NEW HORIZON
COLLEGE OF ENGINEERING

A MINI PROJECT

REPORT

for

MINI PROJECT IN C (19CSE39)

LUGGAGE HANDLER

submitted by

S REVANTH KUMAR

1NH19CS149

SEM-III

SEC -3C

In partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

Certificate

This is to certify that the mini project work titled

LUGGAGE HANDLER

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

Submitted by

S REVANTH KUMAR

1NH19CS149

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.________________________	________________________

LUGGAGE HANDLER

ORIGINALITY REPORT

7%	%	7%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	H. Krishna Murthy, P S Ashok kumar, Punith Palagi, Noor Bhasaha. "Data transmission for cluster-based wireless sensor networks using SET protocol", 2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), 2017 Publication	3%
2	"Enhancement of Data Security using Circular Queue Based Encryption Algorithm", International Journal of Innovative Technology and Exploring Engineering, 2019 Publication	1%
3	Cong Wang, Kui Ren, Jia Wang, Karthik Mahendra Raje Urs. "Harnessing the Cloud for Securely Solving Large-Scale Systems of Linear Equations", 2011 31st International Conference on Distributed Computing Systems, 2011 Publication	1%
4	Jin Zhang. "Formal Specification and Implementation of Priority Queue with Starvation Handling", Studies in Computational	<1%

Intelligence, 2011
Publication

5	Kayvan Memarian, Victor B. F. Gomes, Brooks Davis, Stephen Kell, Alexander Richardson, Robert N. M. Watson, Peter Sewell. "Exploring C semantics and pointer provenance", Proceedings of the ACM on Programming Languages, 2019 Publication	<1%
6	"Question Papers and Solutions: DC-05 (Problem Solving through ‘C’)", IETE Journal of Education, 2015 Publication	<1%
7	Christopher D. Watkins, Stephen R. Marenka. "TECHNICAL CONSIDERATIONS FOR VIRTUAL REALITY SYSTEMS", Elsevier BV, 1994 Publication	<1%
8	HUANG, RUNHE, and JIANHUA MA. "Lists", Data Structures and Algorithms, 2003. Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

ABSTRACT

The main purpose of developing the project was to store all the information about the passenger's luggage and help in reaching the luggage to the right passenger. The software is capable of storing the passenger's information while checking in and also delete the passenger's data while checking out. This program displays the passengers in different flights with their respective destinations and bag weights. Myproject is aimed at reducing the instances of lost bags, increasing the ability to track and report on baggageincluding weight changes to prevent theft and loss. Luggage handler can add, delete and feed information like name, age, flight number, destination. We use a data structure for updating luggage one by one. This project can be used in Airports, Railway Station, Ports and even in hotels as it can help the luggage reach the right hands in a much faster way. Using data structures moreover, will help us easily locate luggage as we'll know in which flight the luggage is present.

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 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 **Ms. Archana Nair S, 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.

S REVANTH KUMAR(1NH19CS149)

CONTENTS

LIST OF FIGURES

LUGGAGE HANDLER

CHAPTER 1

INTRODUCTION

A Luggage Handler is a conveyor system installed in airports/railways/ports that help in transporting luggage from ticket counters to luggage loadout areas.

It delivers checked baggage coming from airplanes to baggage claims. The first automated Luggage Handler was invented by BNP Associates in 1971, and this technology is in use in almost every major airport worldwide today.

Luggage Handler will then scan and sort the bags. It helps people in transporting their luggage without any problem and makes sure that their journey is comfortable and enjoyable.

1.1 PROBLEM STATEMENT

Luggage regulation machine can add, delete and feed information like name, age, flight number, destination. We use a data structure for updating luggage one by one. We can check in or check out using the luggage handler. We can also display all the entries stored in the Luggage Handler and view them to gain information.

1.2 OBJECTIVES

This project can be used in Airports, Railway Station, Ports and even in hotels as it can help the luggage reach the right hands in a much faster way. Using data structures moreover, will help us easily locate luggage as we'll know in which flight the luggage ispresent.

1.3 METHODOLOGY

The main methodology used here are implementation of data structures. The main aim here is to keep the luggage on track with all the necessary information so that it won't be misplaced by any chance.

1.4 EXPECTED OUTCOMES

My project here is based on queues and it's expected outcome is to read the users data and their luggage's data properly and making sure the luggage is safe and easily retrievable. We can check in or check out using the luggage handler. We can also display all the entries stored in the Luggage Handler and view them to gain information.

1.5 HARDWARE AND SOFTWARE REQUIREMENTS

Processor	: Intel i3 or above / AMD Ryzen 3 or above
RAM	: 4GB
Hard Disk	: 10 GB
Input device	: Standard Keyboard and Mouse
Output device	: 1920x1080 Resolution Monitor
Operating system	: Windows XP or higher
C Compiler	: CODEBLOCKS or any c compiler

CHAPTER 2

DATA STRUCTURES

Data Structure is a way of collection and organization of data in such a way that we can perform operations on these data and make use of them effectively. The structure should be simple and effective enough so that it can process the data. The main aim of data structures is to reduce the complexity and increases the efficiency.

Space Complexity - It's the amount of space needed for the algorithm during its execution.

Time Complexity – The time required by the program to run till its completion is termed as time complexity.

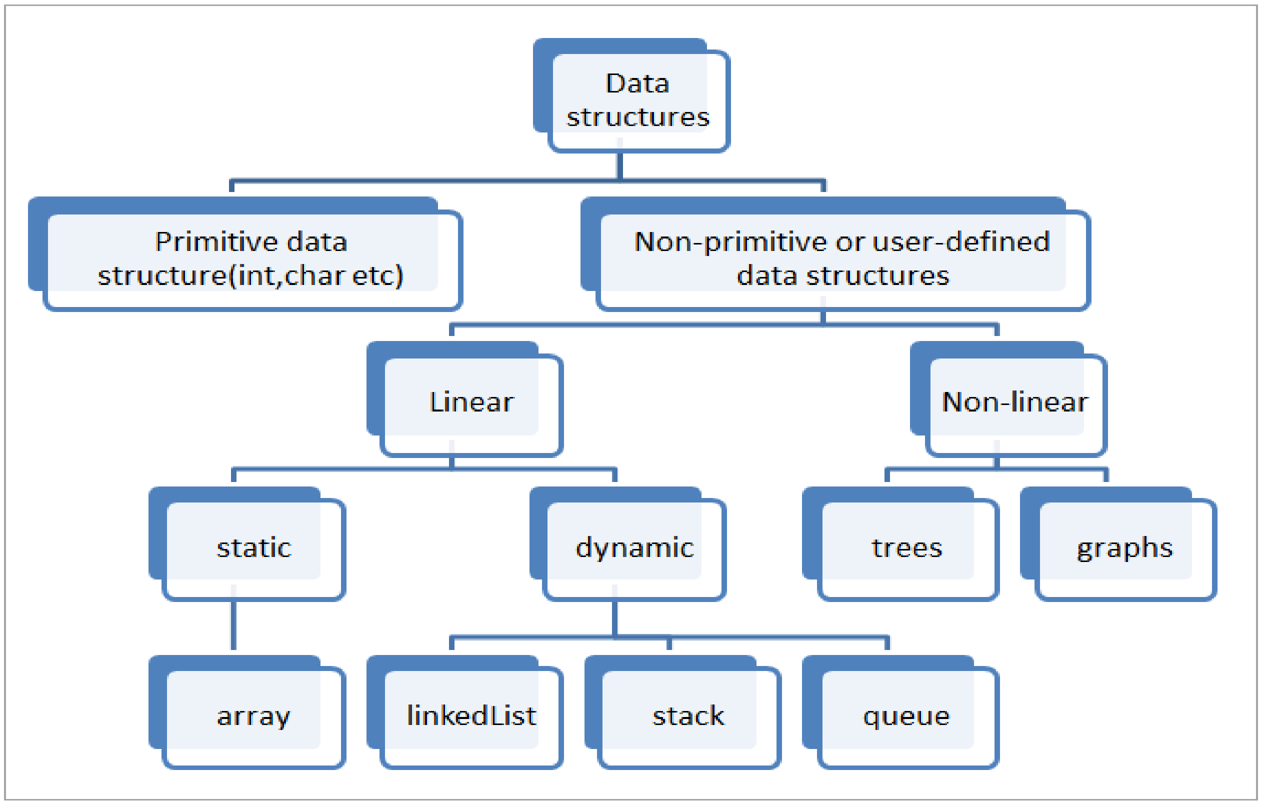

Data structures
Primitive data structure(int,char etc)
Non-primitive or user-defined data structures
Linear
Non-linear
static
dynamic
trees
graphs
array
linkedList
stack
queue

Fig.2.1.1

Types of data structures:

Data structure types are determined by the type of operations that are required. These types include:

2.1 Stacks - A stack is an algorithm in which the element can be added or deleted fromone end known as the top of the stack. Push means we are adding a term to the stack and Pop means that we are deleting an element from the stack.

