CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION:

PURPOSE

The purpose of this source is to describe the railway reservation process, which provides the train timing details, reservation, and cancellation . In addition, This Railway reservation service will not only enhance the reservation but will also help the commuters in getting support

- confirm Reservation for confirm Seat.
- Reservation against Cancellation.
- Waiting list Reservation.

SCOPE

Technology is changed in many ways of life in the 21st century, including boking a train ticket. For example, to make booking more easy for passengers, an online booking processs helps us in booking tickets from our homes aswellas offices also . While this is easy for most people, it has made things particularly easier for people living in remote areas. The various advantages of using the online booking process are as follows: Convenient – You can book or cancel your tickets sitting in your home or office.

Saves Time and Effort

You can save the time needed to travel to the railway reservation office and waiting in the queue for your turn. Towards a greener planet – Instead of printing your ticket you can also choose to travel with the soft copy of your booked ticket in your laptop or even on your mobiles.

1.2. OBJECTIVES

- Provides the searching facilities based on various factors: train, seat ,enquiry.
- Manage the information of booking.
- Mange the information of seats.
- When the seats are filled the passenger is kept in to waiting list and when there is availability the ticket is automatically filled.
- The main objective of this train ticket booking project is to reduce consumption of time for booking ticket.

1.3. METHODOLOGY TO BE FOLLOWED

- The various methods used in this project are
    1. Linked list in data structures.
    2. Dynamic memory allocation.
    3. Queues in some minute part of the code.
    4. Declaring, Defining and calling different user defined functions what all required to use in the program.

1.4. EXPECTED OUTCOMES

- Creating the user.
- Registration or booking of ticket.
- If vacencies are there ticket will be confirmed else will sent to waiting list
- Cancelation of ticket is also done if they wants to cancel the ticket.
- Enquiry of details regarding passenger ticket is done.
- Finally display of details regarding the train ticket.

1.5. HARDWARE AND SOFTWARE REQUIRMENTS

Minimum Hardware Requirements

- Processor
- 500MB storage
- RAM 3GB

Preferred Hardware Requirements

- Processor Core i5
- 1TB storage
- 6 GB RAM
- Cache 512kb

Software requirements

For user: Windows Operations System with DOS Support for developing code. Windows 10, having turbo c++ Installed with a working LAN connection to be mandatory.

For Using: windows Operating System with at Least google chrome Installed and having minimum of working LAN compulsorily.

CHAPTER 2

DATASTRUCTURES

The data structure that I have used in this project is linked list

In computer science, a data structure is a data organization, management, and storage format that enables efficient access and modification. More precisely, a data structure is a collection of data values, the relationships among them, and the functions or operations that can be applied to the data.

Types of data structures

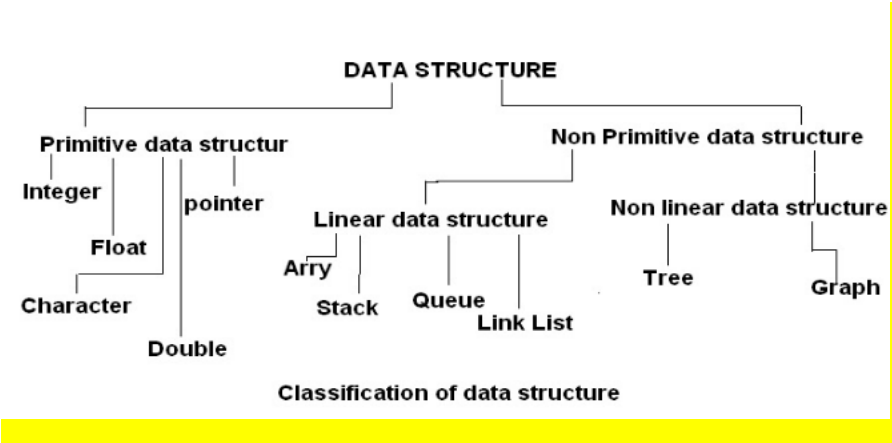


fig 1: classification of data structures

PRIMITIVE DATA STRUCTURE

Primitive data structures are those which are predefined way of storing data by the system. ... Primitive data structures are char, int, float, double. Characters are internally considered as int and floats also falls under double and the predefined operations are addition, subtraction, etc.

