NEW HORIZON
COLLEGE OF ENGINEERING

A MINI PROJECT

REPORT

for
MINI PROJECT (19CSE39)

AIRLINE RESERVATION

submitted by

MANOJKUMAR RAJAVARAPU
1NH19CS138
3/C

In partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

NEW HORIZON
COLLEGE OF ENGINEERING

Golden Jubilee

Certificate

This is to certify that the mini project work titled

AIRLINE RESERVATION

Submitted in partial fulfillment of the degree of Bachelor of Engineering in Computer Science and Engineering

Submitted by

R.MANOJKUMAR

1NH19CS138

DURING

ODD SEMESTER 2020-2021

For

19CSE39

Signature of Reviewer Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner	*Signature with date*
1.____________________________	____________________________
2.____________________________	____________________________

Airline Reservation

ORIGINALITY REPORT

13%	%	13%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	. "Airline Reservation System", university degree/mathematical and computer sciences/computer science/1, 2013. Publication	13%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

ABSTRACT

Airline reservation System is a computerized system used to store and retrieve information and conduct transactions related to air travel. The project is aimed at exposing the relevance and importance of Airline Reservation . It is projected towards enhancing the relationship between customers and airline agencies through the use of ARs, and thereby making it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations.

Today all persons are busy with their schedule and no one has time to make a trip for holidays with their family. And this Airline Reservation Process is very difficult to understand in General meaning. But we are providing a Solution for that Problem. This system provides a facility to easy access towards customers and real time users. They can easily connect through it and just 3 steps. There is no requirement for any type of Agent. We are giving a all this facility in one project “Airline Reservation ”

Keywords : passport number , easy access , limited features

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani** , Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating a good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal NHCE, for his constant support and encouragement.

I would also like to thank Dr **. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Mr. Dinesh G** Sr.Assistant Professor my, project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

R.MANOJKUMAR(1NH19CS138)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1:

INTRODUCTION

1.1 PROBLEM DEFINITION

A small airline has just purchased a computer for its new automated reservation . The owner has asked to program the new creed in C. It is required to write a program to assign seats on each flight of the airline's only place. The program should never assign a seat which is already assigned. If there’s no seat available, then print the message " the flight is full " , set the corresponding elements of the array to 1.

1.2 OBJECTIVES

Simple Airline Seat Reservation System is based on the concept of reserving airline seats for the passengers. There’s no login available for this system, the user can freely use its feature. This mini project contains limited features, but the essential one. Make sure that you have a code editor for the C programming language. So, after that, open the project in the code editor and first build the project. After that, run the project.

Talking about the features of this Airline reservation , the user has to enter the data of the passenger when he selects the no.of seats he wantṣ. After the details have been entered as asked by the code, the user has to enter that particular seat number in order to reserve it.

1.3 METHODOLOGY TO BE FOLLOWED

This reservation gets done by having passengers' passport numberṣ. Which is done in a simple and easy handling process. Airline reservation is a computerized system used to store and retrieve information. The project is aimed at exposing the relevance and importance of Airline Reservation. It is projected towards enhancing the relationship between customers and airline agencies through the use of AR’s, and thereby making it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations. This project is about SIMPLE AIRLINE RESERVATION, which helps us to book the tickets according to the choice customer of priority towards the kind of seat.

1.4. EXPECTED OUTCOMES

```
.************************************* Welcome To airline reservation

************************************* **Enter your choice**


 1. RESERVE SEAT

 2. CANCEL TICKET

 3. DISPLAY PASSENGER DATA

 4. SEARCH PASSENGER DATA

 5. EXIT SYSTEM
```

1.5.HARDWARE REQUIREMENTS

Processor	: Any Processor above 500MHz
RAM	:512
Hard Disk	:10GB
Input device	: Standard Keyboard and Mouse
Output device	: VGA and High Resolution Monitor

1.5. SOFTWARE REQUIREMENTS

- Operating system: WindowsXP,7,8,10
- Front End: Turbo c

Turbo c: It is a discontinued Integrated Development Environment and compiler for the c programming language. It was introduced in 1987, it is very fast compile speed, comprehensive manual and low price. It was developed by Borland. Even in turbo c we can compile c ++ programming code. In these it is specially meant for learning the code for students. It is a software development tool for written programs in c language.it included a source code editor.

Turbo c features:

In line assembly with full access to the symbolic structures and names

Support for all memory modules Speed or size optimization the compiler could configure to produce an executable program was either fast or small in size, but not both.

CHAPTER 2:

DATASTRUCTURES

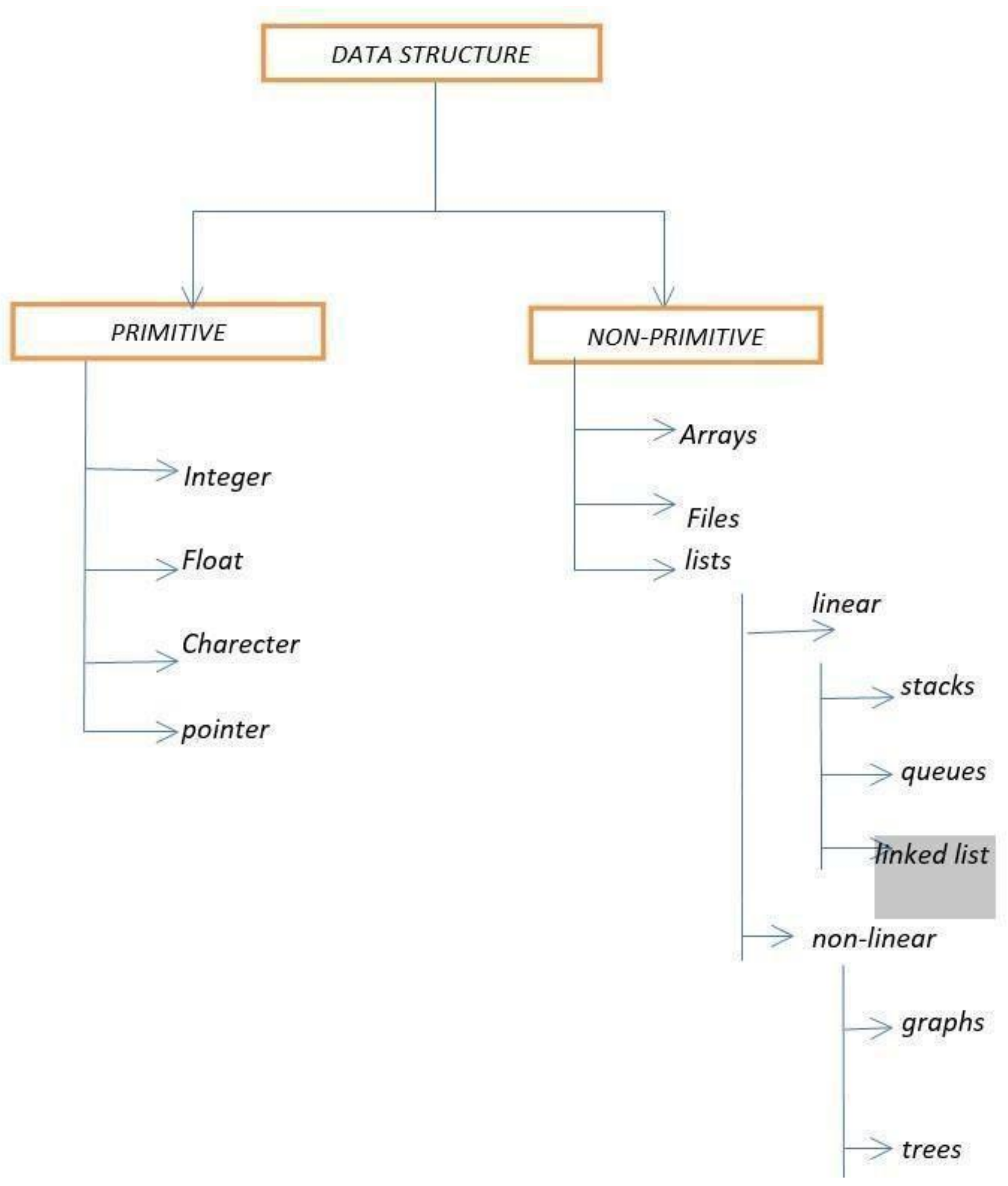


Fig no 2.1: Flowchart of data structure

DATA: Data is a collective of value. e.g., the data of the student includes his USN ,Name, Semester, Section, Address and phone number. Where address and name may have sub data like his/her first name, Middle name, Last name.