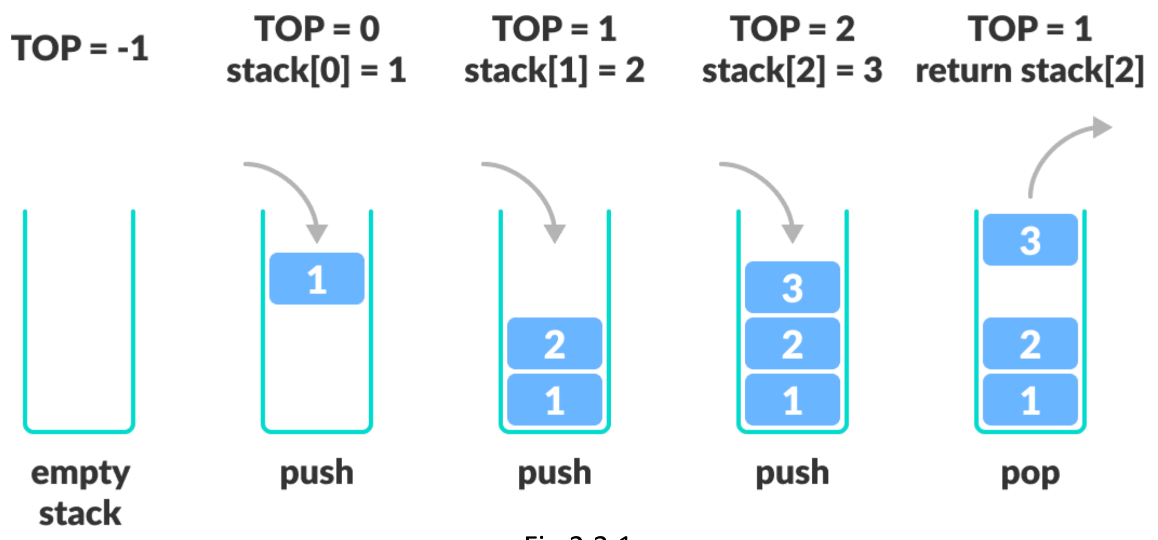


Fig.2.2.1

2.2 Queues – A queue is a type of algorithm in which deletions can happen at one place called Front and insertions can take place at the other end called Rear.

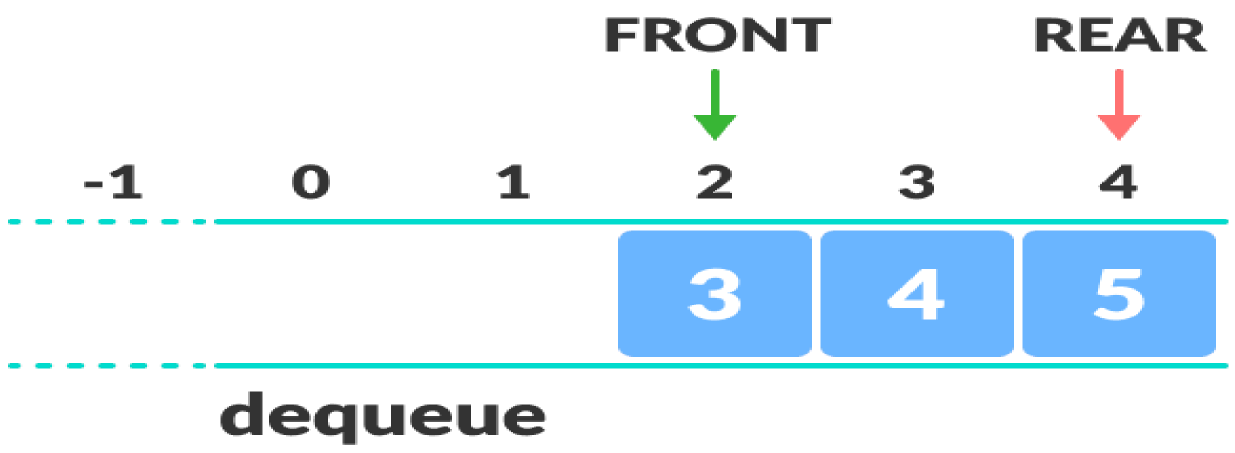


Fig.2.3.1

2.1.1 Circular queues - Circular Queue is a linear data structure in which the operations are performed based on FIFO (First In First Out) principle and the last position is connected back to the first position to make a circle. It is also called "Ring Buffer".

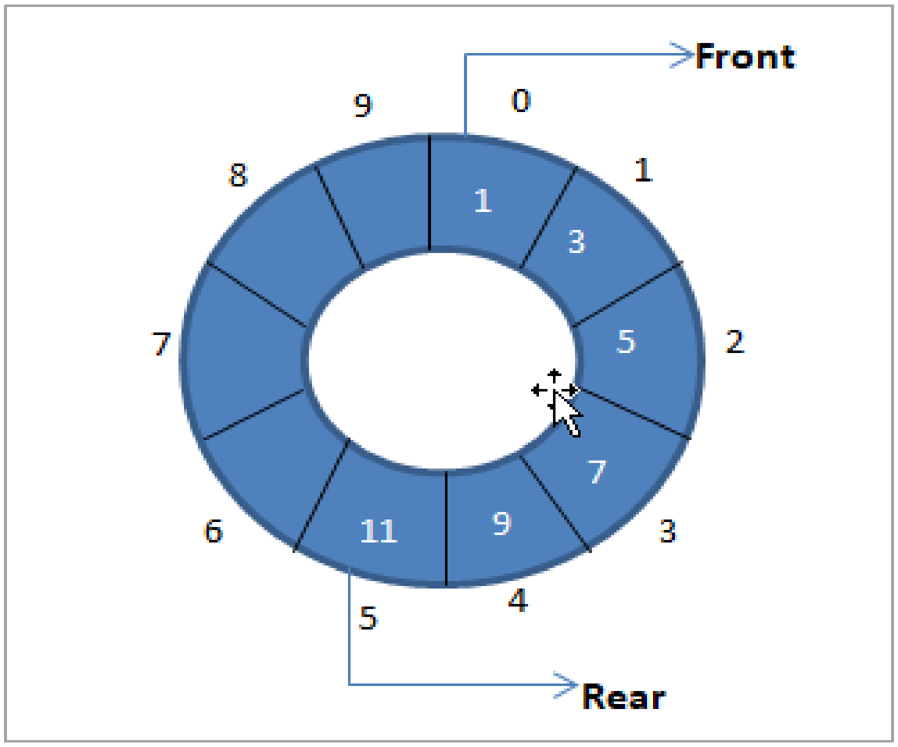


Fig.2.3.1

2.1.2 Double Ended queue - Double ended queue is a more generalized form of queue data structure which allows insertion and removal of elements from both the ends, i.e. front and back.