NON-PRIMITIVE DATA STRUCTURE

The data type that are derived from primary data types are known as non-primitive data type.

The non-primitive data types are used to store the group of values. Examples of non-primitive data type. Array, structure, union, link list, stacks, queue etc.

The data structure that I have used is non-primitive data structure.

LINKED LIST

INTRODUCTION

The everyday usage of the term “list” refers to a linear collection of data items.

Data processing frequently involves storing and processing data in to lists. One way to store such data is by means of arrays. Recall that linear relationship between the data elements of an array is reflected by the physical relationship of the data in memory. not by any information contained in data elements them selves. This makes it easy to compute the address of an element in an array. On the other hand, arrays have certain disadvantages example-it is relatively expensive to insert and delete in an array. Also, since an array usually occupies a block of memory space.

Another way of storing a list in memory is to have each element in the list contain a field, called a link or pointer, which contains address of next element in the list. Thus successive elements in the list need not occupy adjacent space in the memory. This will make it easier to insert and delete element in the list. Accordingly, if one were mainly interested in searching through data for inserting and deleting, as in word processing, one would not store the data in an array but rather in a list using pointers. This latter type of data structure is called a linked list

DEFINITION

A linked list, or one-way list, is a linear collection of data elements called nodes. where the linear order is given by means of pointers. That is, each node is divided in to two parts: the first part contains the information of the element and second part, called the link field or next pointer field, contains the address of the next node in the list

What are the drawbacks of using sequential storage to represent stacks and queues?

One major drawback is that a fixed amount of storage remains allocated to the stack or queue

even when the structure is using a smaller amount or possibility no storage at all. Further, no more than that fixed amount of storage may be allocated thus introducing the possibility of overflow

unfortunately, although such a scheme allows two stacks to share a common area, no such simple solution exist for three or more stacks or even for two queues. Instead, one must keep the tracks of the tops and bottoms (or front and rare) of all the structure sharing a single large array.

Each time that a growth of structure sharing single large array. each time that the growth of one structure is about to impinge on the storage currently being used by another, neighboring structures must be shifted within the single array to allow for the growth.

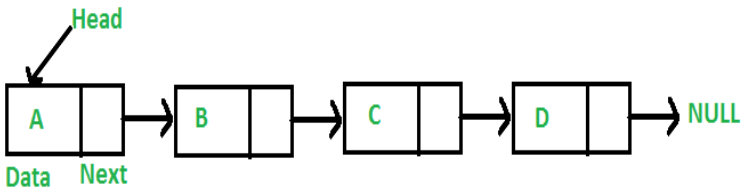


fig 2: Representation of linked list

Inserting and removing nodes from a list

A list is a dynamic data structure. The number of nodes on a list may vary dramatically as elements are inserted and removed. The dynamic nature of a list may be constrained with the static nature of an array, whose size remains constant.

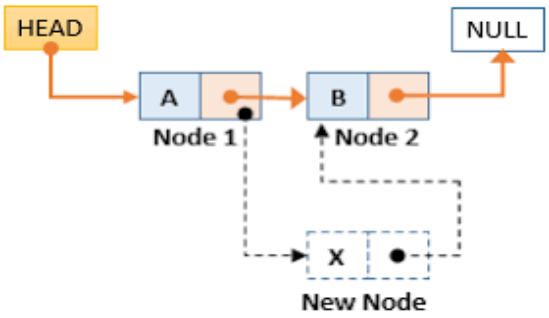


fig 3: insertion of element

there are different types of linked list

- Single linked list
- Double linked list
- Circular linked list

SINGLE LINKED LIST

Singly linked list is a collection of nodes linked together in a sequential way where each node of singly linked list contains a data field and an address field which contains the reference of the next node.

the main drawback of this single linked list is that once if we traverse to the end of list we can not traverse back to the first