Data structure**:**

It is used for storing data in a format.

Data structures are of two types.

- Primitive data structures
- Non-primitive data structures.

- Primitive data structures can be directly manipulated by machine instructions. Examples :

  Integer

  i. Float

  ii. Char

  iii. Pointer

i.Integer: It is a whole number without decimal points; it consists of two bytes.

ii. Float: It is a decimal number where it consists of 4 bytes.

iii.Char: It consists of all the string variables and it consists of two bytes. *iv.* pointer: It gives the address of the other variable.

*2.*Non-primitive data structures can not be directly manipulated using machine instructions.

Examples :

- Arrays
- List
- File

1.Array: It is a sort of information structure where it is an assortment of comparative information type variables. An exhibit is an information type in C, which is built from basic information of the C language.

2.Syntax:

Data type Array-Name[index]; E.g.: -int A[10];

Char S[20]; float F[15];

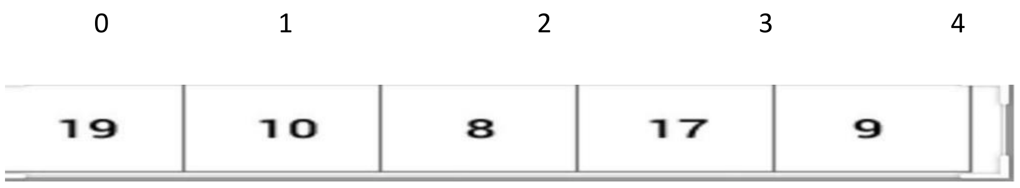


Fig 2.2: SYNTAX OF ARRAY

Types of array

- Single dimensional array
- Two dimensional array
- Multi dimensional array

• **Single dimensional array:** It is a collection of data items which can be stored under a variable name only by using one subscript

Syntax :

Data _type array _name[array_size_1] Ex: int a[5]

ii.Two dimensional array : Arrays with two sets of brackets [][] are called two dimensional arrays.

It can be used when the data items are arranged in row wise or in column wise in a tabular column.

Syntax :
Data _type array _name[array_size_1][array_size_2]
Ex : int a[5][5]

iii.Multi dimensional array : It is also called a three dimensional array where it is used for representing the total number of tables of matrix. It is used when we want to make the two or more tables of matrix elements for declaring the array elements we can use like this way. Syntax :

Data _type array _name[array_size_1][array_size_2] [array_size_3] Ex :
int a[5][5][5]

***3.* List :**It is a sequence of links which contains items. Each link is connected to one another link. Lists are again classified into

- Linear lists
- Non-linear lists.

1.Linear list consists of:

- Stack
- queues
- linked list.

2.Non-linear list consists of :

- trees
- graph.

We have 2 types of Memory Managements/Allocations

- Static Memory Allocation
- Dynamic Memory Allocation

1. **Static Memory Management:**If memory is allocated to the variables during compile time then it is called static memory allocation.
e.g. int a [10]; int *p;

Disadvantages:

I. Memory space is fixed during compilation time. i.e. it can’t be altered.

II. If all the memory allocated is not used then it leads to memory wastage.
III. Memory can't be increased later, if we require it to be.

2 . Dynamic Memory Management: It is the process of allocating memory during execution/run time. It uses predefined functions to allocate and release memory. Memory is allocated to the nodes using dynamic memory allocation functions such as malloc() , calloc () , realloc() and free().

i. **malloc() :**This function is used to allocate a complete single block of memory of the specified size. A pointer is used to store the address returned by malloc.
Syntax – data type*ptr=(datatype*)malloc(size)

ii. **calloc() :** It is a function which allocates a specified size of memory in multiple blocks of the same size. Each block should be assigned to null. A pointer is used to store the address.
Syntax – datatype *ptr =(datatype*)calloc(size, number of blocks)

iii. **realloc() :**For reallocating the allocated memory this function is used. A pointer is used to store the address returned

Syntax – data type*ptr=(datatype*)realloc(ptr,size) *iv.* **free() :**It is a function which is used to free the allocated memory. Syntax – free(pointer name)

4.File: It represents the sequence of bytes on disks where the group of data is stored.

2.1.STACK

A stack is a linear data structure where the elements are inserted and deleted from the same which follows last in first out (LIFO) .

Ex. Stack of books.
The basic primitive operation of stack is

1.push: Push inserting an element on the top of stack

2.pop: Pop deleting the top most element of the stack

Application of stack:

- Recursion
- Fibonacci series
- Tower of Hanoi problems
- Conversion of expressions
- Evaluation of postfix expression

  - **Recursion**: It is a programming technique which allows the programmer to express the operations in the term of c ,this takes the form of a function that calls itself by its own these is known as recursion.

  - **Fibonacci series :**we use loops as well as recursion in these series. Let's take the first few numbers seriously like 0,1,1,2,3,5,8. Where in these it tells that adding the first two numbers partially the next two elements see add 0 and 1 it gives 1 next add 1 and 1 it gives two. This is known as the Fibonacci series.

  - **Towers of Hanoi problems :**Towers of Hanoi is a puzzle where we have three rods and n disks .in these objective of the puzzle is to move the entire stack to another rod by obeying the following rule

i. only one disk can be moved at a time.

ii. Each disk is moved by taking the upper disk one of the stack and placing it on top of the other
. operation

iii. none of the big disk should be placed above the small disk.

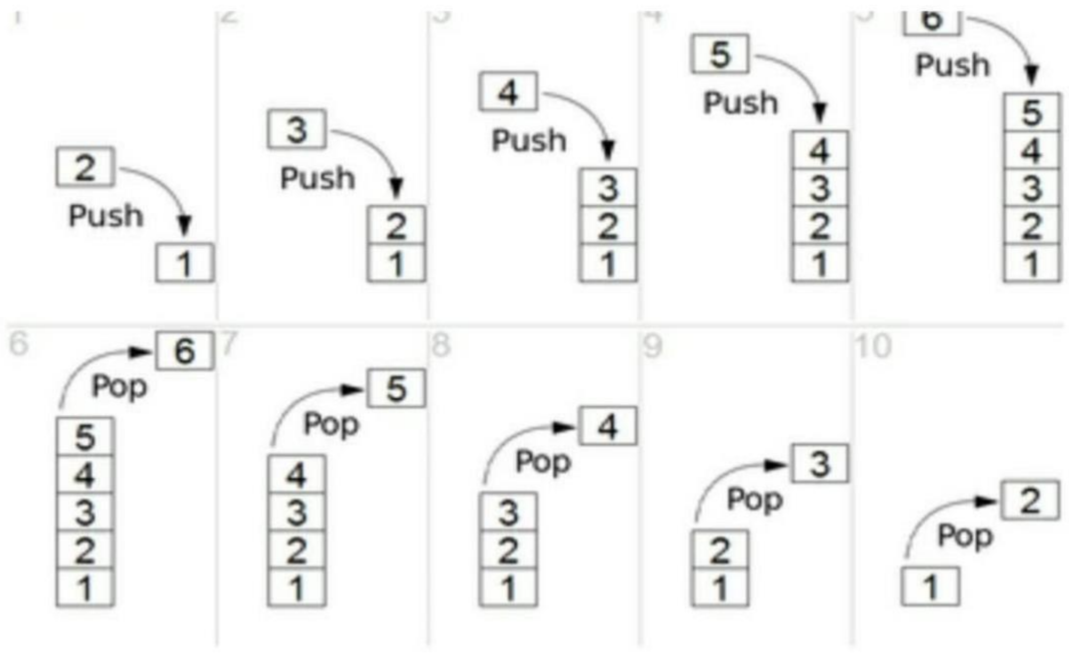


Fig no 2.3 : Stack operations

Advantages and disadvantages of stack

Advantages

- Easy to start
- Less hardware
- requirement It is a cross
- platform

Disadvantages

- Its not flexible
- it is lack of scalability
- its unable for copy and
- paste

2.2. QUEUES

A Queue is an ordered collection of data such that data is inserted at one end and deleted from another end. It follows first in first out(FIFO) order.