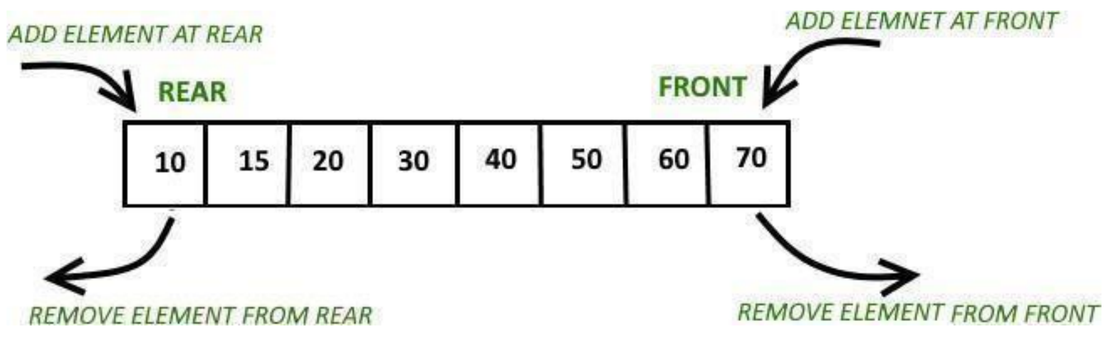


Fig.2.3.2

2.1.3 Priority Queue - A priority queue is a collection in which items can be added at any time, but the only item that can be removed is the one with the highest priority. In Priority queue items are ordered by key value so that item with the lowest value of key is at front and item with the highest value of key is at rear or vice versa. So, we're assigned priority to item based on its key value. Lower the value, higher thepriority.

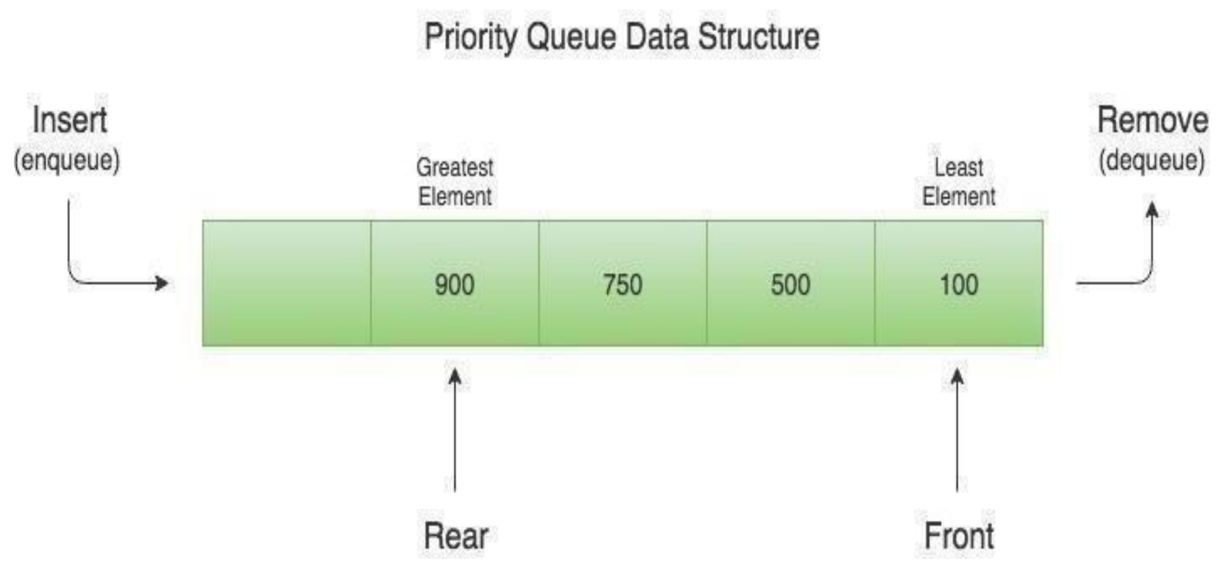


Fig.2.3.3

2.2 Linked lists - A linked list is linear collection of data elements known as Nodes. Nodes consists of two parts i.e. Data and Pointer part. The data part holds the data inserted and the pointer part holds the pointer which directs to the nextnode.

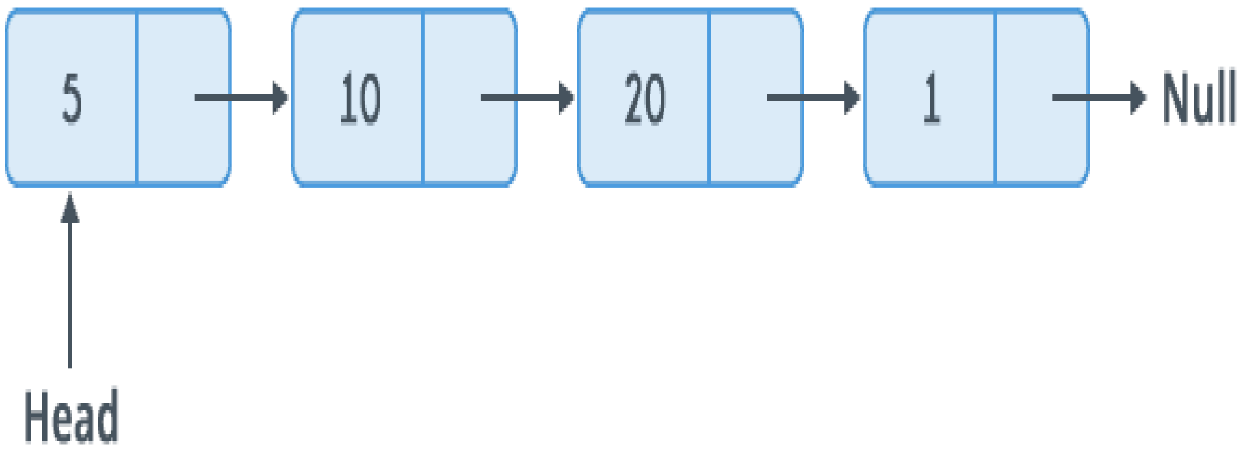


Fig.2.4.1

2.3 Trees - A tree is a nonlinear algorithm in data structure that consists of nodes connected by edges. A node contains a key or value to its child nodes. Root is the topmost or parent node. Trees are definitely useful as linear data structures take a lot of time with the increase in data.

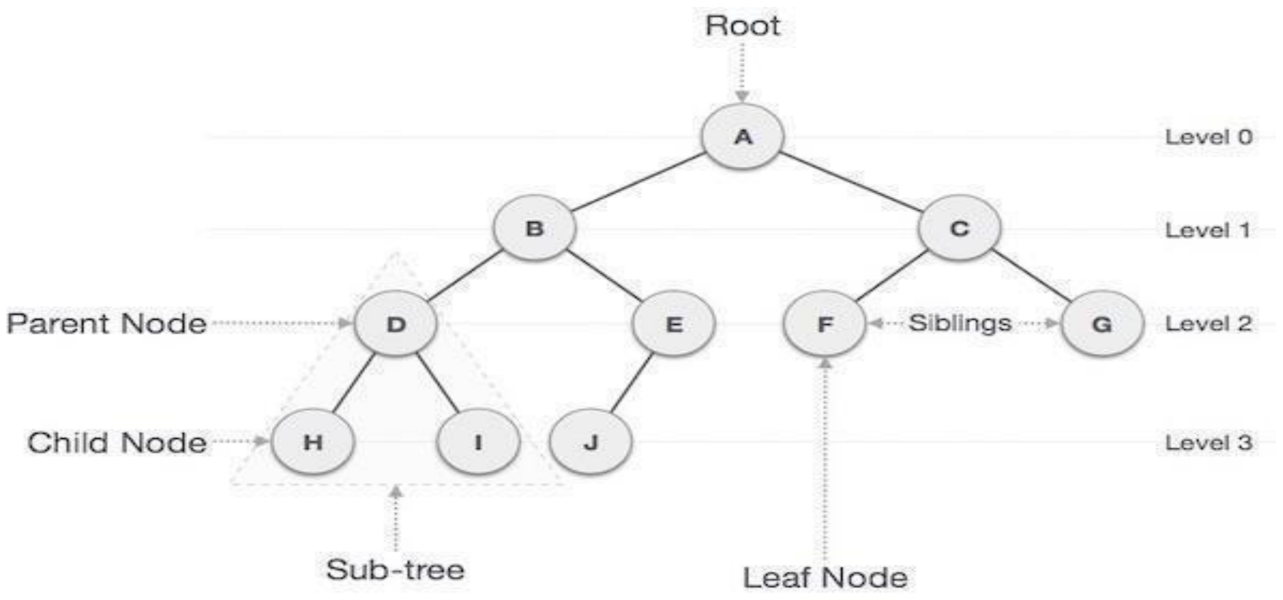


Fig.2.5.1

2.4 Graphs- A graph stores a data in a non-linear fashion. Graphs are made of a finite set of nodes known as vertices and lines that connect them are known asedges.

