# PHONEBOOK MANAGEMENT SYSTEM USING DOUBLY LINKED LIST


## Submitted to:

## Professor Kauser Ahmed

## Data Structures and Algorithms

https://www.youtube.com/watch?v=mEYkd0C4HQo

## Review 3

## Submitted

## By Group

## 20

**RAHUL KOLAY-19BEC0169**
**DEEPANSHU SRIVASTAVA-19BEC0125**
**BRIJEESH THUNNOLY-19BEC0153**

# ABSTRACT:

Phone Book is a project that helps us to store our contacts. We can use it to replace our hard phonebook or even use it as an office-wide phone directory. This will help user to easily search and manage contacts using this system. Phonebook is the one, which contain details of an individual along with their mail id. This system is developed using the general need required by the user while using the phone directory book. In order to keep updated the phone book directory, the admin will have the authority to add and delete as well as modify the existing records within the phone book directory. As an added security feature the user will be asked for entering a user id and password ,so that only the officials can exploit the data . The users of the directory will only have the authority to search any particular record and listing details of all available records. Admin will have the authority to perform various operations such as add customer records, search any particular record, delete record, modify existing record etc. To provide the search result within short interval of time optimized search algorithm code have been used that will able to provide the results within seconds, ie here we use double linked list instead of single linked list. To make all operations as easier as possible, user-friendly approach has been taken .into account by which users have to only give their answer during final confirmation to make their operations successful. The background processing system will take care of all processing task and maintain data integrity in order to reduce the redundancy of data. For searching operation, users will able to get any particular record using their contact

# KEYWORDS:

Doubly Linked List, creating nodes using structures, creating functions, traversing the list, Binary Search , exit.

# AIM:

The aim of this project is to create a PHONEBOOK MANAGEMENT SYSTEM using double linked list in C language. Using this program the user will be able to manage his/her contacts in a much safer and easier way when compared to the old methord of using books for managing their contacts.

# Objective:

The objectives of this project are:

a) To come up with a working algorithm for the problem.

b) To make the algorithm precise and functional.

c) To fine-tune the algorithm and hence reduce the time complexity and space complexity.

d) To make this program as simple as possible for user to work with.

# Applicability:

This algorithm may be of atmost use in storing the contact details of employs in a company with atmost privacy and ease . This becomes more secure as the user is asked for a security password to acess these details .

In addition to that, this program reduces the time and effort needed to search for any employ's data. Hence this program has a wide ranging applications in areas that involves the handiling of large number of datas .

For example this program can be used to maintain the contact details of students in VIT.

# Introduction:

Nowadays people are very busy with their work and they feel inconvenient with the old method of maintaining a phone book, as it consumes an immense amount of time in searching and editing. They also feel inconvenient because the contact details can be easily accessed by anyone. In this system, the above problem is overcome here, as the system keeps a record of the customer contact and makes it easier for exploiting these data with maximum security as only the person with correct user id and password can access it.

Contact Management System is based on the concept to generate Personal or an Organization's Contact Records and to add their records & update it. Here User can add Contact details safely and it's not time-consuming. This System makes easy to store records of each. The whole project is designed in 'C' language and different variables and strings have been used for the development of this project. This mini project is easy to operate and understand by the users. This is a simple mini project in C, built as a console application without using any graphics features.

# Alternate Methods and Their Drawbacks:

The alternate method is implementing the phonebook management system using single linked list algorithm.

The advantages of doubly linked list over single linked list are:

1. A doubly linked list can be traversed in both forward and backward direction.
2. The delete operation in doubly linked list is more efficient if pointer to the node to be deleted is given .
3. We can quickly insert a new node before a given node. In singly linked list, to delete a node, pointer to the previous node is needed. To get this previous node, sometimes the list is traversed. In doubly linked list, we can get the previous node using previous pointer.

# Proposed Method and Advantages of Proposed Method:

The Phonebook Management system is being implemented using doubly linked list queue. Linked list plays a good role in implementing program of any kind of management or organization where records are being managed.

Advantages of Linked List:-

1. Linked List is Dynamic data Structure .
2. Linked List can grow and shrink during run time.
3. Insertion and Deletion Operations are Easier

4. Faster Access time, can be expanded in constant time without memory overhead
5. Linear Data Structures such as Stack, Queue can be easily implemented using Linked list.

# Proposed System Architecture:

The User is a person who is creating his or her phonebook.

>The User enters his or her name.

>The program welcomes the user and shows the operation menu.

>The user creates his/her contact list by inserting the contact details.

>The insert function inserts the details to the list entered by the user.

>The display function displays the details.

>The delete function deletes particular details as wished by the user.

>The update function updates any details as wished by the user, it can be any parameter.

>The search function works with three parameters name, number and e-mail.

>The exit function is used to exit from the program.

# Implementation Details:

When this program starts the user will be asked to enter a username and password. If the password that the user entered is correct then they will be provided with the following operations

1.INSERT CONTACT: used to insert the following details:

A)NAME

B)NUMBER

C)G-MAIL

2.DISPLAY CONTACT LIST: used to display contacts.

3. UPDATE DETAILS: used to update an existing contact.

A.NAME

B.NUMBER

C.E-MAIL

4.DELETE CONTACT: deletes contact by name

5.DELETE CONTACT SAME NAME: deletes contacts whose names are same.

