        cout << "\nInvalid time.";
        goto dTime;
    }
    if((i!=':')||(temp->stDeparture_Time.iHrs>=24)||(temp->stDeparture_Time.iHrs<0)
        ||(temp->stDeparture_Time.iMins>=60)||(temp->stDeparture_Time.iMins<0)){
        cout << "\nInvalid time !\n\t\tOnly format HH:MM accepted\n";
        goto dTime;
    }

```

to:

```

    cout << "Enter destination : ";
    fflush(stdin);
    gets(temp->chGoesTo);
    if(strlen(temp->chGoesTo)>20){
        cout << "\nOnly 20 characters or less";
        goto to;
    }

```

from:

```

    cout << "Enter starting place : ";
    fflush(stdin);
    gets(temp->chGoesFrom);
    if(strlen(temp->chGoesFrom)>20){
        cout << "\nOnly 20 characters or less";
        goto from;
    }

```

```

    cout << "Enter fare of one ticket : ";
    fflush(stdin);
    cin >> temp->fFare;
    temp->iTicketSold=0;
    int i1,i2;
    for(i1=0;i1<8;i1++){

```

```

        for(i2=0;i2<4;i2++){
            strcpy(temp->chSeat[i1][i2],chEmpty);
        }
    }
    if(p==NULL){
        p=temp;
        p->next=NULL;
    }
    else{
        while(m->next!=NULL)
            m=m->next;
        m->next=temp;
        m=temp;
        m->next=NULL;
    }
}

void Print(struct node *q){
    struct node *r;
    r=q;
    int cnt=1;
    while(r!=NULL){
        line();
        cout << "\n\nBus code -> " << r->bus_no;
        cout << "\tBus number -> " << cnt;
        cout << "\tDriver's name -> " << r->chDrivers_Name;
        cout << "\nArrival time -> " << r->stArrival_Time.iHrs << i << r->stArrival_Time.iMins;
        cout << "\tDeparture time -> %d%c%d" << r->stDeparture_Time.iHrs << i << r->stDeparture_Time.iMins;
        cout << "\nFrom -> " << r->chGoesFrom;
        cout << "\tTo -> %s" << r->chGoesTo;
        cout << "\nFare -> " << r->fFare;
        cout << "\tTickets sold -> " << r->iTicketSold;
    }
}

```

```

    cnt++;
    int i,j,index=0;
    for(i=0;i<8;i++){
        cout << endl;
        for(j=0;j<4;j++){
            index++;
            cout << index << "." << r->chSeat[i][j] << "\t";
        }
    }
    r = r->next;
}
cout << endl;
}

```

```

void reservation(struct node *q){
    struct node *r;
    r=q;
    int num,count=0,Seat;
    res:
    cout << "Enter bus number : ";
    fflush(stdin);
    cin >> num;
    if(num>cunt(q)||(num<1)){
        printf("Sorry, no bus found with the given Bus number.\n");
        goto res;
    }
    while(count<num-1){
        r=r->next;
        count++;
    }
    seat:
    cout << "Enter seat number : ";

```



```

fflush(stdin);
cin >> Seat;
char Name[20];
if(Seat > 32){
    cout << "Only 32 seats are there.\n";
    goto seat;
}
else if(Seat<1){
    cout << "Invalid input.\n";
    goto seat;
}
else if(strcmp(r->chSeat[Seat/4][Seat%4-1],"Empty")==0){
    cout << "Enter the passenger's name : ";
    name:
    fflush(stdin);
    gets(Name);
    if(strlen(Name)>20){
        cout << "Max 20 characters allowed\n";
        goto name;
    }
    strcpy(r->chSeat[Seat/4][Seat%4-1],Name);
    cout << "\nSeat successfully reserved!\nDo you want to continue reservation ? : ";
    r->iTicketSold++;
    char ch;
    ch=getchar();
    if(ch=='y'||ch=='Y'){
        reservation(q);
    }
    else
    main();
}
else{

```