Ex: queue in cafeteria.

Queues can be represented using an array and linked list similar to stack.

Queue operation

- Enqueue
- Dequeue

- **Enqueue** : The addition of a new element to the queue is called insertion and every time before inserting a new item rear is incremented first then is inserted.

The condition to be checked in insertion is queue overflow, where reaches maximum size

- **Dequeue** : the removal of the front element from the queue is called deletion after deleting the element .We need to increment the front pointer.

The condition to be checked in deletion is queue underflow.

2.3. LINKED LIST

A linked list is a linear data structure , in which the elements are not sorted at contiguous memory locations , a linked list contains nodes where each node contains a data field and a reference (link) to the next node in the list. The first node is called head. If the linked list is empty, then the value of head is NULL.

Some types of linked list are

- Single linked list
- double linked list
- circular linked list
- header linked list

- **Single linked list:**-Each node contains only one pointer which will point to the next node. accessing from left to right is possible.
- **Double linked list :-**Each node contains two pointers one is the previous node and the other is the next node. Accessing from left to right and right to left is possible.
- **Circular linked list:** In any linked list, if the last node is to be pointed back to the first node then it is called circular linked list. It is of two types

1. Circular linked lists
2. Circular doubly linked lists

In double linked lists we have two-way access through prev>Which is left pointer and
Next>Which is right pointer

ADVANTAGES OF CIRCULAR LINKED LISTS:
Any node can be accepted from any other node including the first node and last node

Insertion: - Insertion in singly linked list: Insertion is possible at 3 positions.

a) At beginning
b) At ending
c) At any position.

Deletion: -Deletion a node in circular single linked list is possible at 3 positions.

a) Deleting first node
b) Deleting last node
c) Deleting the required node.

Display: We can traverse a list from left to right (or) right to left.

iv Header linked list:- header linked list is one or more types of linked list. In these types, we have a special node created in the linked list. This special node is used to store some useful information about the linked list.

Application of linked list

- Josephus problem
- Addition of two long integer
- stacks using linked list
- queue using linked list

● **Josephus problem :** In these it tells about the problem in a circle where at each count it skips the count and gain start from the beginning later the last one remaining will be the winner

● **Addition of two long integers :** Basically we can add too many long numbers present in the list hence we use linked lists to add more numbers.

● **stack using linked list :** Every new element is inserted as top element .That means every newly inserted element is pointed on top.

● **Queue using linked list :** In these it consists of two parts one is data part and the other is the linked part where each element of the queue is pointed to its immediate next element in the memory.

LINKED REPRESENTATION OF STACK:

- Disadvantage of representing stack using arrays is wastage of memory. This can be overcome by using linked representation.

- Since stack follows LIFO order, we should put only insertion at ending and deletion at ending cases of linked lists to represent push and pop operations respectively.

LINKED REPRESENTATION OF QUEUE:

Since Queue follows FIFO order, we should put insertion at ending and deletion at beginning cases of linked lists.
Typedef: - typedef is a keyword, which allows us to make a variable as a data type.
e.g. if we give typedef struct list node
the normal variable node has become a data type called struct list. So to declare a pointer variable struct, we can use node *struct; instead of struct list *start

MAIN ADVANTAGES OF
LINKED LIST OVER ARRAYS IS :

1. Size of the array is fixed ,we must know its upper limit in advance. But the linked list size is not fixed.
2. Insertion and deletion is easy compared to array.
3. No memory wastage will be there in linked list

1. DISADVANTAGES OF LINKED LISTS:

There were so many problems with linked lists .The major disadvantage of linked lists. lists are access time to individual elements .Array is randomly accessed memory ,which means it takes to access any element in the array .Linked lists take for access to an element. In the list in the worst case. Another disadvantage of arrays in access time is special locality in memory .Arrays are defined as contiguous blocks of memory and lots of array elements will be temporarily near its neighbouring nodes .This greatly benefits from the latest CPU holding methods.

Although the dynamic allocation of storage is a great advantage ,the overhead with storing and retrieving data can make a big difference .Sometimes linked lists are hard for manipulation. If the last item is deleted, the last but one must have its pointer changed to hold a NULL pointer. This required that the linked list is traversing to find the last before link, and this pointer set to a null pointer

Structure of single linked list :

HEAD

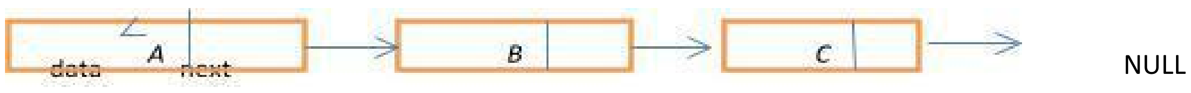


fig no 2.4 : STRUCTURE OF SINGLE LINKED LIST

The self referential code I have used is:

1.Single linked list linked list

```
"Struct node

{

Char data[20]; Char

m[][]; int count; struct

node*link

};

Struct node*find[]; "
```

Structure for doubly linked list

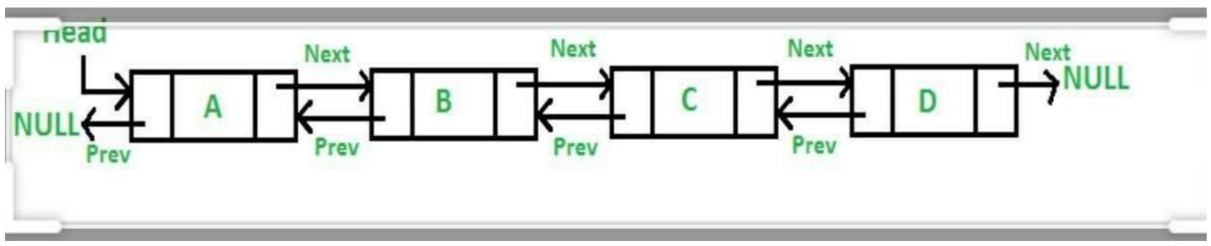


Fig no 2.5 : Structure for doubly linked list

2.Double linked list

“Struct d list

{ int data; struct d list *p
rev ; struct d list*next;

}”

2.4. TREES

It is non linear data structures ,compared to array ,linked list, stack which are linear data structures .A tree can be empty with no nodes or a tree is a mode consisting of one tree.