6.SEARCH: search any contact by:-

A. NAME

B. NUMBER

C. E-MAIL

7.EMERGENCY DIAL LIST: Display contacts in your city

8.EXIT: exit from the program

# Testing of Algorithm and Output:

```
#include<iostream>
#include<cstring>
#include<process.h>
using namespace std;
class dnode
{
   public:
      char number[50];
      char gmail[40];
      char name[30];
      char city[30];
      dnode *prev,*next;
      dnode(char n[],char r[],char g[],char c[])
{
```

```
strcpy(name,n);
                strcpy(number,r);
            strcpy(gmail,g);
            strcpy(city,c);
             next=NULL;
             prev=NULL;
}
        friend class dlist;
};
class dlist
{
   dnode *head,*temp,*ptr;

   dnode *ptr1, *ptr2, *dup;
   public:
   dnode *prevn;

   void insert();
   void deletecontact(char n[20]);
   void deletesamename();
   void searchbyname(char p[20]);
   void searchbynumber(char no[30]);
   void searchbygmail(char g[20]);
   void emergencydial(char c1[30]);


      void accept();
      void display();
      void update(char ch[10]);
      dlist()
            {
                  head=NULL;
                  temp=NULL;
                  ptr=NULL;
                  ptr1=NULL;
                  ptr2=NULL;
                  dup=NULL;
            }
};

void dlist::accept()
{
      char number[50];
      char gmail[40];
      char name[30];
      char city[30];
```

```
char ans;
      do
        {
    cout<<"ENTER NAME      :";
    cin>>name;

    cout<<"ENTER NUMBER    :";
    cin>>number;
    while(strlen(number)!=10)
    {
    cout<<"ENTER VALID NUMBER  :";
    cin>>number;
    }
    cout<<"ENTER E-MAIL    :";
    cin>>gmail;
    cout<<"ENTER CITY    :";
    cin>>city;
    temp=new dnode(name,number,gmail,city);
    if(head==NULL)
    {
       head=temp;
    }
    else
    {
      ptr=head;
                while(ptr->next!=NULL)
                {
                        ptr=ptr->next;
                }
                ptr->next=temp;
                temp->prev=ptr;
    }
            cout<<"\n\nDO YOU WANT TO CONTINUE?????????";
          cin>>ans;
          cout<<endl;
}while(ans=='y');

}
void dlist::display()
{
    ptr=head;
          while(ptr!=NULL)
          {
                        cout<<"\n\nNAME  ::\t"<<ptr->name;
                        cout<<"\nNUMBER::\t+91-"<<ptr->number;
                        cout<<"\nE-MAIL ID::\t"<<ptr->gmail;
```

```
cout<<"\nCITY::  \t"<<ptr->city;
                        ptr=ptr->next;
        }
}

    void dlist::insert()
    {
        accept();
    }

    void dlist::deletecontact(char s[20])
    {
        int c=0;
        ptr=head;
        while(ptr!=NULL)
        {
            if(strcmp(s,ptr->name)==0)
            {
                c=1;
                break;
            }
            else
            {
                c=2;
            }
            ptr=ptr->next;
        }
        if(c==1 && ptr!=head && ptr->next!=NULL)
        {
            ptr->prev->next=ptr->next;
           ptr->next->prev=ptr->prev;
           delete(ptr);
           cout<<"YOUR CONTACT IS SUCCESSFULLY DELETED\n\n";
        }
        if(ptr==head)
        {
            head=head->next;
            head->prev=NULL;
            delete(ptr);
           cout<<"YOUR CONTACT IS SUCCESSFULLY DELETED\n\n";
        }
        if(ptr->next==NULL)
        {
            ptr->prev->next=NULL;
            ptr->prev=NULL;
            delete(ptr);
```

```
cout<<"YOUR CONTACT IS SUCCESSFULLY DELETED\n\n";
    }
    if(c==2)
    {
       cout<<"YOUR ENTERED NAME IS NOT IN THE LIST...";
    }
 }
 void dlist::deletesamename()
 {
    ptr1=head;
    while (ptr1 != NULL && ptr1->next != NULL)
 {
    ptr2 = ptr1;
    while (ptr2->next != NULL)
    {
       if (strcmp(ptr1->name,ptr2->next->name)==0)
       {
          dup = ptr2->next;
          ptr2->next = ptr2->next->next;
          delete(dup);
       }
       else
       {
          ptr2 = ptr2->next;
    }
    }
    ptr1 = ptr1->next;
 }
 }
 void dlist::searchbyname(char na[10])
 {
    ptr=head;
    while(ptr!=NULL)
    {
       if(strcmp(na,ptr->name)==0)
       {
          cout<<"\nNAME FOUND"<<endl;
          cout<<"CONTACT DETAILS ARE BELOW:\n"<<endl;
      cout<<"\n\nNAME  ::\t"<<ptr->name;
                    cout<<"\nNUMBER::\t+91-"<<ptr->number;
                    cout<<"\nE-MAIL ID::\t"<<ptr->gmail;

       }
       else
       {
```

```
cout<"\nNAME NOT FOUND";
                        }
          ptr=ptr->next;
       }
    }
    void dlist::searchbynumber(char num[20])
    {
     ptr=head;
       while(ptr!=NULL)
       {
          if(strcmp(num,ptr->number)==0)
          {
             cout<<"\nNUMBER FOUND\n"<<endl;
             cout<<"CONTACT DETAILS ARE BELOW:\n"<<endl;
         cout<<"\n\nNAME  ::\t"<<ptr->name;
                        cout<<"\nNUMBER::\t+91-"<<ptr->number;
                        cout<<"\nE-MAIL ID::\t"<<ptr->gmail;

          }
          else
          {
                cout<"NUMBER NOT FOUND";
                        }
          ptr=ptr->next;
       }
    }
    void dlist::searchbygmail(char gm[20])
    {
       ptr=head;
       while(ptr!=NULL)
       {
          if(strcmp(gm,ptr->gmail)==0)
          {
             cout<<"\nE-MAIL ID FOUND\n"<<endl;
             cout<<"CONTACT DETAILS ARE BELOW:\n"<<endl;
         cout<<"\n\nNAME  ::\t"<<ptr->name;
                        cout<<"\nNUMBER::\t+91-"<<ptr->number;
                        cout<<"\nE-MAIL ID::\t"<<ptr->gmail;

          }
          else
          {
                cout<"E-MAIL ID NOT FOUND";
                        }
          ptr=ptr->next;
```

```
}
   }
   void dlist::emergencydial(char ct[30])
   {
      ptr=head;
      while(ptr!=NULL)
      {
         if(strcmp(ct,ptr->city)==0)
         {
        cout<<"\n\nNAME  ::\t"<<ptr->name;
                        cout<<"\nNUMBER::\t+91-"<<ptr->number;
                        cout<<"\nE-MAIL ID::\t"<<ptr->gmail;
                        cout<<"\nCITY::\t"<<ptr->city;

         }
         ptr=ptr->next;
      }
   }
   void dlist::update(char n[20])
   {
      char ans;
      int c;
      ptr=head;
      while(ptr!=NULL)
      {
         if(strcmp(n,ptr->name)==0)
         {

            do
            {
               cout<<"\nWHAT DO YOU WANT TO UPDATE?\n1.NAME\n2.PHONE
NUMBER\n3.E-MAIL ID\n4.CITY\n";
            cin>>c;
            switch(c)
            {
               case 1:
               cout<<"\nENTER NEW-NAME: ";
               cin>>ptr->name;
               break;
               case 2:
               cout<<"\nENTER NEW PHONE-NUMBER: ";
               cin>>ptr->number;
               while(strlen(ptr->number)!=10)
   {
   cout<<"ENTER VALID NUMBER  :";
```