```

    cout << "Seat already booked!\n";
    goto seat;
}
}

```

```

void cancel(){
    if(Num==0){
        cout << "Bus list is empty.\n";
        main();
    }
    struct node *r;
    r=p;
    enter:
    cout << "Enter Bus number : ";
    int bus,ind=1;
    fflush(stdin);
    cin >> bus;
    if(bus<1||bus>cunt(p)){
        cout << "\nInvalid input.\n";
        goto enter;
    }
    else{
        while(ind<bus){
            r=r->next;
            ind++;
        }
    }
    cout << "Enter seat number : ";
    int sea;
    scanf("%d",&sea);
    if(strcmp(r->chSeat[sea/4][sea%4-1],"Empty")==0){
        cout << "The seat is already empty !\n";
    }
}

```

```

goto enter;
}
else{
strcpy(r->chSeat[sea/4][sea%4-1],"Empty");
cout << "Successfully canceled reservation.\n";
r->iTicketSold--;
main();
}
}

```

```

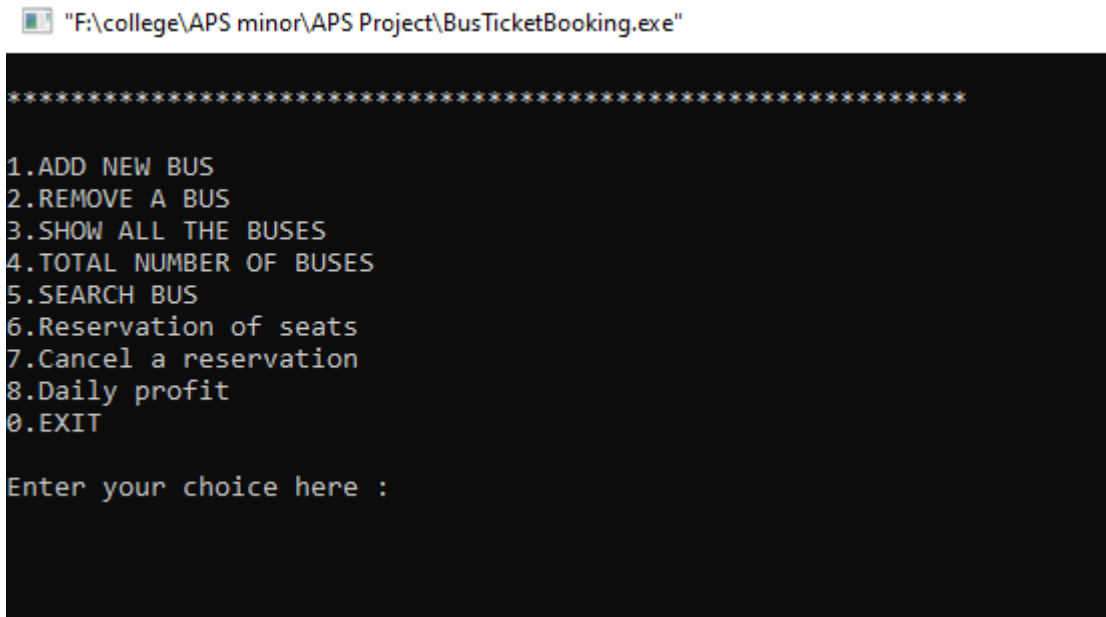
void daily_profit(){
    struct node *r,*s;

    r=p;
    s=p;
    int cnt=0;
    if(cunt(p)==0){
        cout << "No buses are available\n";
        main();
    }
    while(s!=NULL){
        if(s->iTicketSold!=0){
            cnt++;
        }
        s=s->next;
    }
    if(cnt==0){
        cout << "No booking is done yet.\n";
        getch();
        main();
    }
    float Total=0;
    while(r!=NULL){

```