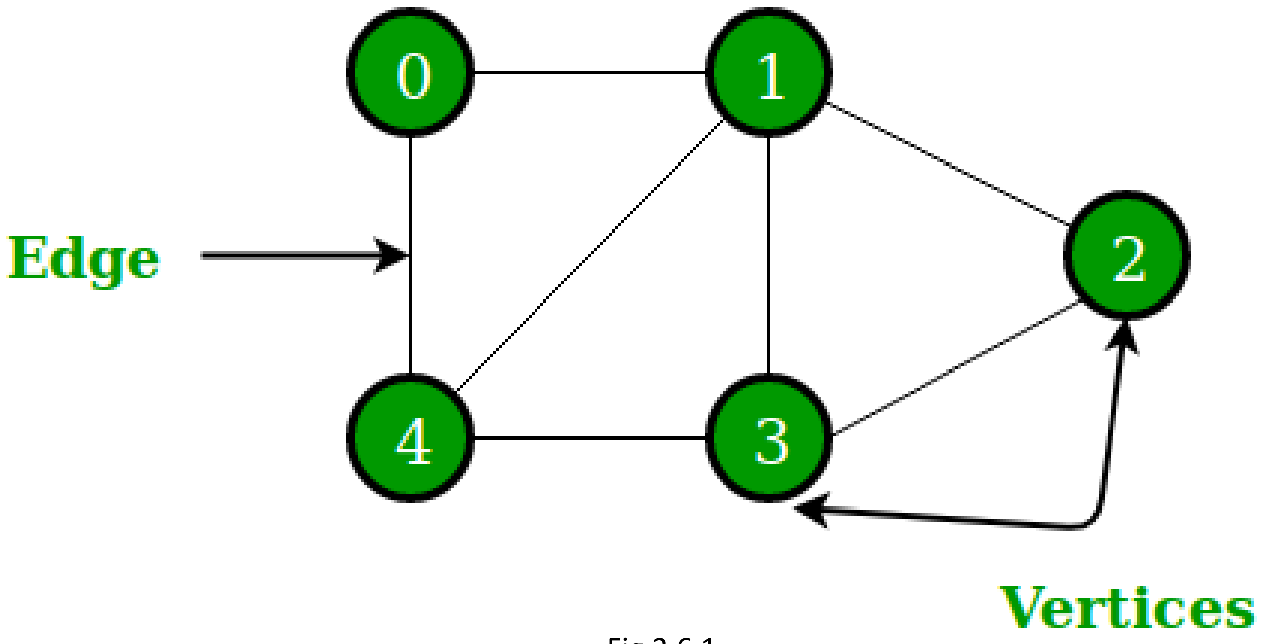


Fig.2.6.1

Why Data Structures are needed?

With increasing complexities in computing, the amount of data usage being used is increasing, this can affect the performance of the application,

Processing speed: To handle large data, high speed data processing is required. Such algorithms in data structures can help us organize data much faster and in an efficient way.

Data Search: Retrieving a particular record from database should be faster and with optimum use of resources.

Multiple requests: To handle multiple requests from multiple users.

Data is organized to form a data structure in such a way that all are not required to be searched and only the required data can be searched.

Data Structure Advantages:

Efficient Memory use: Memory plays a vital part when the data we are handling is huge, so use of data structures can help in optimization of memory by not using excess of it rather only what is required.

Reusability: Data structures can be reused, once we create a data structure, we will have the power to use it at any other place. They can be compiled into libraries so that others can also make use of their true efficiency.

Secure: Data structures hold information securely on a computer system. Most data structures do not take up a lot of our computer's memory rather a small portion of it.

Data stored be accessed at any time. It is also impossible to lose data, hence it provides security for the data.

Software: Data structures gives us the capacity to use and process our data on a software system. We can even log our work hours and produce a report and even make it as an automated process.

Data structures make all processes involving them and effective.

2.8 Operations in queue (General Algorithms)

2.8.1 Insertion

```
void insert ()
{
             if(CAPACITY ==rear)
               {
                   printf("Queue is full \n");
                }
              else
{
          int element;
          printf("Enter a number into queue : \n");
          scanf(" %d", &element);

          queue[rear]=element;
          rear++;
   }
}
```

2.8.2 Deletion

```
void delete ()

{
      if (front == rear)
     {
         printf("Queue is empty");
     }
      else
    {
     printf("Deleted element: %d", &queue[front]);
     for (i=0;i<rear-1;i++)

{
                  queue[i] = queue[i+1];
        }
              rear--;
      }
}
```

2.8.3 Display

```
void display ()
{
       if( front == rear)
         {
             printf(“Queue is empty”):
         }
        else
         {
                printf(“Queue elements:”);
                for(i=0; i< rear; i++)
                          {
                                   printf(“%d\n”, queue[i]);
                           }


            }


    }
```

2.9 FLOWCHART FOR ENQUEUE AND DEQUEUE

a. enqueue ()

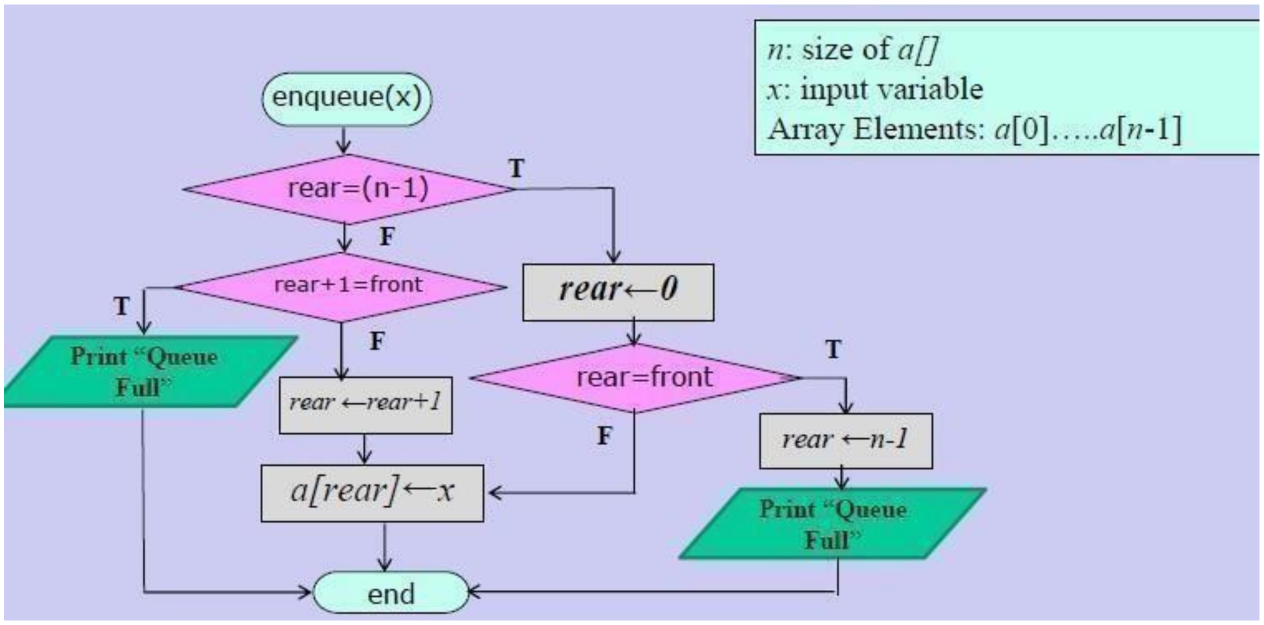


Fig.2.9.1(a)

b. dequeue ()