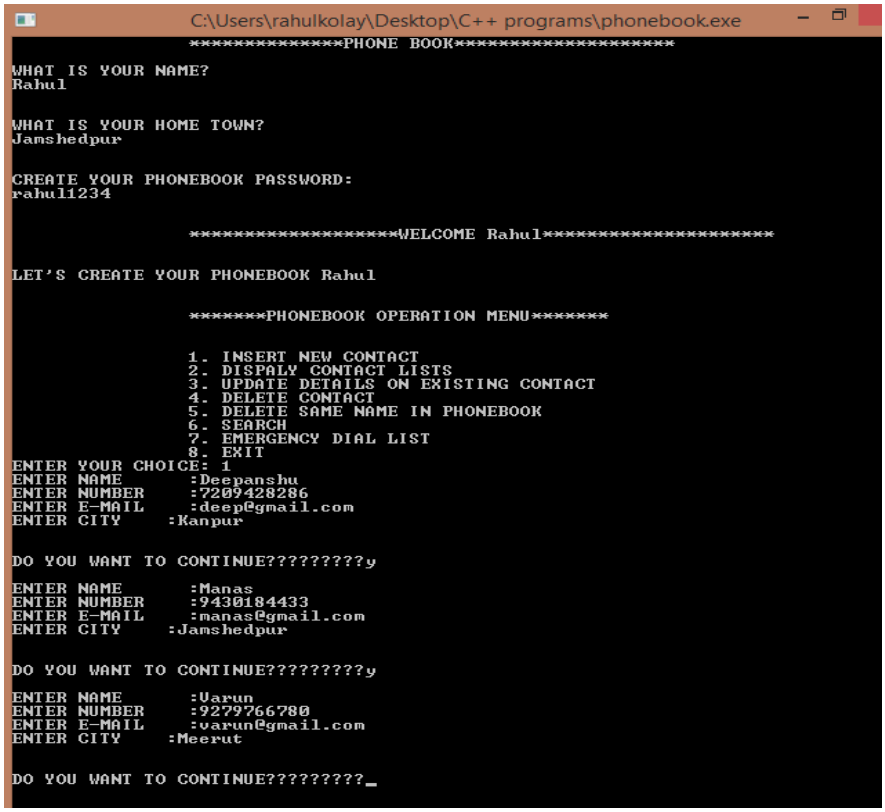
```
 cin>>ptr->number;
   }
                break;
                case 3:
                cout<<"\nENTER NEW E-MAIL ID: ";
                cin>>ptr->gmail;
                break;
                case 4:
                cout<<"\nENTER NEW CITY : ";
                cin>>ptr->city;
                break;
             }
             cout<<"\n\nDO YOU WANT TO CONTINUE UPDATING?";
             cin>>ans;
             }while(ans=='y');
          }
          ptr=ptr->next;
       }
   }
int main()
{
    char n[20];
    char nam[20];
    char name[10];
    char number[10];
    char gmail[20];
    char city[30];
    char password[30];
    dlist d1;
    char ans;
    int ch,a;
    cout<<"\t\t**************PHONE BOOK********************";
    cout<<"\n\nWHAT IS YOUR NAME?\n";
    cin.getline(name,20);
    cout<<"\n\nWHAT IS YOUR HOME TOWN?\n";
    cin.getline(city,20);
    cout<<"\n\nCREATE YOUR PHONEBOOK PASSWORD:\n";
    cin.getline(password,20);
    cout<<"\n\n\t\t*******************WELCOME "<<name<<"********************";
    cout<<"\n\n\nLET'S CREATE YOUR PHONEBOOK "<<name<<"  \n\n";
    do
    {
    cout<<"\n\t\t*******PHONEBOOK OPERATION MENU*******";
    cout<<"\n\n\n\t\t1. INSERT NEW CONTACT\n";
    cout<<"\t\t2. DISPALY CONTACT LISTS\n";
    cout<<"\t\t3. UPDATE DETAILS ON EXISTING CONTACT\n";
```