```
Total=Total+(r->iTicketSold*r->fFare);  
r=r->next;  
}  
cout << "The total income of the buses is : " << Total << endl;  
main();  
}
```

OUTPUTS



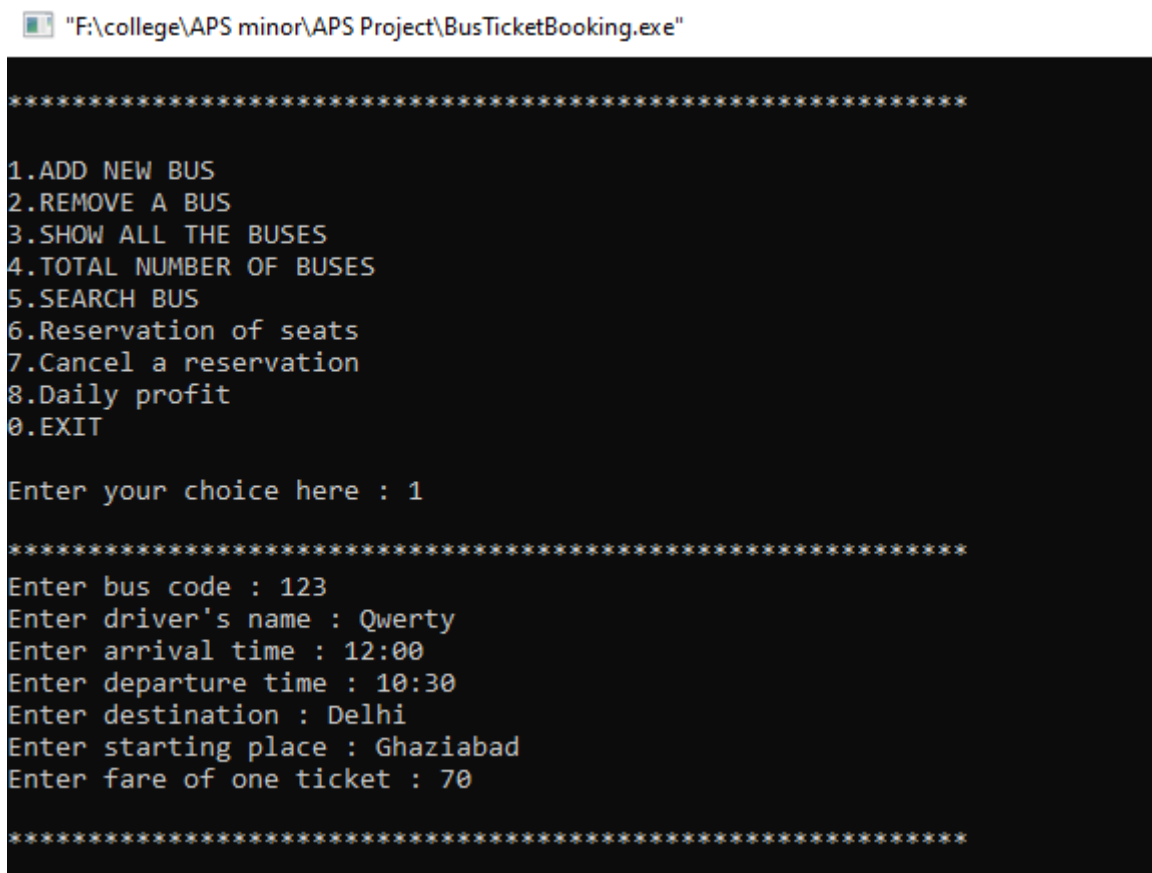
The screenshot shows a Windows application window titled "F:\college\APS minor\APS Project\BusTicketBooking.exe". The application displays a main menu with a list of options separated by asterisks. The options are: 1.ADD NEW BUS, 2.REMOVE A BUS, 3.SHOW ALL THE BUSES, 4.TOTAL NUMBER OF BUSES, 5.SEARCH BUS, 6.Reservation of seats, 7.Cancel a reservation, 8.Daily profit, and 0.EXIT. Below the list, it prompts the user to "Enter your choice here :".

```
"F:\college\APS minor\APS Project\BusTicketBooking.exe"

*****
1.ADD NEW BUS
2.REMOVE A BUS
3.SHOW ALL THE BUSES
4.TOTAL NUMBER OF BUSES
5.SEARCH BUS
6.Reservation of seats
7.Cancel a reservation
8.Daily profit
0.EXIT

Enter your choice here :
```

Fig.1 Program Main Menu



The screenshot shows the same application window after selecting option 1. It prompts the user to "Enter your choice here : 1". After pressing enter, it prompts for various details to add a new bus: "Enter bus code : 123", "Enter driver's name : Qwerty", "Enter arrival time : 12:00", "Enter departure time : 10:30", "Enter destination : Delhi", "Enter starting place : Ghaziabad", and "Enter fare of one ticket : 70".