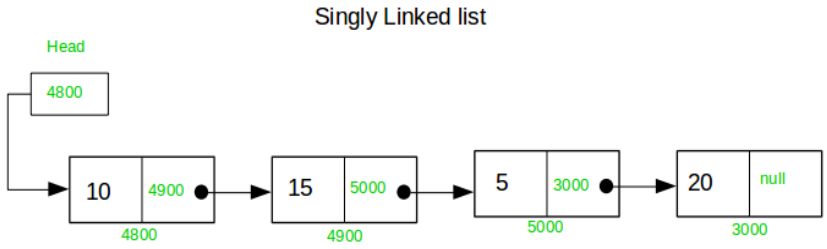


fig 4: representation of single linked list

DOUBLE LINKED LIST

Doubly linked list is a type of linked list in which each node apart from storing its data has two links. The first link points to the previous node in the list and the second link points to the next node in the list. The first node of the list has its previous link pointing to NULL similarly the last node of the list has its next node pointing to NULL.

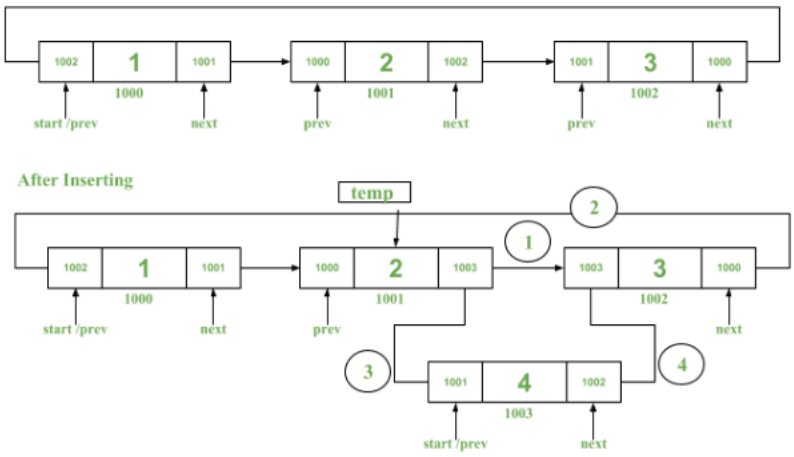


fig 5: representation of doubly linked list and also insertion of element

CIRCULAR LINKED LIST

Circular linked list is a linked list where all nodes are connected to form a circle. There is no NULL at the end. A circular linked list can be a singly circular linked list or doubly circular linked list. We can maintain a pointer to the last inserted node and front can always be obtained as next of last.

The last node of the list is connected the first node of the linked list where the head contains the address of first node

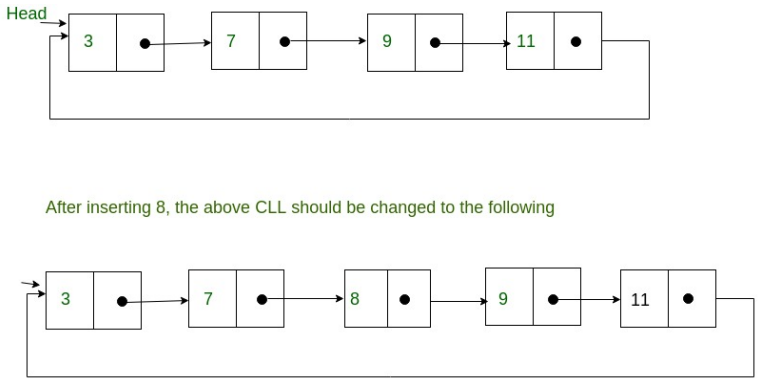


fig 6: representation of circular linked list and insertion of element

CIRCULAR DOUBLE LINKED LIST

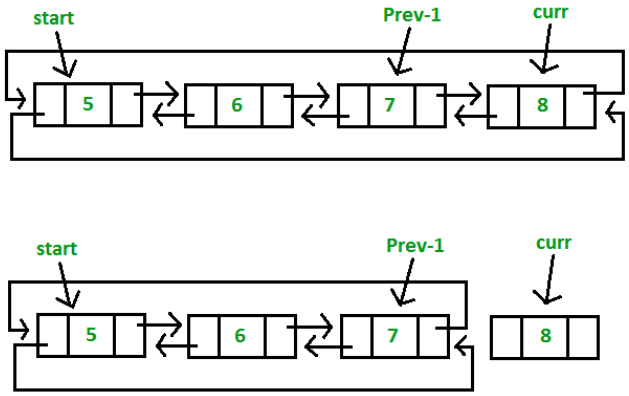


fig 7: representation of circular doubly linked list

REPRESENTATION OF STACKS AND QUEUES USING LINKED LIST

Stacks and queues can also be represented using linked list. Using insertion and deletion operations

Stack can be implemented using operations insertion at end and deletion at end.