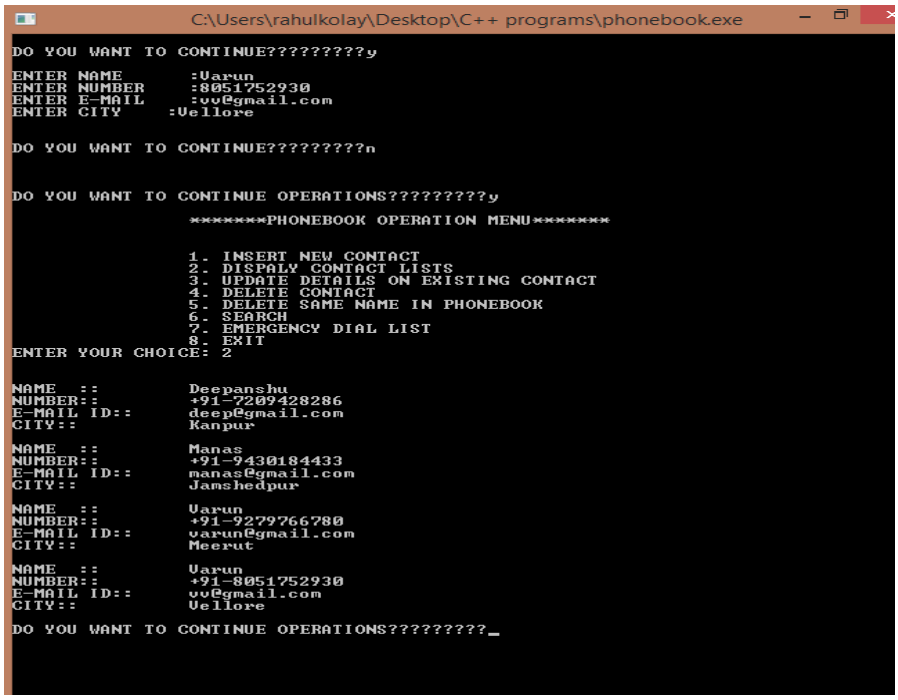
```
    cout<<"\t\t4. DELETE CONTACT\n";
    cout<<"\t\t5. DELETE SAME NAME IN PHONEBOOK\n";
    cout<<"\t\t6. SEARCH\n";
    cout<<"\t\t7. EMERGENCY DIAL LIST\n";
    cout<<"\t\t8. EXIT\n";
    cout<<"ENTER YOUR CHOICE: ";
    cin>>ch;
    switch(ch)
    {
    case 1:
    d1.insert();
          break;

    case 2:
    d1.display();
    break;
    case 3:
          char p0[30];
          cout<<"\nENTER YOUR PASSWORD TO ACCESS THE OPERATION :\n";
          cin>>p0;
          if(strcmp(p0,password)==0)
          {
                        cout<<"\n\nENTER THE NAME OF PERSON WHOSE DETAILS
YOU WANT TO UPDATE...\n";
          cin>>n;
          d1.update(n);

                  }
                  else
                  {
                        cout<<"INCORRECT PASSWORD\n";
                  }
    break;
    case 4:
          char p[30];
          cout<<"\nENTER YOUR PASSWORD TO ACCESS THE OPERATION :\n";
          cin>>p;
          if(strcmp(p,password)==0)
          {
                   cout<<"\nENTER THE NAME YOU WANT TO DELETE FROM
PHONEBOOK\n";
            cin>>name;
                   d1.deletecontact(name);
                  }
```