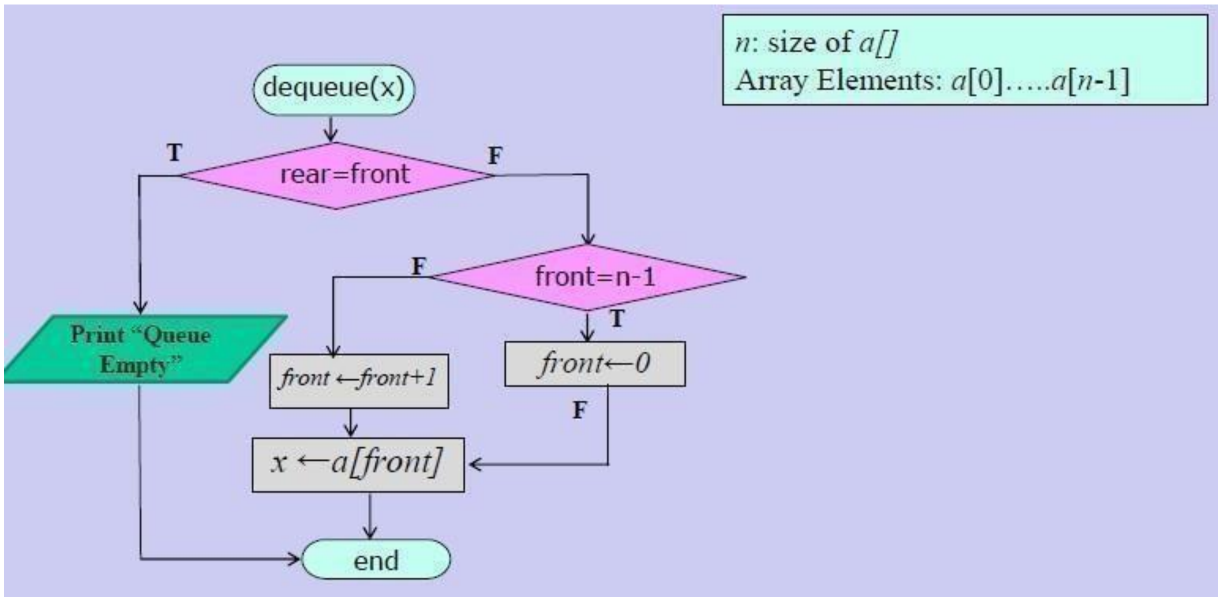


Fig.2.9.1(b)

CHAPTER 3

DESIGN

3.1 DESIGN GOALS

To enable secure outsourcing of file our mechanism design should achieve the following security:

No sensitive information from the customer’s private data can be derived by the machine

during the process. Store some of the basic information regarding passengers. Store information of passengers like passengers name, age, weight, destination and flight number.

Also adding a checkout option which can be used to delete that passenger’s data.

3.2 ALGORITHM / PSEUDOCODE

```
//preprocessor directives (<stdio.h>,<conio.h>,<stdlib.h>,<string.h>)

create structure bag
{
   initialize age,weight
   initialize name,destination and flight number
};

struct bag queue; //create queue

void checkin()
initialize length to 0
void checkout()
void display()
//void search()

void checkin()
{    initialize length to zero

          if length is equal to 0
            {
                 frontq=&queue[0]; rearq=&queue[0];
            }

          else
                 increment rearq

  print length
  print name of passenger
  print destination
  print flight number
  print age
  print weight of bag
 }


void display()
{

    if length is equal to 0
        print Queue is empty
    else
    { initialize i;
          print age,name,destination,flight number,bag weight

       read all the values from the structure bag (i.e age,name,destination,flight number,weight of
bag)
    }
}

void checkout(int key)
{
     if (key is equal to 1 or length is equal to 1)
       { initialize i
         if( length is equal to 1)
           { equate frontq and rearq to NULL
             decrement length
             print Element Deleted
           }
else

       {
         for(i is equal to 1;i less than or equal to length; increment i)
         queue[i-1] is equal to queue[i]
         decrement rearq
         print Element deleted
```

```
            decrement length
         }





     else if(( key greater than 1) and (key less than or equal to length))
     {
             if(key less than length)
                 for(i equal to key; i less than length; increment i)
                    { queue[i-1]=queue[i]; }

             decrement rearq
             print Elementdeleted
     }
     else
     print Invalid Choice
}


int main()
{
  initialize choice to -1
  //open a do while loop
  do{
  print Number of bags in queue
  print 1.Check in a new bag 2.Check out a bag
  print 3.View all bags 4.Search a bag
  print 5.Exit and clear

  print Enter your choice
  scan into choice

  switch(choice)
{

  case 1://check-in
           checkin();//calling checkin function
           break
  case 2://check-out
                if(length is equal to 0)
           printf Number of bags in queue to checkout

         else if(length is equal to 1)
           { checkout(1); }
           else if(length is greater than 1)
           { display(); initialize pos to 0
             print Enter bag number to checkout
             scan pos
             checkout(pos)
           }
           break


  case 3: //View all
                display()
                break

  case 4: //Search
          // search()
          break
  case 5://Exit
          print Thank you for using.Enter any key to exit.
          getch()
          return 0
  default: print Invalid Choice

             }
             }while(choice greater than or equal to 0)
}
```

CHAPTER 4

IMPLEMENTATION

4.1 MODULE 1 FUNCTIONALITY:

```
#include<stdio.h>
#include<conio.h>
#include<stdlib.h>
#include<string.h>
struct bag
{
        int age,wght;
        char name[30], dest[20], fl_no[10];
};
struct bag queue[50];
struct bag *frontq=NULL, *rearq=NULL, *current=NULL;
void checkin();
int length=0;
void checkout(int);
void display();
void search();

void checkin()
{
  if(length==0)
   {
     frontq=&queue[0]; rearq=&queue[0];
   }
  else
        { rearq++;}
  length++;
  printf("\n%d\n",length);
  printf("\nEnter name of passenger: ");
  scanf("%s",queue[length-1].name);
  printf("\nEnter Destination: ");
  scanf("%s",queue[length-1].dest);
  printf("\nEnter flight number: ");
```

```
   scanf("%s",queue[length-1].fl_no);

   printf("\nEnter Age: ");
   scanf("%d",&queue[length-1].age);
   printf("\nEnter the weight of the bag: ");
   scanf("%d",&queue[length-1].wght);
 }
```

In the above module structure bag queue is created. Next, in the check in function, we retrieve information of the passengers like name, age, weight, destination and flight number.