Queue can be implemented using operations insertion at end and deletion at front.

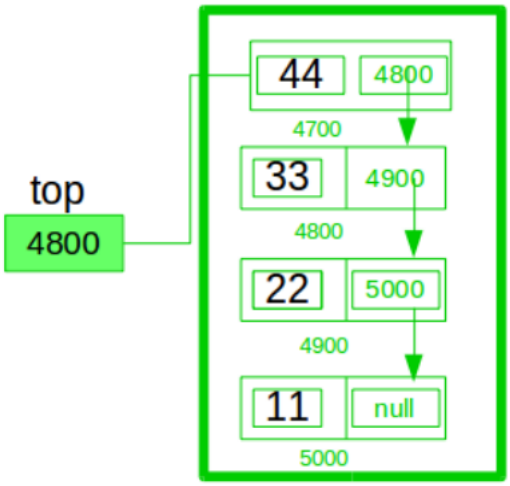


fig 8: representation of stack using linked list

Linked list as data structure

Linked lists are important not only as a means of implementing stack and queues but also as data structures in their own right. An item is accessed in a linked list by traversing the list from the beginning. An array implementation allows access to the nth item in a group using a single operation, whereas a list implementation requires n operations. It is necessary to pass through each of the first n-1 elements before reaching the nth element because there is no relation between the memory location occupied by an element of a list and its position with in that list.

The advantages of a list over an array occurs when it is necessary to insert or delete an element in the middle of a group of other elements.

Finally out of these many types of linked list I have used singly linked list in my code implementation.

CHAPTER 3

DESIGN

3.1 DESIGN GOALS

Design goals in railways ticket booking include the final goals that are expected to reach the costumer by implementing the different parts of the code

The different parts of the code contains:

- Reservation of passenger train ticket.
- Cancellation of the ticket.
- Display the train ticket details.
- And also if the booking is full send them in to waiting list

3.1.1 RESERVATION OF PASSENGER TICKET

- In this part the ticket is booked by asking the basic information that is required (i.e. name, age, from where to where etc.)
- Once the ticket is booked its ok enjoy journey , if the booking is full it will send to waiting list and wait for the passenger to cancel ticket

3.1.2 CANCELATION OF TICKET

- If the passenger is willing to cancel the ticket the system is programed in such a way that he/she can cancel the ticket.
- When the costumer cancels the ticket the ticket of the passenger who is send in to waiting list are booked automatically.

3.1.3 DISPLAY THE TRAIN TICKET DETAILS

- This function displays all the ticket details regarding the passenger

3.2 ALGORITHM

Algorithm for railway ticket booking

•Begin.

•Declare the header files that required in the program

Eg: #include<stdio.h>

#include<conio.h>

#include<stdlib.h>

•Create the type define structure for creating node.

•The structure node contains registration no, name, age, address of next node.

•Typedef struct node

Begin;

int regno;

char name[20];

char gender[7];

char place[10];

char destination[10];

int age;

struct node *ptr;

end node;

•Declare the function that are required in the program.

Create();

Reserve();

Cancel();

Enq();

Display();

• Declare count=0,num=0;

• Declare max size.

• Define the functions what and all are declared to use the functions in the main program.

void create()

begin;

- Declare new1; \\new node
- Allocate memory dynamically for new node. \\dynamic memory allocation

- Initialize regno to 1;
- Input name;
- Read name;
- Input age;
- Read age;
- Input from to destination;
- Read from to destination;
- Initialize head to new and new1 pointer to NULL and also increase num;

end;

- reserve()

  begin;

  If head is equal to NULL call create function and return 1.