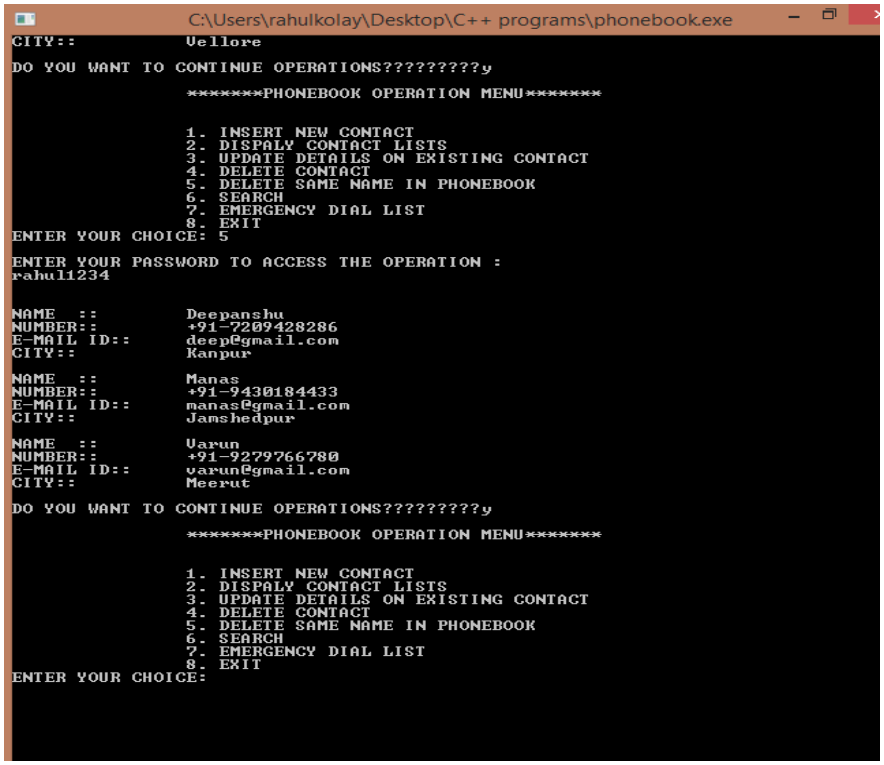
```
else
                {
                        cout<<"INCORRECT PASSWORD\n";
                }
   break;
   case 5:
        char p1[30];
        cout<<"\nENTER YOUR PASSWORD TO ACCESS THE OPERATION :\n";
        cin>>p1;
        if(strcmp(p1,password)==0)
        {
                 d1.deletesamename();
                d1.display();
                }
                else
                {
                        cout<<"INCORRECT PASSWORD\n";
                }
   break;
   case 6:
   do
   {
   cout<<"1.SEARCH BY NAME\n2.SEARCH BY NUMBER\n3.SEARCH BY E-MAIL
ID\n";
   cin>>a;
   switch(a)
   {
      case 1:
      cout<<"ENTER THE NAME TO BE SEARCHED\n";
      cin>>name;
      d1.searchbyname(name);
      break;
      case 2:
      cout<<"ENTER THE NUMBER TO BE SEARCHED\n";
      cin>>number;
      d1.searchbynumber(number);
      break;
      case 3:
      cout<<"ENTER THE E-MAIL ID TO BE SEARCHED\n";
      cin>>gmail;
      d1.searchbygmail(gmail);
      break;
      default:cout<<"\nNO PROPER INPUT GIVEN.....\n";
   }
```

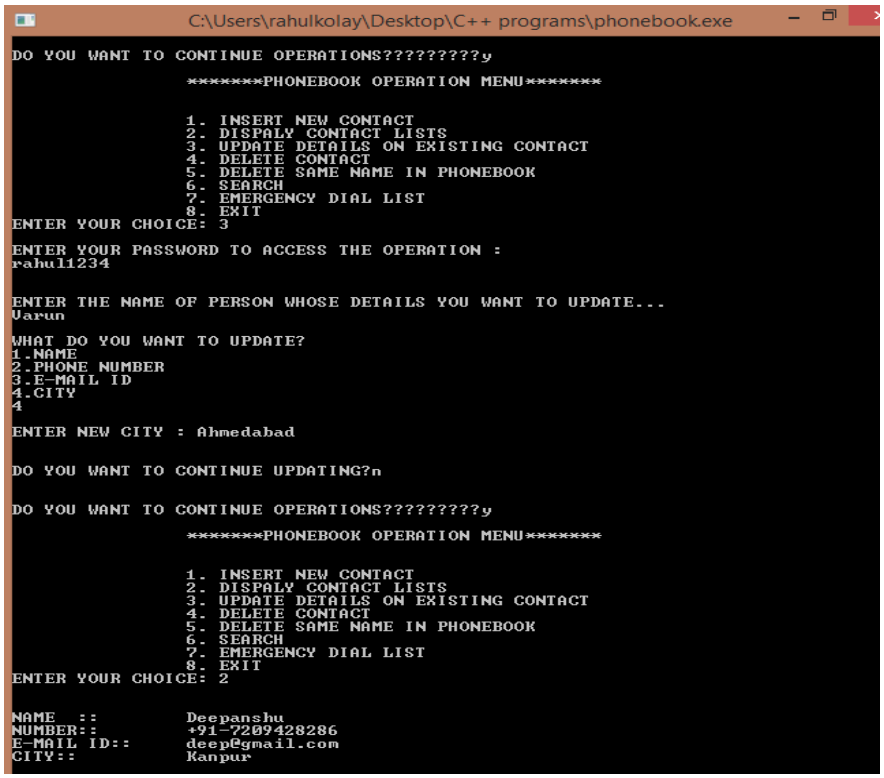
```
cout<<"\n\nDO YOU WANT TO CONTINUE SEARCHING?????????";
   cin>>ans;

}while(ans=='y');

   break;
   case 7:
        d1.emergencydial(city);
        break;

   case 8:
        cout<<"\n\t\t****THANK YOU***";
        exit(0);
   break;
   default:cout<<"\nNO PROPER INPUT GIVEN..\n";
   }
   cout<<"\n\nDO YOU WANT TO CONTINUE OPERATIONS?????????";
   cin>>ans;

}while(ans=='y');
}
```