Types of tree:

- *Binary tree*
- *Full binary tree*
- *Strictly binary tree*
- *Complete binary tree*
- *Binarysearch tree*
- *Threaded binary tree*
- *Expression tree*
- *Heap tree*

- Binary tree: It is a tree which has 2 child nodes.
- Full binary tree: It's a tree in which every node should have two child nodes and 2 child nodes except the leaf node.
- Strictly binary tree: It is a tree which has 0 or 2 child nodes is called strictly a binary tree.
- Complete binary tree: It is a full binary tree till n-1 level. But at the last level all the nodes are filled from left to right.
- Binary search tree: It is a binary tree in which all the elements off the left subtree are less than the root node and all the elements in the right subtree are greater than the root node.
- Expression binary tree: It is a binary tree on which the internal nodes represents operation and external nodes represents operands

Three types of expression binary trees are

I. Infix expression tree
II. Postfix expression tree
III. Prefix expression tree

- Complete binary tree: It is a full binary tree till n-1 level. But at the last level all the nodes are filled from left to right.

- Binary search tree: It is a binary tree in which all the elements off the left subtree are less than the root node and all the elements in the right subtree are greater than the root node.

- Expression binary tree: It is a binary tree on which the internal nodes represents operation and external nodes represents operands

Three types of expression binary trees are

I. Infix expression tree
II. Postfix expression tree
III. Prefix expression tree

- Threaded binary tree: It is a binary tree in which all the null pointers are replaced with its In order pre disorder and successor address.

- Heap Tree: It is constructed on the heap property .There are two types of heap tree.

  i. Max heap tree
  ii. Min heap tree.

I. Max heap tree : It is a fully binary tree data structure. In max heap tree nodes are arranged based on node value. Max heap tree is specialized full binary tree in which every parent node contains greater or equal value than child nodes

II. Min heap tree : It is a full binary tree data structure. In min heap tree nodes are arranged based on the node value. Min heap tree is specialized with a fully binary tree in which every parent node contains less than or equal to either of its children nodes.

Tree Traversals

i. In order ii. Pre order iii .post order

i. In order : It follows first the left node of the sub tree than its root node, later its right node.

ii. Pre order : It follows first the root node than left node than the right node.
iii. Post order : It follows first the left node than the right node and later the root node

APPLICATIONS OF BINARY SEARCH TREE

Applications of BST Binary Search Tree, is a hub based doubletree information structure which has the accompanying properties: The left subtree of a hub contains just hubs with keys lesser than the hub's critical.
The privilege subtree of a hub contains just hubs with keys more noteworthy than the hub's critical.
The left and right subtree each must likewise be a paired inquiry tree.
There must be no copy hubs.

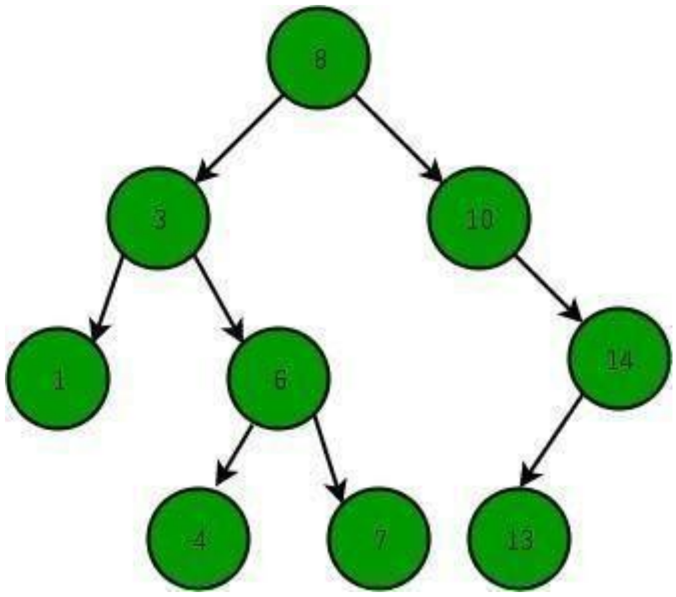


Fig no 2.6 : Binary Search Tree

A BST bolsters tasks like search, embed, erase, floor, ceil, more prominent, littler, and so on in O(h) time where h is tallness of the BST. To keep tallness less, self adjusting BSTs (like AVL and Red Black Trees) are utilized practically speaking. These Self-Balancing BSTs keep up the stature as O (Log n). In this way the entirety of the previously mentioned tasks become O (Log n). Together with these, BST likewise permits arranged request traversal of information in O(n) time.

A Self-Balancing Binary Search Tree is utilized to keep up an arranged stream of information. For instance, assume we are getting on the web requests set and we need to keep up the live information (in RAM) in arranged requests of costs. For instance, we wish to know the number of things bought at cost underneath a given expense at any minute. Or on the other hand we wish to know the number of things acquired at greater expense than given expense.

A Self-Balancing Binary Search Tree is utilized to execute doubly finished need lines. With a Binary Heap, we can either execute a need line with help of extract Min() or with extract Max(). On the off chance that we wish to help both the tasks, we utilize a Self-Balancing Binary Search Tree to do both in O (Log n)

There are a lot more calculation issues where a Self-Balancing BST is the most appropriate information structure, similar to Count Smaller Elements on Right, Smallest Greater Element On Right Side, as so on.

2.5.GRAPHS

It is a non linear data structure. It is a pictorial representation of a set of objects where some pairs are connected into links.

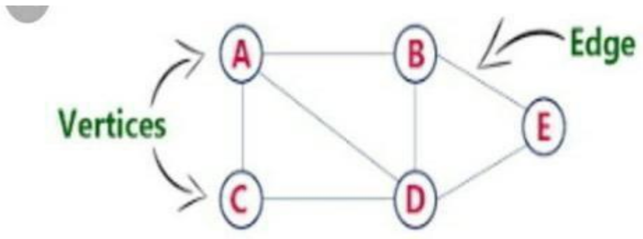


Fig 2.7 Graph Representation

Different types of representing the graph:

2.3.1. Adjacency matrix.

2.3.2. Adjacency linked list.

3. Adjacency matrix : It is a square matrix used to represent a finite graph. The elements of the matrix indicate whether pairs of vertices are adjacent or not in the graph.

4. Adjacency linked list : In an adjacency linked list is a collection of unordered lists used to represent a finite graph. Each linked list describes the set of neighbours of a vertex in the graph.

Types of traversal methods:

- Breadth first search
- Depth first search

1.**Breadth first search** : It is searching a tree or graph data structures. It starts at the tree root, and explores all the neighbour nodes at the present depth prior to moving on to the nodes at the next depth level.

2.**Depth first search :** It is searching a tree or graph data structures. The algorithm starts at the root node and explores as far as possible along each branch before brackets.

CHAPTER 3:

DESIGN

3.1 ALGORITHM/PSEUDOCODE:

Step1 : Start

Step2 : Displays the options to be chosen ,as follows

Step3 : As continuing with reservation ,asks to insert the passenger details , continuous with information.

Step4 : Once the insertion and information inserting is done, returns to the main display and asks for our choice

Step5 : Choosing option 2 cancel , the options are displayed and by choosing the option we can cancel the ticket.

Step6 : Again displays the main display and waits for our choice, as we continue with display option.3 , which displays all the data along with passenger details.

Step7 : Again displays the main display and waits for our choice, as we continue with display option.4,searching the passenger data, by using passport number,name.,etc., displays the details of the passenger.

Step8 : Again displays the main display and waits for our choice, as we continue with display option.5, we are exiting from the program.