  Else traverse up to the end of linked list and link new node at the end;

  If num is less than size increment num, regno is num, return 1;

  Else send to enquiry for waiting listand return zero;

  end;

- cancel()

  begin;

  if there is only one ticket booked and the given regno is euqual to present node then delete the node;

  else traverse through the linked list till it is not equal to the given register number and then free the node;

- enq()

  begin;

  if rear is NULL then initialize rear pointer to NULL and front to rear

  else

```
            initialize cur to new node
            rear pointer to cur;
            cur pointer to NULL;
            rear to cur;
          end;
Void display()
      begin;
      Declare temporary node called   node *tmp;
      Initialize temp to start;
      While temp node not equal to NULL
      begin;
      display Registration Number;
      display Name;
     display age ;
      display from to destination;
      then move to next node;
      end;
      end;
```

CHAPTER 4

IMPLEMENTATION

4.1 MODULE 1 FUNCTIONALITY

In module1 we will declare the all the functions structures that all needed and used in the program

```
#include<stdio.h>
#include<stdlib.h>
#include<malloc.h>
#define size 5
typedef struct node
{
        int regno;
        char name[20];
        char gender[7];
        char place[10];
        char destination[10];
        int age;
        struct node *ptr;
}node;
node* deq();
void create();
int reserve();
int cancel(int);
void enq(node*);
void display();
node *new1,*cur,*head,*front,*rear,*pre;
int count=0;
int num=0;
```

4.2 MODULE 2 FUNCTIONALITY

In module 2 we create a node that takes the passenger details and helps to book one passenger details

The further passenger details are taken by the reserve function

```
void create( )
{
        new1=(node*)malloc(sizeof(node));
        new1->regno=1;
        printf("Name:\n");
        scanf("%s",new1->name);
        printf("Age :\n");
        scanf("%d", &new1->age);
        printf("Enter gender\n");
        scanf("%s",new1->gender);
        printf("Enter your place and destination\n");
        scanf("%s",new1->place);
        scanf("%s",new1->destination);
        head=new1;
        new1->ptr=NULL;
        num++;
}

int reserve(head)
{
        if(head==NULL)
        {
          create(head);
          return 1;
        }
        else
```

```
{
  cur=head;
  while(cur->ptr!=NULL)
   {
     cur=cur->ptr;
   }
  new1=(node*)malloc(sizeof(node));
  printf("Name:\n");
  scanf("%s", new1->name);
  printf("Age :\n");
  scanf("%d", &new1->age);
  printf("Enter gender\n");
  scanf("%s",new1->gender);
  printf("Enter your place and destination\n");
  scanf("%s",new1->place);
  scanf("%s",new1->destination);
  new1->ptr=NULL;
  if(num<=size)
   {
        num++;
        new1->regno=num;
        cur->ptr=new1;
        return 1;
   }
  else
   {
        enq(new1);
        return 0;
   }
}
```

```
}
```

4.3 MODULE 3 FONTIONALITY

In this function defines about the cancelation of ticket

```
int cancel(int reg)
{
        cur=head;
        pre=NULL;
        if(head==NULL)
        return -1;
        if(cur->ptr==NULL && cur->regno==reg)
         {
           head=NULL;
           num--;
           free(cur);
           return 1;
         }
    else
    {
        while(cur->regno!=reg && cur->ptr!=NULL)
                {
                  pre=cur;
                  cur=cur->ptr;
                }
                if(cur==NULL && cur->regno!=reg)
                   return -1;
                else
                        pre->ptr=cur->ptr;
                free(cur);
                new1=deq();
                while(pre->ptr!=NULL)
```

```
                pre=pre->ptr;
        pre->ptr=new1;
        num--;
        return 1;
    }
}
```

4.4 M0DULE 4 FUNCTIONALITY

These funtions are used to put the passengers in the waiting list when the tickets are full

```
void enq(node *new1)
{
    if(rear==NULL)
    {
        rear=new1;
        rear->ptr=NULL;
        front=rear;
    }
    else
    {
        cur=new1;
        rear->ptr=cur;
        cur->ptr=NULL;
        rear=cur;
    }
    count++;
}

node* deq()
```

```
{
        node *front1;
        front1=front;
        if(front==NULL)
                return NULL;
        else
        {
           count-- ;
                if(front->ptr!=NULL)
                {
                        front=front->ptr;
                        front1->ptr=NULL;
                        return front1;
                }
                else
                {
                        front=NULL;
                        rear=NULL;
                        return front1;
                }
        }
}
```