```
"F:\college\APS minor\APS Project\BusTicketBooking.exe"

*****
1.ADD NEW BUS
2.REMOVE A BUS
3.SHOW ALL THE BUSES
4.TOTAL NUMBER OF BUSES
5.SEARCH BUS
6.Reservation of seats
7.Cancel a reservation
8.Daily profit
0.EXIT

Enter your choice here : 1

*****
Enter bus code : 123
Enter driver's name : Qwerty
Enter arrival time : 12:00
Enter departure time : 10:30
Enter destination : Delhi
Enter starting place : Ghaziabad
Enter fare of one ticket : 70

*****
```

Fig.2 Add New Bus

```
"F:\college\APS minor\APS Project\BusTicketBooking.exe"

*****

Bus code -> 123 Bus number -> 1 Driver's name -> Qwerty
Arrival time -> 12:00 Departure time -> %d%c%d10:30
From -> Ghaziabad To -> %sDelhi
Fare -> 70 Tickets sold -> 0
1.Empty 2.Empty 3.Empty 4.Empty
5.Empty 6.Empty 7.Empty 8.Empty
9.Empty 10.Empty 11.Empty 12.Empty
13.Empty 14.Empty 15.Empty 16.Empty
17.Empty 18.Empty 19.Empty 20.Empty
21.Empty 22.Empty 23.Empty 24.Empty
25.Empty 26.Empty 27.Empty 28.Empty
29.Empty 30.Empty 31.Empty 32.Empty

*****
```

Fig.3 Show all the Buses available

```
"F:\college\APS minor\APS Project\BusTicketBooking.exe"

Enter your choice here : 5

*****

Enter Bus number : 1

Bus code -> 123 Driver's name -> Qwerty
Arrival time -> 12:00 Departure time -> 10:30
From -> Ghaziabad To -> Delhi
Fare -> 70 Tickets sold -> %d0
1.Empty2.Empty3.Empty4.Empty
5.Empty6.Empty7.Empty8.Empty
9.Empty10.Empty11.Empty12.Empty
13.Empty14.Empty15.Empty16.Empty
17.Empty18.Empty19.Empty20.Empty
21.Empty22.Empty23.Empty24.Empty
25.Empty26.Empty27.Empty28.Empty
29.Empty30.Empty31.Empty32.Empty
*****
```

Fig.4 Searching Bus

"F:\college\APS minor\APS Project\BusTicketBooking.exe"

```
1.ADD NEW BUS
2.REMOVE A BUS
3.SHOW ALL THE BUSES
4.TOTAL NUMBER OF BUSES
5.SEARCH BUS
6.Reservation of seats
7.Cancel a reservation
8.Daily profit
0.EXIT

Enter your choice here : 6

*****
Enter bus number : 1
Enter seat number : 3
Enter the passenger's name : Mridnal

Seat successfully reserved!
Do you want to continue reservation ? : y
Enter bus number : 1
Enter seat number : 5
Enter the passenger's name : Pranati

Seat successfully reserved!
Do you want to continue reservation ? : y
Enter bus number : 2
Sorry, no bus found with the given Bus number.
Enter bus number : 1
Enter seat number : 2
Enter the passenger's name : Pranjal

Seat successfully reserved!
Do you want to continue reservation ? : n
```

Fig.5 Adding Passangers

"F:\college\APS minor\APS Project\BusTicketBooking.exe"

```
*****
1.ADD NEW BUS
2.REMOVE A BUS
3.SHOW ALL THE BUSES
4.TOTAL NUMBER OF BUSES
5.SEARCH BUS
6.Reservation of seats
7.Cancel a reservation
8.Daily profit
0.EXIT

Enter your choice here : 7

*****
Enter Bus number : 1
Enter seat number : 3
Successfully canceled reservation.

*****
```

Fig.6 Cancelling Reservation

Conclusion

Bus reservation system can be very helpful in managing data in the field of bus transport and efficient work can be done in office using the same.

References

<https://tutorialspoint.com/>

<https://www.geeksforgeeks.org/>