IMPLEMENTATION

4.1. MODULE1 FUNCTIONALITY

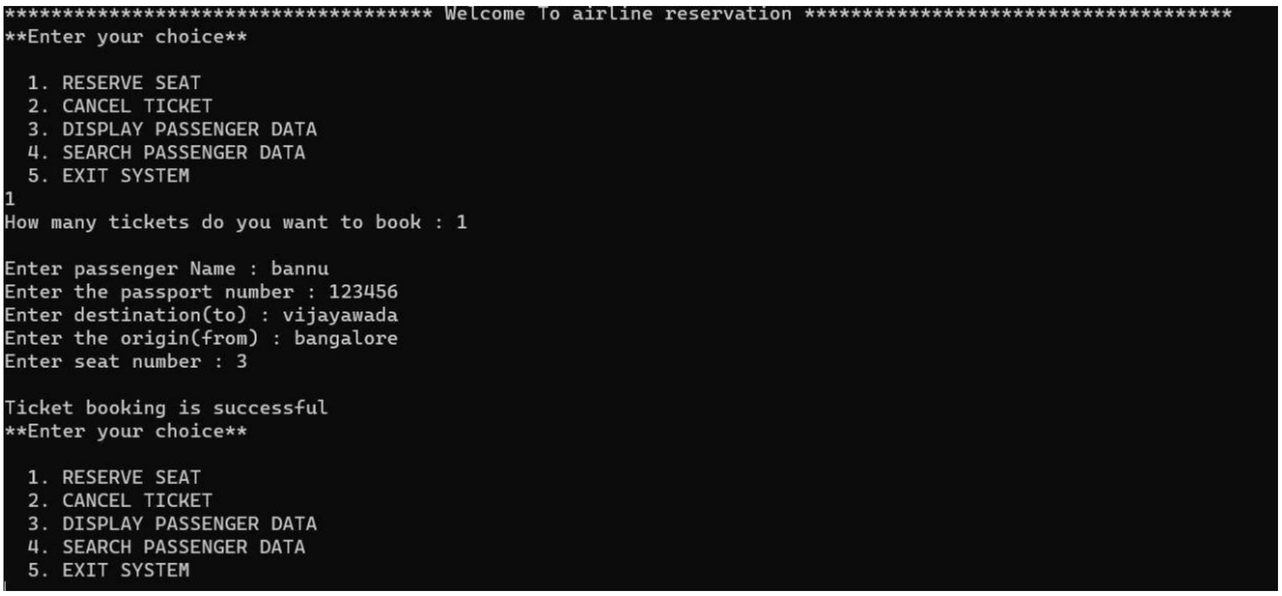
Insertion

```
void reserve()
{

    // printf("\n");
    printf("How many tickets do you want to book : ");
    scanf("%d", &n);
    fflush(stdin);
    sum = n + num;
    for (i = num, j = 0; i < sum; i++)
    {
        printf("\n");
        fflush(stdin);
        fflush(stdin);
        fflush(stdin);
        printf("Enter passenger Name : ");
        getchar();
        gets(x[i].name);
        fflush(stdin);
        printf("Enter the passport number : ");
        gets(x[i].passport);
        fflush(stdin);
        printf("Enter destination(to) : ");
        gets(x[i].destination);
        fflush(stdin);
        printf("Enter the origin(from) : ");
        gets(x[i].origin);
        fflush(stdin);
        printf("Enter seat number : ");
        gets(x[i].seatNum);
        fflush(stdin);
        printf("\n");
        j++;
        a++;
        num++;
    }
    printf("Ticket booking is successful");
    return;
};

void cancel()
{
    int f = 0;
    char s[50];
    printf("Enter the your passport number : ");
    getchar();
    gets(s);
    fflush(stdin);
    while(s==x[f].passport)
        {
            strcpy(x[f].name,x[f+1].name);
            strcpy(x[f].destination,x[f+1].destination);
            strcpy(x[f].origin,x[f+1].origin);
            strcpy(x[f].passport,x[f+1].passport);
            strcpy(x[f].seatNum,x[f+1].seatNum);
            f++;
            printf("Ticket has been cancelled\n");
        }
        num--;
        writeData();
        return;
};
```

OUTPUT



```
************************************* Welcome To airline reservation *************************************
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
1
How many tickets do you want to book : 1

Enter passenger Name : bannu
Enter the passport number : 123456
Enter destination(to) : vijayawada
Enter the origin(from) : bangalore
Enter seat number : 3

Ticket booking is successful
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
```

Fig : 4.1: INSERTION FUNCTIONALITY

4.2. MODULE 2 FUNCTIONALITY

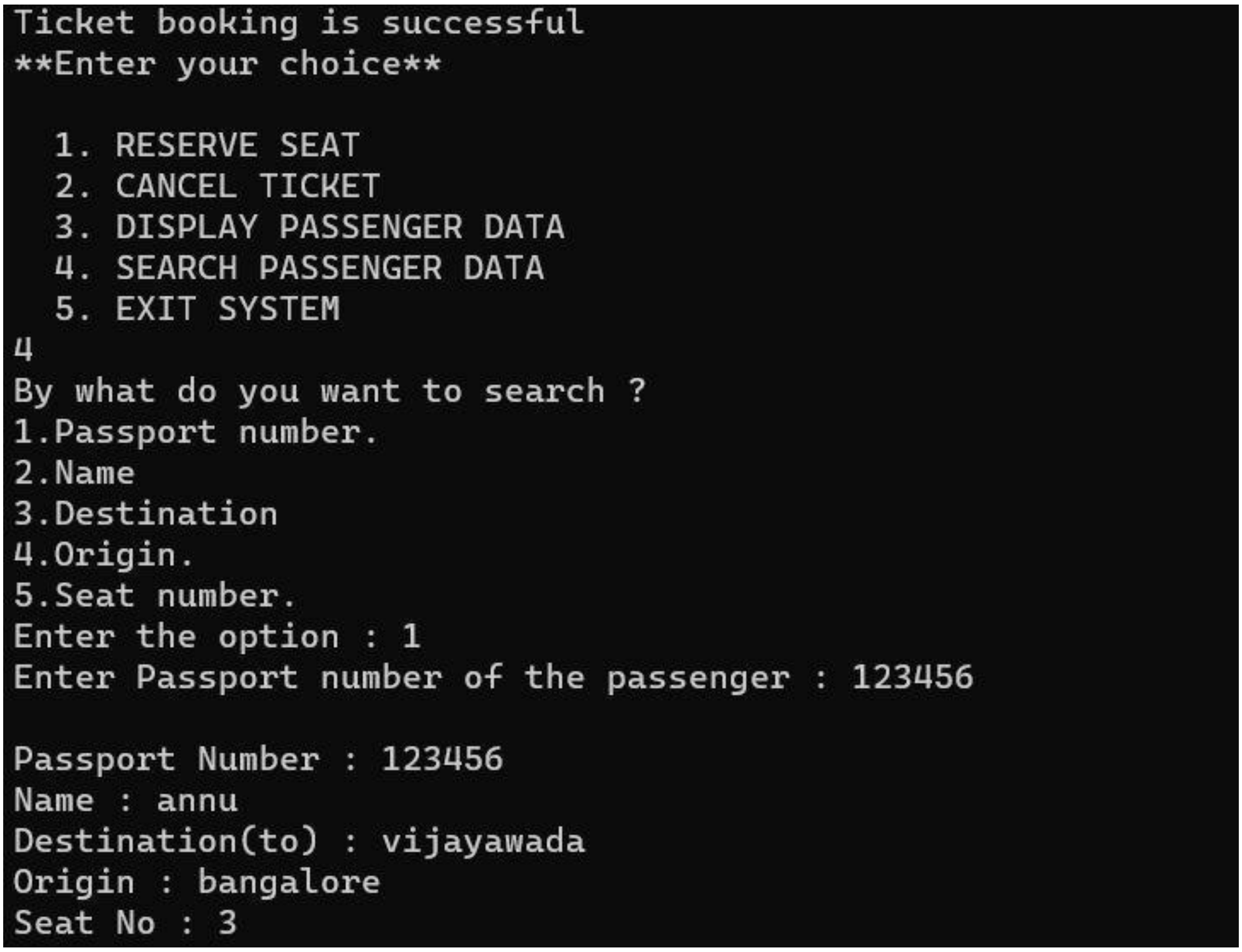
Searching

```
void search()
{
   int h, f;
   char u[100];
   printf("By what do you want to search ?\n");
   printf("1.Passport number.\n2.Name\n3.Destination\n4.Origin.\n5.Seat number.\n");
   printf("Enter the option : ");
   scanf("%d", &h);
   if (h == 1)
   {
      printf("Enter Passport number of the passenger :
      "); getchar();
      gets(u);
      fflush(stdin);
      for (g = 0; g < num; g++)
      {
         if (strcmp(u, x[g].passport)==0)
         {
            printf("\n");
            printf("Passport Number : ");
            puts(x[g].passport);
            printf("Name : ");
            puts(x[g].name);
            printf("Destination(to) : ");
```

```
                puts(x[g].destination);
                printf("Origin : ");
                puts(x[g].origin);
                printf("Seat No : ");
                puts(x[g].seatNum);
                printf("\n");
                return;
            }
            else
                printf("Not Found\n");
        }
    }
    else if (h == 2)
    {
        fflush(stdin);
        printf("Enter passenger name : ");
        getchar();
        gets(u);
        fflush(stdin);
        for (g = 0; g < num; g++)
        {
            if (!strcmp(u,x[g].name))
            {
                printf("\n");
                printf("Passport Number : ");
                puts(x[g].passport);
                printf("Passenger Name : ");
                puts(x[g].name);
                printf("Destination(to) : ");
                puts(x[g].destination);
                printf("Origin(from) : ");
                puts(x[g].origin);
                printf("Seat Num:");
                puts(x[g].seatNum);
                printf("\n");
                return;
            }
            else
            {
                printf("\nNot Found...\n");
            }

        }
                // printf("\nNot Found\n");
    }
    else if (h == 3)
    {
        printf("Enter destination(to) : ");
        getchar();
        gets(u);
        for (g = 0; g < num; g++)
        {
            if (!strcmp(u, x[g].destination))
            {
                printf("\n");
                printf("Passport Number : ");
                puts(x[g].passport);
                printf("Passenger Name : ");
                puts(x[g].name);
                printf("Destination(to) : ");
                puts(x[g].destination);
                printf("Origin(from) : ");
                puts(x[g].origin);
                printf("Seat Num:");
                puts(x[g].seatNum);
                printf("\n");
                return;
            }
        }
        printf("Not Found\n");
    }
    else if (h == 4)
    {
        printf("Enter Origin(from) : ");
        getchar();
        gets(u);
        for (g = 0; g < num; g++)
        {
            if (!strcmp(u, x[g].origin))
            {
                printf("\n");
                printf("Passport Number : ");
                puts(x[g].passport);
                printf("Passenger Name : ");
                puts(x[g].name);
                printf("Destination(to) : ");
                puts(x[g].destination);
                printf("Origin(from) : ");
                puts(x[g].origin);
                printf("Seat Num:");
                puts(x[g].seatNum);
                printf("\n");
                return;
            }
        }
        printf("Not Found\n");
    }
    else if (h == 5)
    {
        printf("Enter Seat number : ");
        getchar();
        gets(u);
        fflush(stdin);
        for (g = 0; g < num; g++)
        {
            if (!strcmp(u, x[g].seatNum))
            {
                printf("\n");
                printf("Passport Number : ");
                puts(x[g].passport);
                printf("Passenger Name : ");
                puts(x[g].name);
                printf("Destination(to) : ");
                puts(x[g].destination);
                printf("Origin(from) : ");
                puts(x[g].origin);
                printf("Seat Num:");
                puts(x[g].seatNum);
                printf("\n");
                return;
            }
        }
        printf("Not Found\n");
    }
    else
        printf("\n\nInvalid input\n\n");
};
```