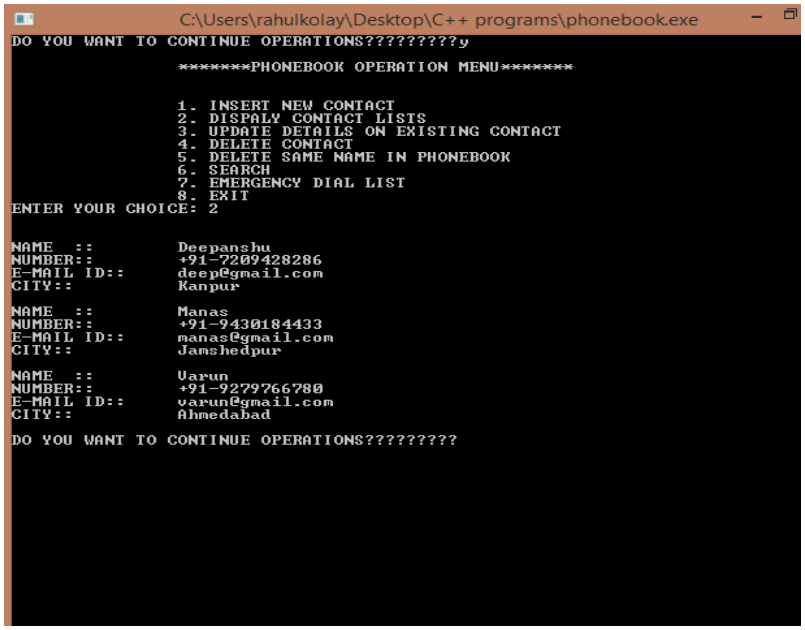
# OUTPUTS:-

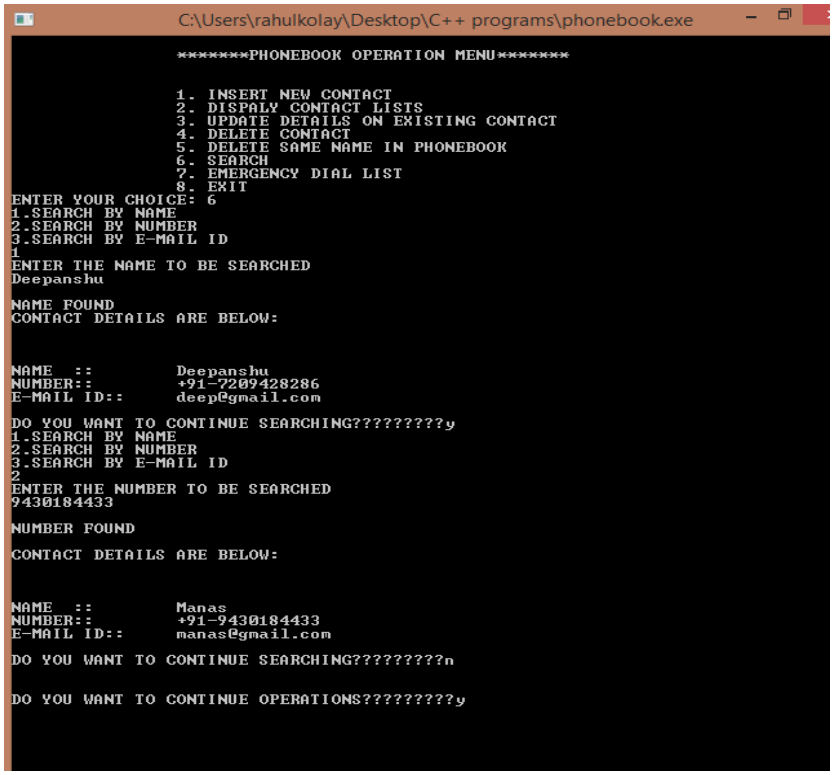

C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
**********PHONE BOOK**********
WHAT IS YOUR NAME?
Rahul
WHAT IS YOUR HOME TOWN?
Jamshedpur
CREATE YOUR PHONEBOOK PASSWORD:
rahul1234
**********WELCOME Rahul**********
LET'S CREATE YOUR PHONEBOOK Rahul
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 1
ENTER NAME :Deepanshu
ENTER NUMBER :7209428286
ENTER E-MAIL :deep@gmail.com
ENTER CITY :Kanpur
DO YOU WANT TO CONTINUE?????????y
ENTER NAME :Manas
ENTER NUMBER :9430184433
ENTER E-MAIL :manas@gmail.com
ENTER CITY :Jamshedpur
DO YOU WANT TO CONTINUE?????????y
ENTER NAME :Varun
ENTER NUMBER :9279766780
ENTER E-MAIL :varun@gmail.com
ENTER CITY :Meerut
DO YOU WANT TO CONTINUE?????????_