4.2 MODULE 4 FUNCTIONALITY:

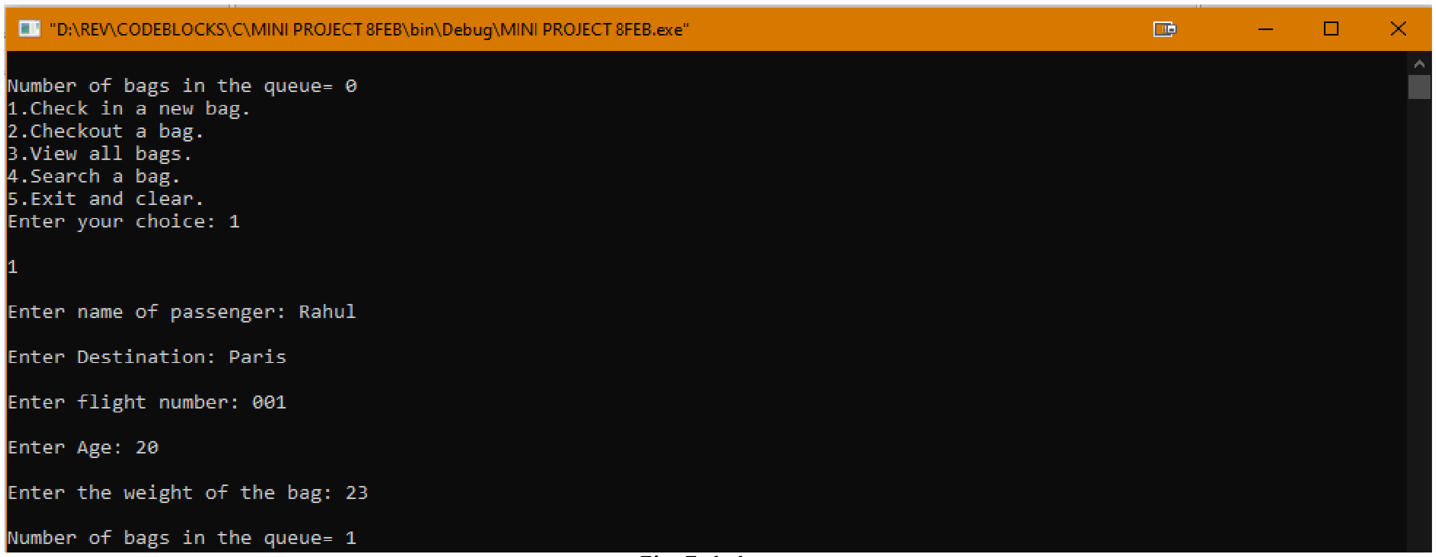
```
int main()
{
            int choice=-1;
            do{
            printf("\nNumber of bags in the queue= %d",length);
            printf("\n1.Check in a new bag.\n2.Checkout a bag.");

            printf("\n3.View all bags.\n4.Search a bag.");
            printf("\n5.Exit and clear.");
           printf("\nEnter your choice: ");
            scanf("%d",&choice);

               switch(choice)
             {
               case 1: //check-in
                     checkin();
                     break;

               case 2: //check-out
                      if(length==0)

           printf("\nNumber of bags in the queue to checkout...");
           else if(length==1)
            {
               checkout(1);
            }
```

```
        else if(length>1)
        {

           display();int pos=0;

           printf("\nEnter bag number to checkout:");
           scanf("%d",&pos);

           checkout(pos);
        }


           break;
           case 3: //View all
                  display();
                  break;

           case 4: //Search
                  search();
                  break;


           case 5: //Exit
                  printf("\nThank you for using.\nEnter any key to exit.");
                  getch();
                  return 0;




           default: printf("\nInvalid Choice");

         }


        } while(choice>=0);

}
```

This is the main () function, here first we create a do-while loop, then print options for storing the passenger’s information. Then we use a switch case to store user’s choice and create cases for the following: Check in a bag, Check out a bag, View all bags and Exit.

5. RESULTS

5.1.1 Input 1:

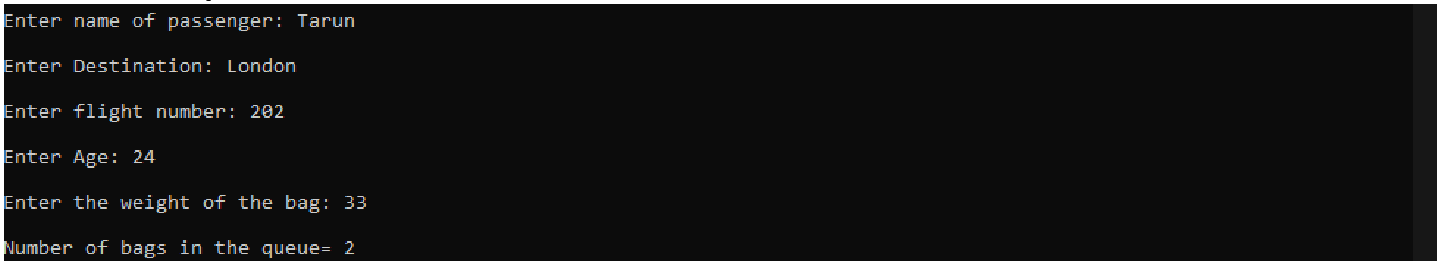


```
"D:\REV\CODEBLOCKS\C\MINI PROJECT 8FEB\bin\Debug\MINI PROJECT 8FEB.exe"

Number of bags in the queue= 0
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 1

1

Enter name of passenger: Rahul

Enter Destination: Paris

Enter flight number: 001

Enter Age: 20

Enter the weight of the bag: 23

Number of bags in the queue= 1
```

Fig.5.1.1

In this input, first we check in a bag of passenger Rahul. His destination being Paris and his flight number 001. He is 20 years old and the weight of his luggage was 23 kgs. Now the number of bags in queue is 1.

5.1.2 Input 2:

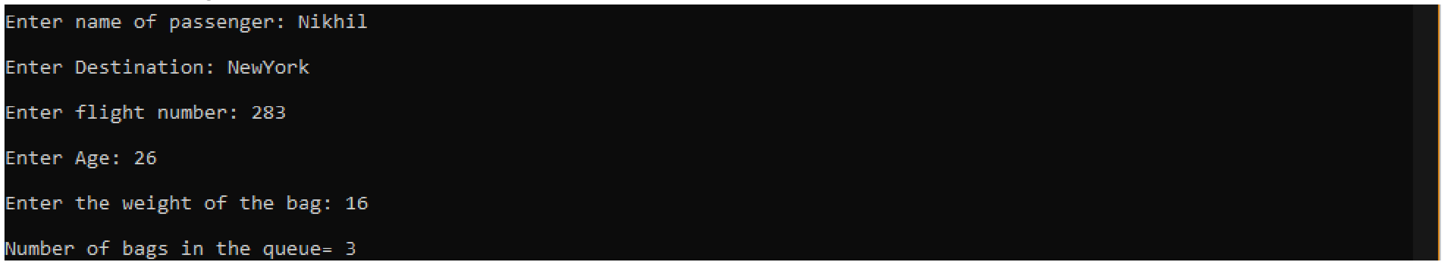


```
Enter name of passenger: Tarun

Enter Destination: London

Enter flight number: 202

Enter Age: 24

Enter the weight of the bag: 33

Number of bags in the queue= 2
```

Fig.5.1.2

In this input, we check in a bag of passenger Tarun. His destination being London and his flight number 202. He is 24 years old and the weight of his luggage was 33 kgs. Now the number of bags in queue is 2.

5.1.3 Input 3:

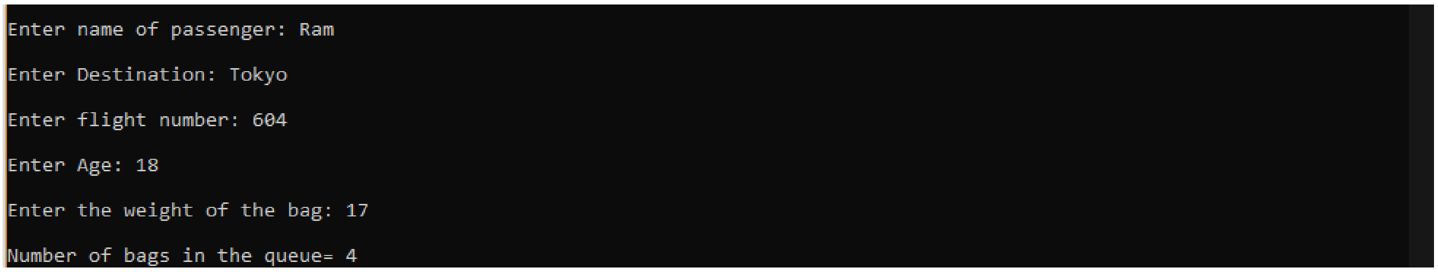


```
Enter name of passenger: Nikhil

Enter Destination: NewYork

Enter flight number: 283

Enter Age: 26

Enter the weight of the bag: 16

Number of bags in the queue= 3
```

Fig.5.1.3

In this input, we check in a bag of passenger Nikhil. His destination being New York and his flight number 283. He is 26 years old and the weight of his luggage was 16 kgs. Now the number of bags in queue is 3.