OUTPUT



```
Ticket booking is successful
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
4
By what do you want to search ?
1.Passport number.
2.Name
3.Destination
4.Origin.
5.Seat number.
Enter the option : 1
Enter Passport number of the passenger : 123456

Passport Number : 123456
Name : annu
Destination(to) : vijayawada
Origin : bangalore
Seat No : 3
```

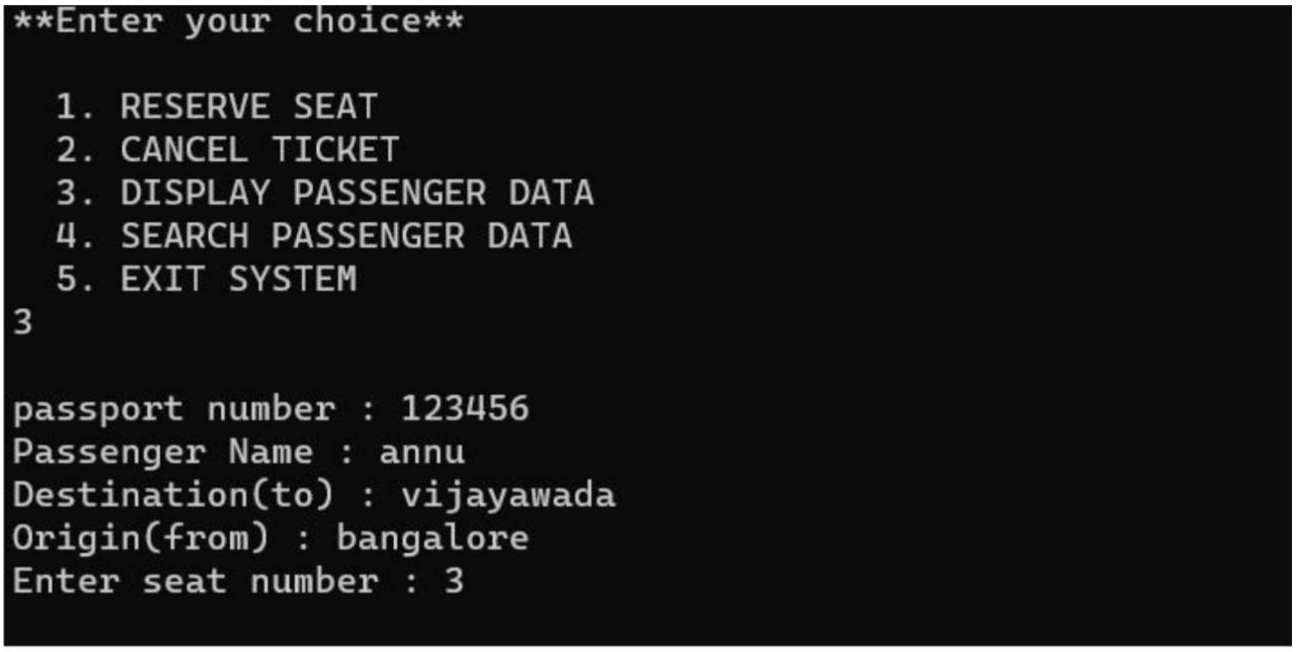
Fig - 4.2: SEARCHING FUNCTIONALITY

4.3. MODULE 3 FUNCTIONALITY

Display

```
void display()
{
   if(num == 0)
   {
       printf("No records found");
   }
   for(i=0; i<num; i++)
   {
       printf("\n");
       printf("passport number : ");
       puts(x[i].passport);
       printf("Passenger Name : ");
       puts(x[i].name);
       printf("Destination(to) : ");
       puts(x[i].destination);
       printf("Origin(from) : ");
       puts(x[i].origin);
       printf("Enter seat number : ");
       puts(x[i].seatNum);
       printf("\n");
   }
}
```