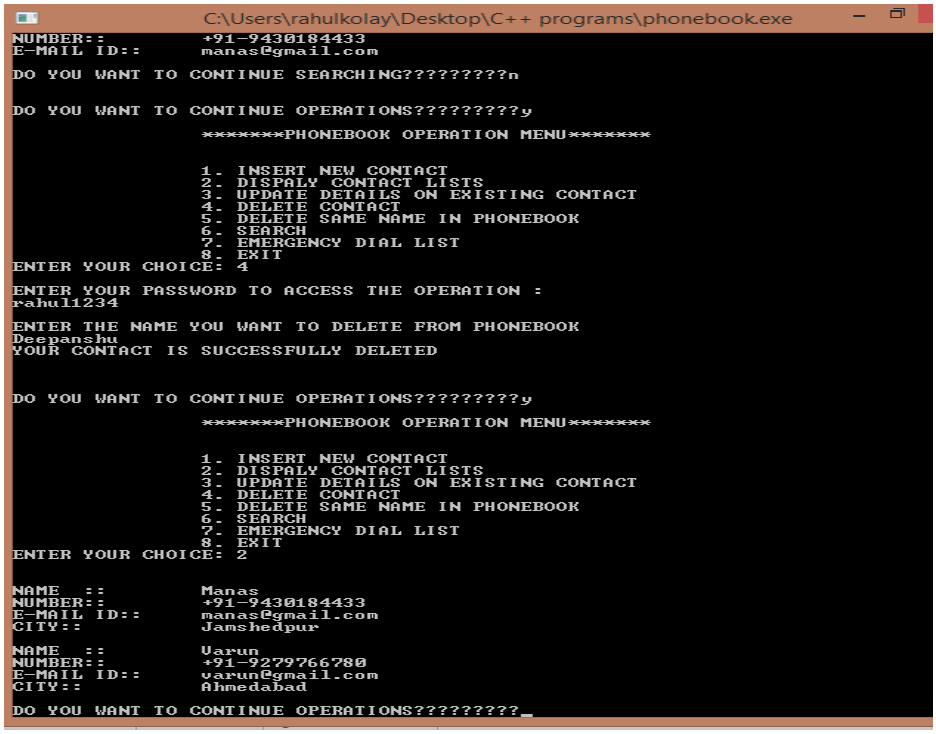

C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
DO YOU WANT TO CONTINUE?????????y
ENTER NAME :Varun
ENTER NUMBER :8051752930
ENTER E-MAIL :vv@gmail.com
ENTER CITY :Vellore
DO YOU WANT TO CONTINUE?????????n
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 2
NAME :: Deepanshu
NUMBER:: +91-7209428286
E-MAIL ID:: deep@gmail.com
CITY:: Kanpur
NAME :: Manas
NUMBER:: +91-9430184433
E-MAIL ID:: manas@gmail.com
CITY:: Jamshedpur
NAME :: Varun
NUMBER:: +91-9279766780
E-MAIL ID:: varun@gmail.com
CITY:: Meerut
NAME :: Varun
NUMBER:: +91-8051752930
E-MAIL ID:: vv@gmail.com
CITY:: Vellore
DO YOU WANT TO CONTINUE OPERATIONS?????????_