5.1.4 Input 4:

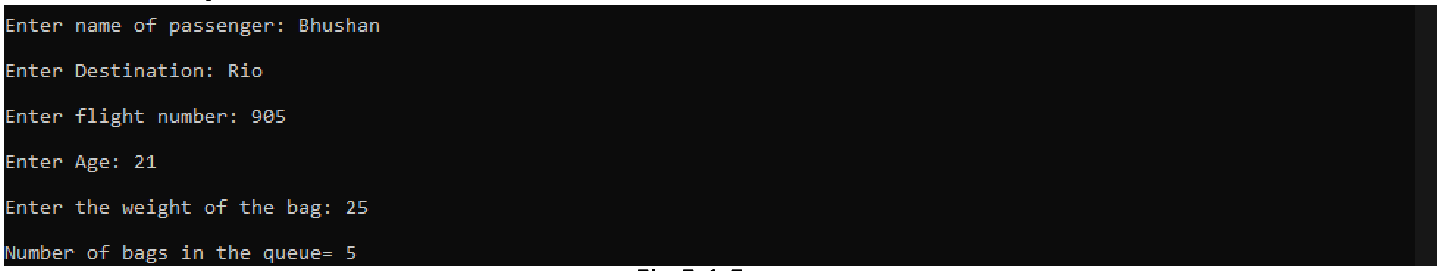


```
Enter name of passenger: Ram

Enter Destination: Tokyo

Enter flight number: 604

Enter Age: 18

Enter the weight of the bag: 17

Number of bags in the queue= 4
```

Fig.5.1.4

In this input, we check in a bag of passenger Ram. His destination being Tokyo and his flight number 604. He is 18 years old and the weight of his luggage was 17 kgs. Now the number of bags in queue is 4.

5.1.5 Input 5:

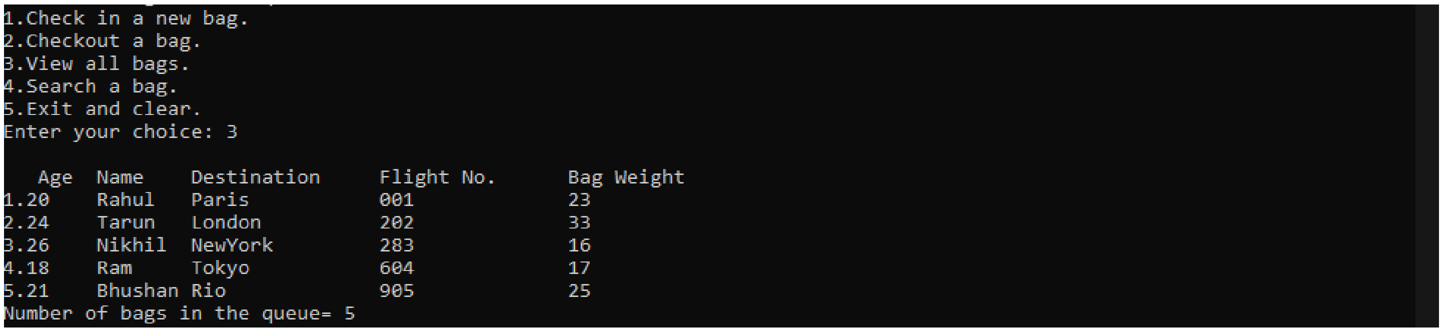


```
Enter name of passenger: Bhushan

Enter Destination: Rio

Enter flight number: 905

Enter Age: 21

Enter the weight of the bag: 25

Number of bags in the queue= 5
```

Fig.5.1.5

In this input, we check in a bag of passenger Bhushan. His destination being Rio and his flight number 905. He is 21 years old and the weight of his luggage was 25 kgs. Now the number of bags in queue is 5.

5.2 View (All 5 inputs):

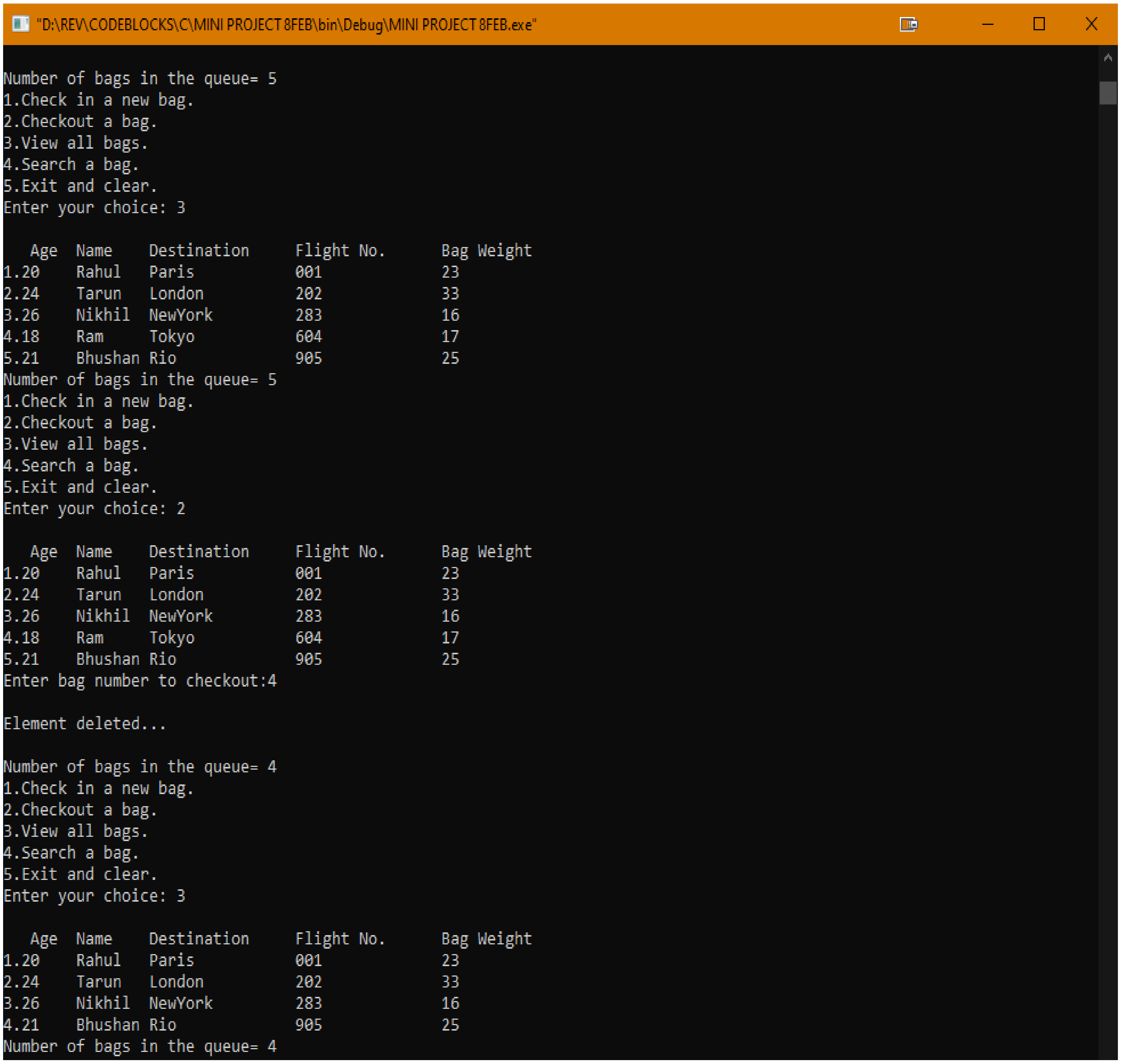


```
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 3

   Age  Name    Destination     Flight No.      Bag Weight
1.20    Rahul   Paris           001             23
2.24    Tarun   London          202             33
3.26    Nikhil  NewYork         283             16
4.18    Ram     Tokyo           604             17
5.21    Bhushan Rio             905             25
Number of bags in the queue= 5
```

Fig.5.2.1

Here we can see all the 5 passenger's information i.e. age, name, destination, flight number and bag weight.