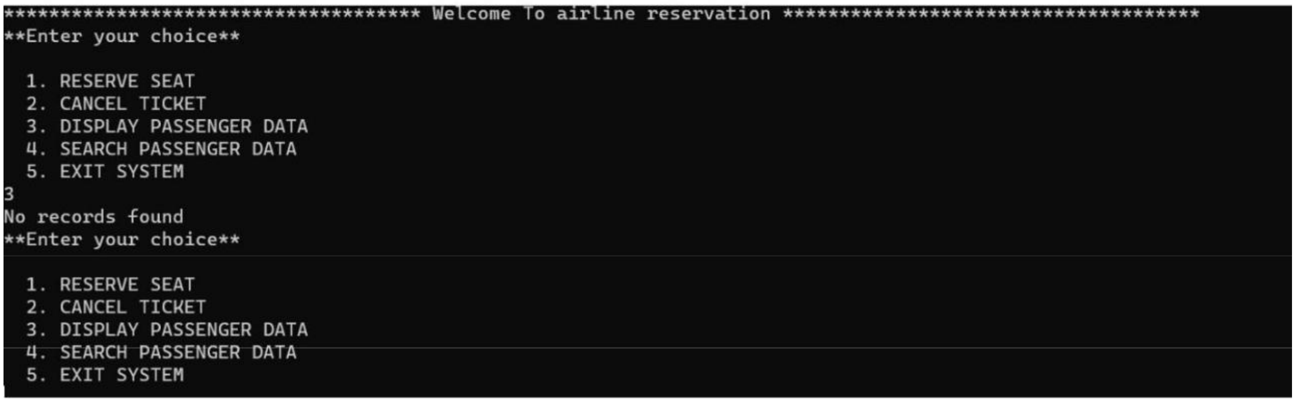
OUTPUT



```
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
3

passport number : 123456
Passenger Name : annu
Destination(to) : vijayawada
Origin(from) : bangalore
Enter seat number : 3
```



```
************************************* Welcome To airline reservation **************************************
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
3
No records found
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
```

Fig - 4.3: DISPLAY FUNCTIONALITY

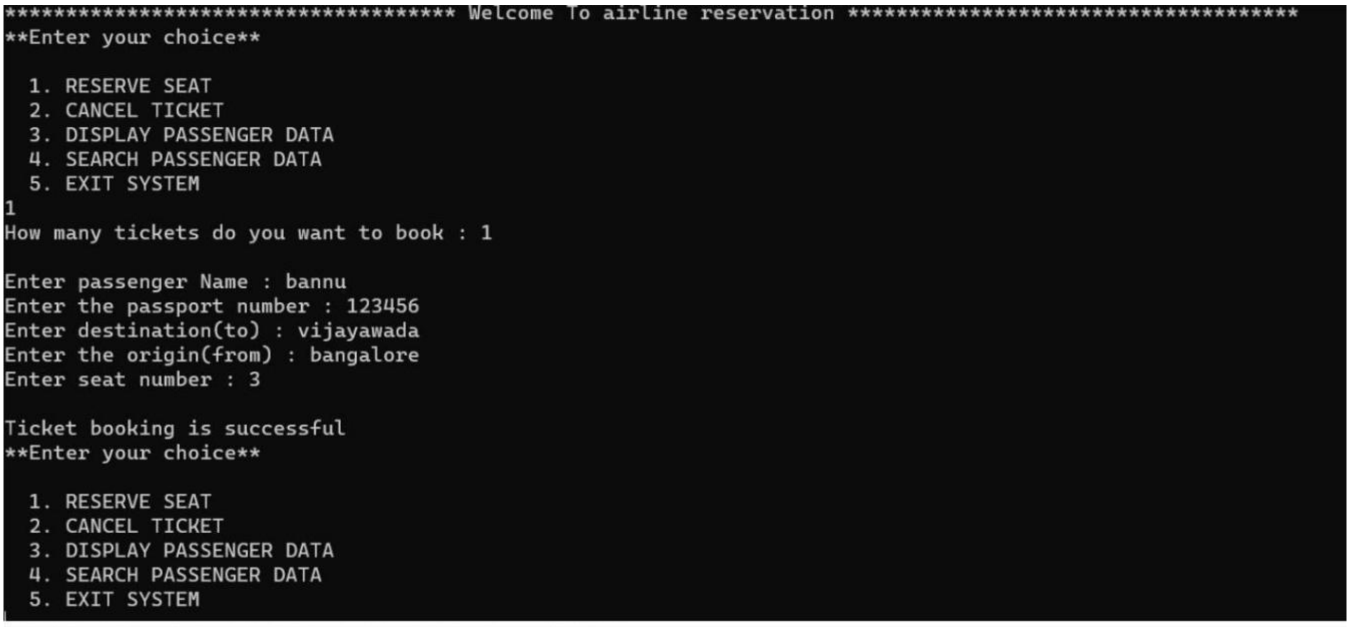
RESULT

By using this code we can finally book the tickets, cancel the tickets we have booked , by choosing the above options we can display the booked tickets and change the infomation.

Talking about the features of this Simple system, the user has to select the seat class whether to choose Business or Economy class. Then the system displays available seats, and the user has to enter that particular kind of seating order to reserve it. After reserving a seat, that particular seat won't be available anymore. The system does not create an external file to store the user's data permanently..

Simple Airline Seat Reservation System is based on the concept of reserving airline seats for the passengers. There's no login system available for this system, the user can freely use its feature. This mini project contains limited features, but the essential one.

OUTCOMES



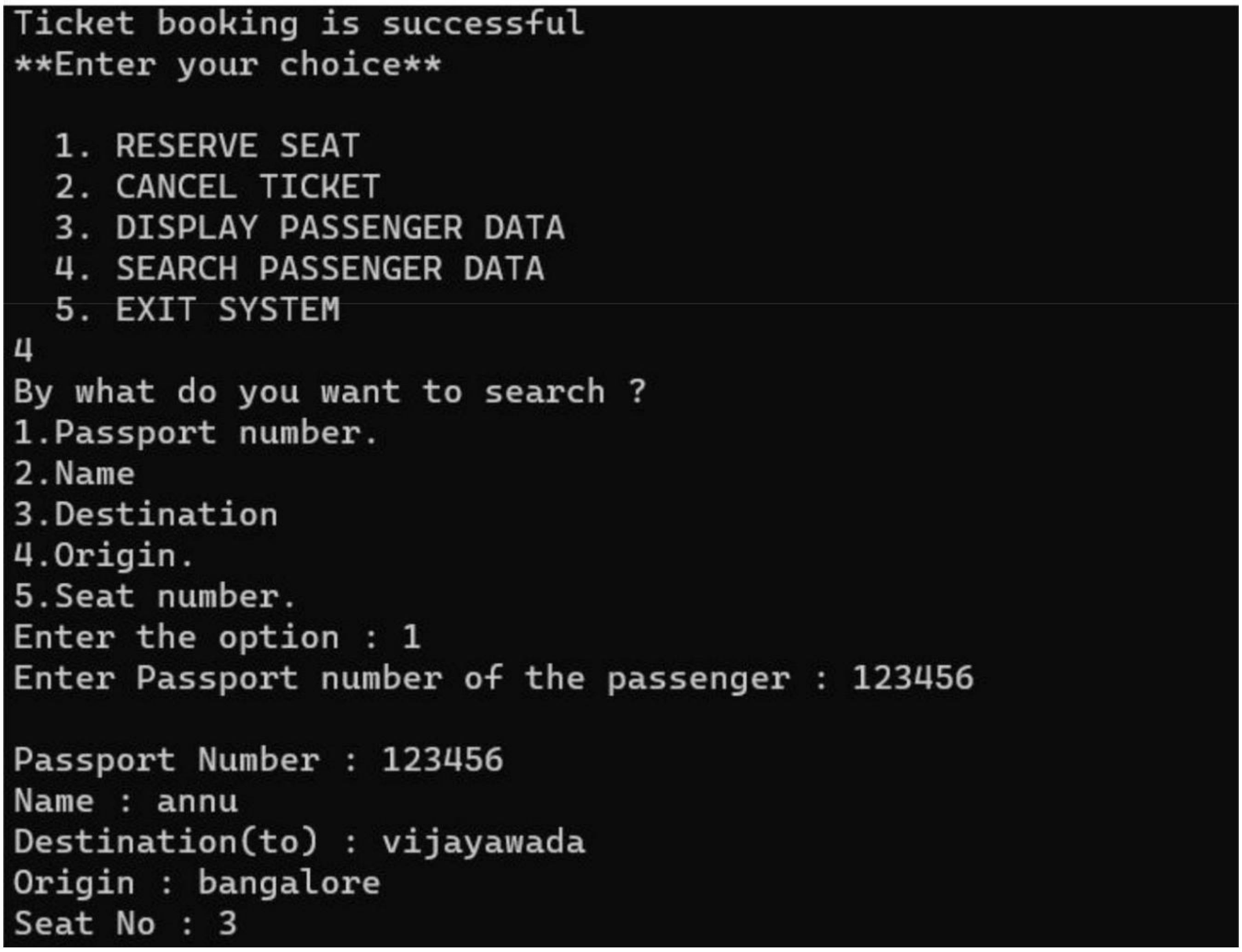
```
************************************* Welcome To airline reservation *************************************
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
1
How many tickets do you want to book : 1

Enter passenger Name : bannu
Enter the passport number : 123456
Enter destination(to) : vijayawada
Enter the origin(from) : bangalore
Enter seat number : 3

Ticket booking is successful
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
```