4.5 MODULE 5 FUNCTIONALITY

This function displays the ticket details of the passengers

```
void display()
{
        cur=head;
        while(cur!=NULL)
        {
                printf("\nRegistration Number: %d\n",cur->regno);
                printf("Name : %s\n", cur->name);
                printf("Gender :%s\n",cur->gender);
                printf("Your journey from %s to %s\n",cur->place,cur->destination);
                cur=cur->ptr;
   }

}
```

4.6 MODULE 6 FUNCTIONALITY

This is the main function where we access all the sub functions

```
int main()
{
int ch, stat=0,canc=0, reg=0;
head=NULL;
rear=NULL;
front=NULL;
while(1)
{
scanf("%d", &ch);
switch(ch)
{
case 1 : stat=reserve(head);
```

```
if(stat==0)
printf("Booking Full\n”);
else
printf(" Booking Successful\n”);

break;
case 2:   printf(" input the Registration number to be cancelled\n");
scanf(" %d", &reg);
                    if(reg>num)
                    printf("Invalid register number\n");
                    else
                    {
                    canc=cancel(reg);
                    if(canc==-1)
                        printf("Registration number invalid!!\n");
                    else
                        printf("Ticket cancelled successfully\n");
                        }
                    break;

          case 3: display();
          break;
          case 4: exit(0);
        }
}
}
```

CHAPTER 5

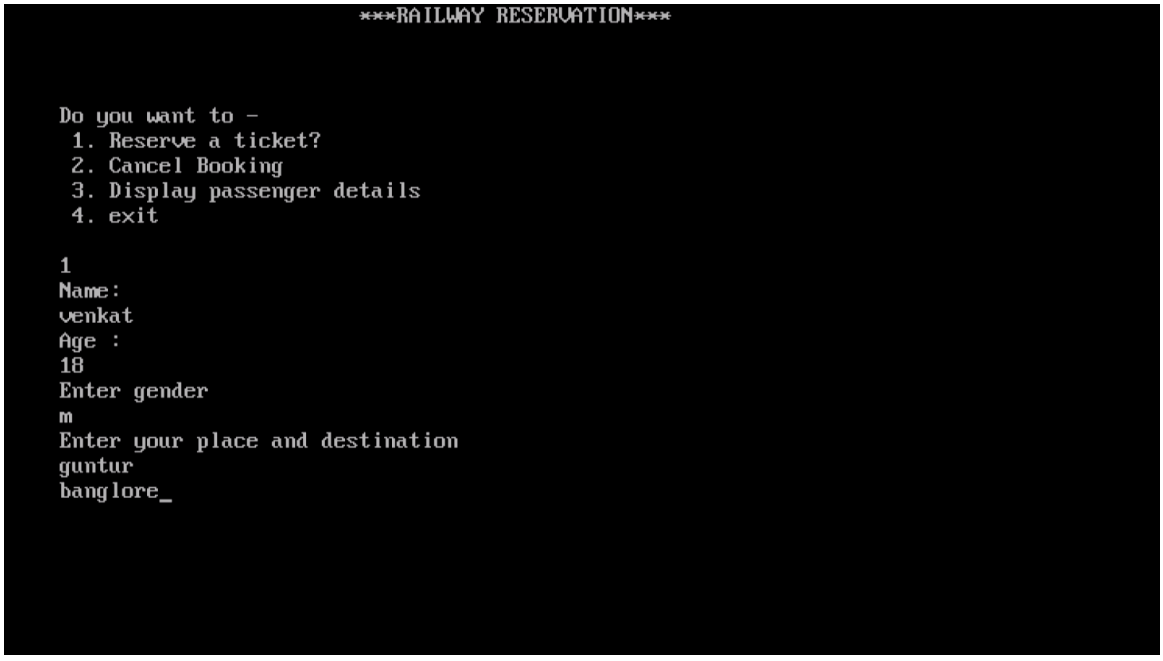
RESULT

the program is executed successfully with out any errors and the output is displayed correctly with out any disturbance

after executing the program the information in output screen we can see is

1. for registering the ticket.
2. For cancelling the ticket.
3. For displaying the details of the ticket
4. And to exit from the output screen exit function is used.