5.3 Delete (Deleting entry number 4 from the list of 5entries):



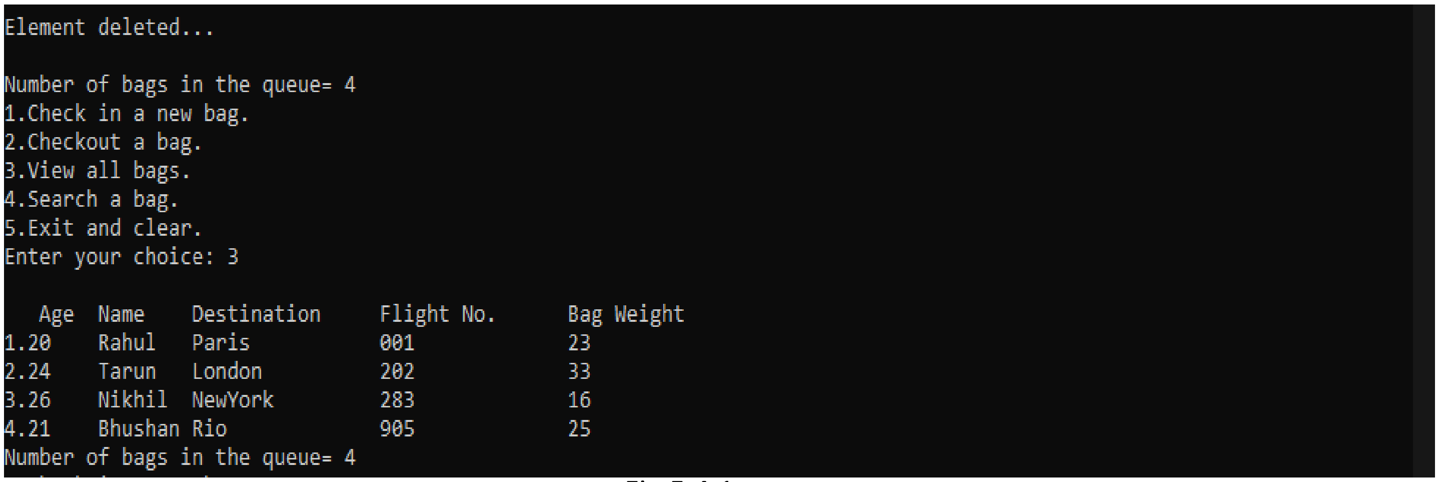
"D:\REV\CODEBLOCKS\C\MINI PROJECT 8FEB\bin\Debug\MINI PROJECT 8FEB.exe"

```
Number of bags in the queue= 5
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 3

   Age  Name    Destination     Flight No.      Bag Weight
1.20    Rahul   Paris           001             23
2.24    Tarun   London          202             33
3.26    Nikhil  NewYork         283             16
4.18    Ram     Tokyo           604             17
5.21    Bhushan Rio             905             25
Number of bags in the queue= 5
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 2

   Age  Name    Destination     Flight No.      Bag Weight
1.20    Rahul   Paris           001             23
2.24    Tarun   London          202             33
3.26    Nikhil  NewYork         283             16
4.18    Ram     Tokyo           604             17
5.21    Bhushan Rio             905             25
Enter bag number to checkout:4

Element deleted...

Number of bags in the queue= 4
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 3

   Age  Name    Destination     Flight No.      Bag Weight
1.20    Rahul   Paris           001             23
2.24    Tarun   London          202             33
3.26    Nikhil  NewYork         283             16
4.21    Bhushan Rio             905             25
Number of bags in the queue= 4
```

Fig.5.3.1

Now we want to delete a passenger's data from the list. Hence, we're deleting number 4's data.

5.4 After deletion:



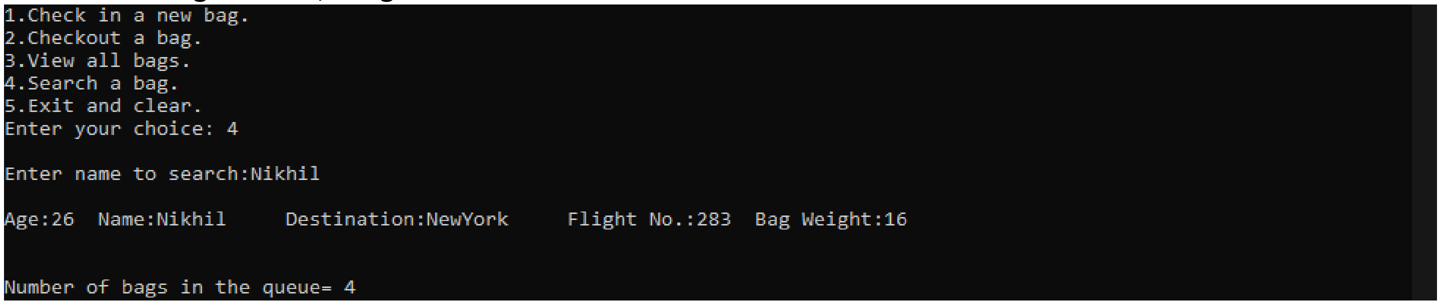
```
Element deleted...

Number of bags in the queue= 4
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 3

   Age  Name    Destination     Flight No.      Bag Weight
1.20    Rahul   Paris           001             23
2.24    Tarun   London          202             33
3.26    Nikhil  NewYork         283             16
4.21    Bhushan Rio             905             25
Number of bags in the queue= 4
```

Fig.5.4.1

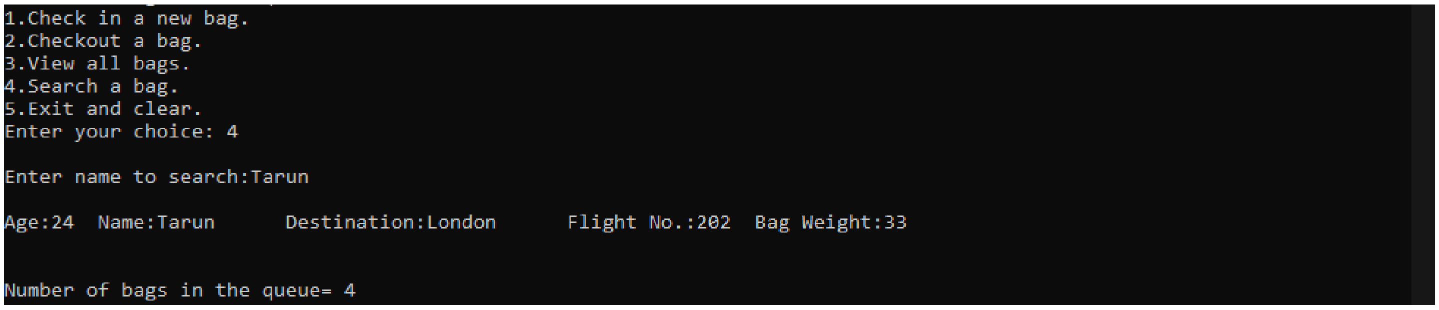
This is the final list after checkout of one of the passengers.

5.5.1 Search:

Searching for data, we get the below results.



```
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 4

Enter name to search:Nikhil

Age:26  Name:Nikhil     Destination:NewYork     Flight No.:283  Bag Weight:16


Number of bags in the queue= 4
```



```
1.Check in a new bag.
2.Checkout a bag.
3.View all bags.
4.Search a bag.
5.Exit and clear.
Enter your choice: 4

Enter name to search:Tarun

Age:24  Name:Tarun      Destination:London      Flight No.:202  Bag Weight:33


Number of bags in the queue= 4
```

CHAPTER 6

CONCLUSION

This program displays the passengers in different flights with their respective destinations and bag weights. We then provide a number for flight in which the luggage is placed, hence not being misplaced. We can check in or check out using the luggage handler. We can also display all the entries stored in the Luggage Handler and view them to gain information. This project can be used in Airports, Railway Station, Ports and even in hotels as it can help the luggage reach the right hands in a much faster way.

CHAPTER 7

REFERENCES

1. https://en.wikipedia.org/wiki/Baggage_handling_systems

2.https://searchsqlserver.techtarget.com/definition/datastructure

3.https://www.studytonight.com/datastructures/introductiontodatastructures

4. https://en.wikipedia.org/wiki/Datastructure

5. Data Structures – A Pseudocode Approach with C

6. Data Structures using C – A.M PADMA REDDY