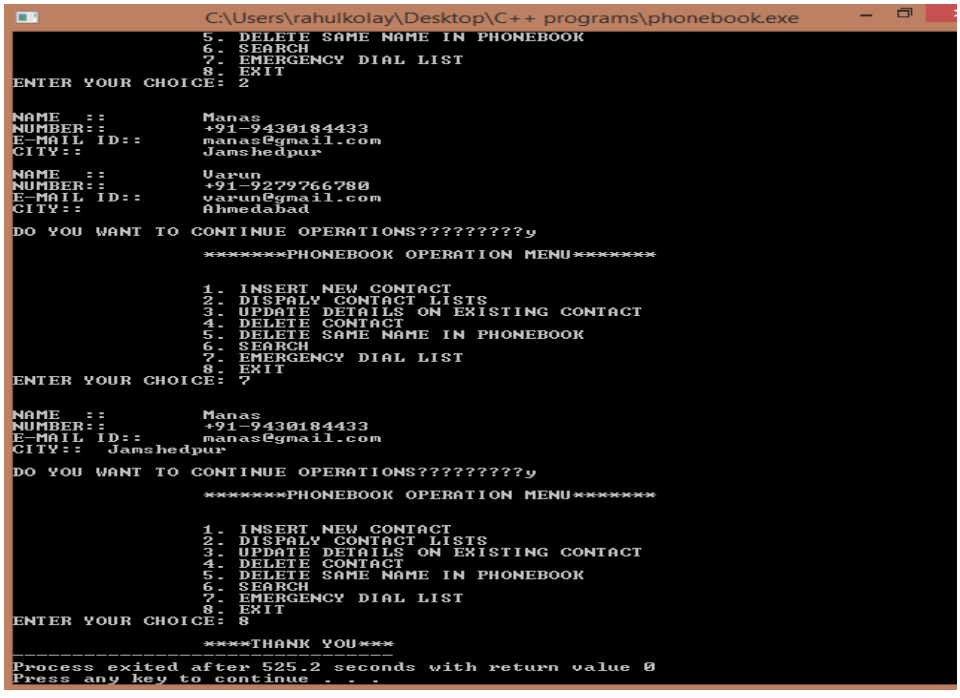

C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
CITY:: Vellore
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 5
ENTER YOUR PASSWORD TO ACCESS THE OPERATION :
rahul1234
NAME :: Deepanshu
NUMBER:: +91-7209428286
E-MAIL ID:: deep@gmail.com
CITY:: Kanpur
NAME :: Manas
NUMBER:: +91-9430184433
E-MAIL ID:: manas@gmail.com
CITY:: Jamshedpur
NAME :: Varun
NUMBER:: +91-9279766780
E-MAIL ID:: varun@gmail.com
CITY:: Meerut
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE:


C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 3
ENTER YOUR PASSWORD TO ACCESS THE OPERATION :
rahul1234
ENTER THE NAME OF PERSON WHOSE DETAILS YOU WANT TO UPDATE...
Varun
WHAT DO YOU WANT TO UPDATE?
1.NAME
2.PHONE NUMBER
3.E-MAIL ID
4.CITY
4
ENTER NEW CITY : Ahmedabad
DO YOU WANT TO CONTINUE UPDATING?n
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 2
NAME :: Deepanshu
NUMBER:: +91-7209428286
E-MAIL ID:: deep@gmail.com
CITY:: Kanpur


C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 2
NAME :: Deepanshu
NUMBER:: +91-7209428286
E-MAIL ID:: deep@gmail.com
CITY:: Kanpur
NAME :: Manas
NUMBER:: +91-9430184433
E-MAIL ID:: manas@gmail.com
CITY:: Jamshedpur
NAME :: Varun
NUMBER:: +91-9279766780
E-MAIL ID:: varun@gmail.com
CITY:: Ahmedabad
DO YOU WANT TO CONTINUE OPERATIONS?????????


C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 6
1.SEARCH BY NAME
2.SEARCH BY NUMBER
3.SEARCH BY E-MAIL ID
1
ENTER THE NAME TO BE SEARCHED
Deepanshu
NAME FOUND
CONTACT DETAILS ARE BELOW:
NAME :: Deepanshu
NUMBER:: +91-7209428286
E-MAIL ID:: deep@gmail.com
DO YOU WANT TO CONTINUE SEARCHING?????????y
1.SEARCH BY NAME
2.SEARCH BY NUMBER
3.SEARCH BY E-MAIL ID
2
ENTER THE NUMBER TO BE SEARCHED
9430184433
NUMBER FOUND
CONTACT DETAILS ARE BELOW:
NAME :: Manas
NUMBER:: +91-9430184433
E-MAIL ID:: manas@gmail.com
DO YOU WANT TO CONTINUE SEARCHING?????????n
DO YOU WANT TO CONTINUE OPERATIONS?????????y


C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
NUMBER::  +91-9430184433
E-MAIL ID::  manas@gmail.com
DO YOU WANT TO CONTINUE SEARCHING?????????n
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 4
ENTER YOUR PASSWORD TO ACCESS THE OPERATION :
rahul1234
ENTER THE NAME YOU WANT TO DELETE FROM PHONEBOOK
Deepanshu
YOUR CONTACT IS SUCCESSFULLY DELETED
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 2
NAME  ::  Manas
NUMBER::  +91-9430184433
E-MAIL ID::  manas@gmail.com
CITY::  Jamshedpur
NAME  ::  Varun
NUMBER::  +91-9279766780
E-MAIL ID::  varun@gmail.com
CITY::  Ahmedabad
DO YOU WANT TO CONTINUE OPERATIONS?????????


C:\Users\rahulkolay\Desktop\C++ programs\phonebook.exe
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 2
NAME  ::  Manas
NUMBER::  +91-9430184433
E-MAIL ID::  manas@gmail.com
CITY::  Jamshedpur
NAME  ::  Varun
NUMBER::  +91-9279766780
E-MAIL ID::  varun@gmail.com
CITY::  Ahmedabad
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 7
NAME  ::  Manas
NUMBER::  +91-9430184433
E-MAIL ID::  manas@gmail.com
CITY::  Jamshedpur
DO YOU WANT TO CONTINUE OPERATIONS?????????y
*******PHONEBOOK OPERATION MENU*******
1. INSERT NEW CONTACT
2. DISPALY CONTACT LISTS
3. UPDATE DETAILS ON EXISTING CONTACT
4. DELETE CONTACT
5. DELETE SAME NAME IN PHONEBOOK
6. SEARCH
7. EMERGENCY DIAL LIST
8. EXIT
ENTER YOUR CHOICE: 8
****THANK YOU***
Process exited after 525.2 seconds with return value 0
Press any key to continue . . .

# Conclusion :

In this program we have added two new than we have did it in our review 2.

1. We have added the home town of the user and the home town of the contacts so that in case of emergency those contacts will be shown who lives in the home town of the user.

2.We have added a password for the user which will be needed to access the update and delete contact operation.

.

# Note:

Sir, we request you to see the video in high quali

https://www.youtube.com/watch?v=mEYkd0C4HQo