OUTPUT:



```
                        ***RAILWAY RESERVATION***


Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit

1
Name:
venkat
Age :
18
Enter gender
m
Enter your place and destination
guntur
banglore
```

```
 3. Display passenger details
 4. exit

1
Name:
venkat
Age :
18
Enter gender
m
Enter your place and destination
guntur
banglore

Booking Successful!!! Enjoy your journey! Your Reg No is 1


Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit
```

```
 4. exit

1
Name:
sai
Age :
19
Enter gender
male
Enter your place and destination
banglore to guntur

Booking Successful!!! Enjoy your journey! Your Reg No is 2



Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit

Name:
Age :
-
```

```
 3. Display passenger details
 4. exit

1
Name:
kumar
Age :
19
Enter gender
male
Enter your place and destination
guntur
tirupathi

Booking Successful!!! Enjoy your journey! Your Reg No is 3



Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit
```

```
 3. Display passenger details
 4. exit

1
Name:
giridhar
Age :
44
Enter gender
male
Enter your place and destination
banglore
tirupathi

Booking Successful!!! Enjoy your journey! Your Reg No is 5



Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit
```

```
 3. Display passenger details
 4. exit

1
Name:
prasanthi
Age :
20
Enter gender
female
Enter your place and destination
tirupathi
vijayawada

Booking Successful!!! Enjoy your journey! Your Reg No is 6



Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit

-
```

```
 2. Cancel Booking
 3. Display passenger details
 4. exit

1
Name:
guntupalli
Age :
20
Enter gender
male
Enter your place and destination
guntur
madras

Booking Full!!
You are added to waiting list. Waiting list number 1

Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit
```

```
Your journey from guntur to tirupathi

Registration Number: 4
Name : sailaja
Gender :female
Your journey from banglore to guntur

Registration Number: 5
Name : giridhar
Gender :male
Your journey from banglore to tirupathi

Registration Number: 6
Name : prasanthi
Gender :female
Your journey from tirupathi to vijayawada


Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit

-
```

```
Gender :female
Your journey from tirupathi to vijayawada


Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit

2

 Give the Registration number to be cancelled
2

Registration cancelled successfully


Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit
```

```
Your journey from banglore to guntur

Registration Number: 5
Name : giridhar
Gender :male
Your journey from banglore to tirupathi

Registration Number: 6
Name : prasanthi
Gender :female
Your journey from tirupathi to vijayawada

Registration Number: 0
Name : guntupalli
Gender :male
Your journey from guntur to madras


Do you want to -
 1. Reserve a ticket?
 2. Cancel Booking
 3. Display passenger details
 4. exit

-
```

CHAPTER 6

CONCLUSION

This project titled “RIALWAY TICKET BOOKING” is a simple program built for the booking railway ticket. The purpose of this source is to describe the simple railway ticket booking process, which provides different options to book a ticket, to cancel a ticket, to display the details of the ticket. In addition to this you can save the time needed to travel to the railway reservation office and waiting in the queue for your turn. Towards a greener planet – Instead of printing your ticket you can also choose to travel with the soft copy of your booked ticket in your laptop or even on your mobiles.

Outcomes-

- Creating the user.
- Registration or booking of ticket.
- If vacancies are there ticket will be confirmed else will sent to waiting list
- Cancelation of ticket is also done if they wants to cancel the ticket.
- Enquiry of details regarding passenger ticket is done.
- Finally display of details regarding the train ticket.

Advantages-

- Provides the searching facilities based on various factors: train, seat ,enquiry.
- Manage the information of booking.
- Mange the information of seats.
- When the seats are filled the passenger is kept in to waiting list and when there is availability the ticket is automatically filled.
- The main objective of this train ticket booking project is to reduce consumption of time for booking ticket.

CHAPTER 7

REFERENCES

- Programing with c (by Ms. DEEPIKA)
- Programing in c and data structure (by Mr. SIVABALAN)
- C programing and data structures(by E BALAGURUSWAMY)
- http://www.oreily.com/library/view/introduction-to-digital/9780470900550/chap2-sec003.html