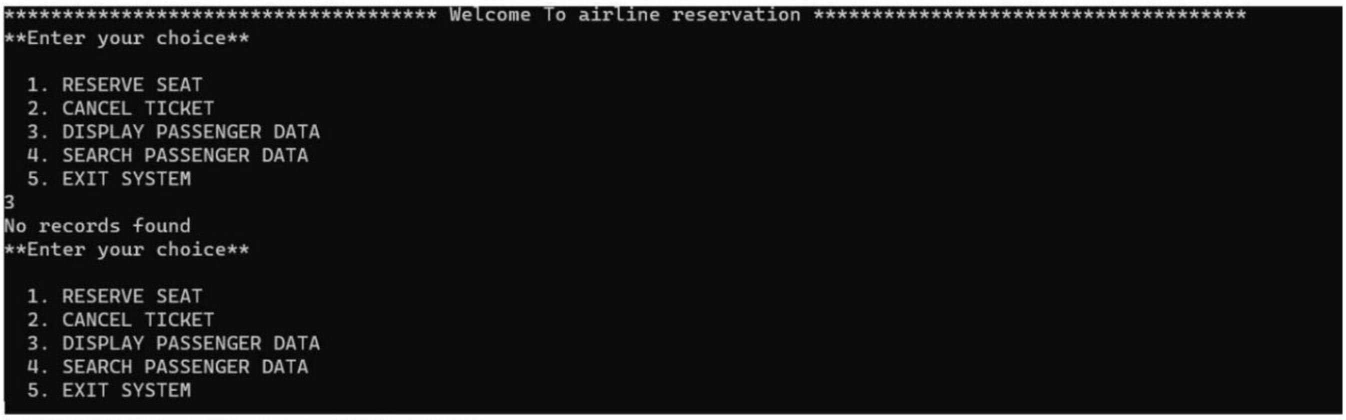


```
Ticket booking is successful
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
4
By what do you want to search ?
1.Passport number.
2.Name
3.Destination
4.Origin.
5.Seat number.
Enter the option : 1
Enter Passport number of the passenger : 123456

Passport Number : 123456
Name : annu
Destination(to) : vijayawada
Origin : bangalore
Seat No : 3
```

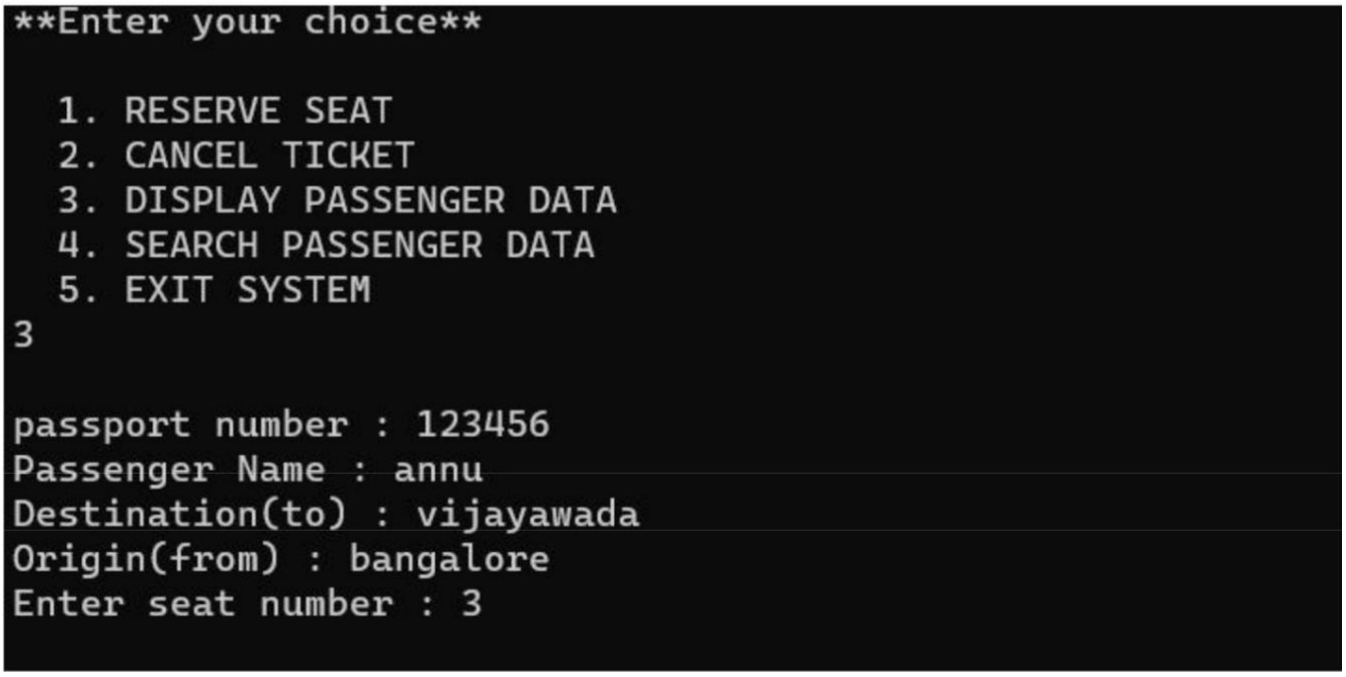
Fig - 5.1: OUTCOMES



```
************************************ Welcome To airline reservation ************************************
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
3
No records found
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
```



```
**Enter your choice**

  1. RESERVE SEAT
  2. CANCEL TICKET
  3. DISPLAY PASSENGER DATA
  4. SEARCH PASSENGER DATA
  5. EXIT SYSTEM
3

passport number : 123456
Passenger Name : annu
Destination(to) : vijayawada
Origin(from) : bangalore
Enter seat number : 3
```

Fig- 5.2: OUTCOMES

CONCLUSION

This reservation gets done by having passengers' passport number . Which is done in a simple and easy handling process. Airline reservation is a computerized system used to store and retrieve information.

The project is aimed at exposing the relevance and importance of Airline Reservation. It is projected towards enhancing the relationship between customers and airline agencies through the use of AR’s, and thereby making it convenient for the customers to book the flights as when they require such that they can utilize this software to make reservations. This project is about AIRLINE RESERVATION, which helps us to book the tickets according to the choice customer of priority towards the kind of seat.

REFERENCES

1. COLLEGE BOOKS
2. GEEKS FOR GEEKS
3. ONLINE C PROJECTS