

**DELHI TECHNOLOGICAL UNIVERSITY (DTU)**

**---------------------------------------------------------**

**DELHI COLLEGE OF ENGINEERING (DCE)**

Since **1941**

Submitted by,

***Sonu Kumar Kushwaha***

**2k19/CO/383**

*(Student of Computer Engineering, 3rd Semester, Batch A6)*

**CERTIFICATE**

This is hereby to certify that the original and genuine Innovative Work (Project) has been carried out to investigate about the subject matter and the related data. The Project has been sincerely and satisfactorily done by,

**Sonu Kumar Kushwaha**

**2k19/CO/383**

*(Student od Computer Engineering, 3rd semester, batch A6, academic year 2019-2022)*

This Innovative Project entitled **“PHONEBOOK”** is completed the guidelines of **“Mrs. Diksha Ruhela Ma’am”** as per the requirement of Delhi Technological University (DTU) **,** including all the contents required to justify and investigate the matters of Project.

**Object Oriented Programming**

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or [objects](https://searchapparchitecture.techtarget.com/definition/object), rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior.

OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. This approach to programming is well-suited for [programs](https://searchsoftwarequality.techtarget.com/definition/program) that are large, complex and actively updated or maintained.

Object-oriented programming is based on the following principles:

* [Encapsulation](https://searchnetworking.techtarget.com/definition/encapsulation). The implementation and state of each object are privately held inside a defined boundary, or class. Other objects do not have access to this class or the authority to make changes but are only able to call a list of public functions, or methods. This characteristic of [data hiding](https://searchsqlserver.techtarget.com/definition/data-hiding) provides greater program security and avoids unintended [data corruption](https://searchsqlserver.techtarget.com/definition/data-corruption).
* [Abstraction](https://whatis.techtarget.com/definition/abstraction). Objects only reveal internal mechanisms that are relevant for the use of other objects, hiding any unnecessary [implementation](https://searchcustomerexperience.techtarget.com/definition/implementation) code. This concept helps developers more easily make changes and additions over time.
* [Inheritance](https://whatis.techtarget.com/definition/inheritance). Relationships and subclasses between objects can be assigned, allowing developers to reuse a common logic while still maintaining a unique hierarchy. This property of OOP forces a more thorough data analysis, reduces development time and ensures a higher level of accuracy.
* [Polymorphism](https://whatis.techtarget.com/definition/polymorphism). Objects can take on more than one form depending on the context. The program will determine which meaning or usage is necessary for each execution of that object, cutting down the need to duplicate code.

**ACKNOWLEDGEMENT**

First of all, I am thankful to **Delhi Technological University (DTU)** to have this Project as a beautiful opportunity for some Study and research on my behalf. This Project is definitely going to help me to improve my learning skills and the way to represent them.

Discrete Structures is a subject of much confusions, but project led me to know the deep knowledge contained in this subject. A special thanks to **Mrs. Diksha Ruhela Ma’am** (Object Oriented Programming Teacher) for lending me this golden opportunity as a project.

This project is a great utilization of the valuable time in this intense condition of Lockdown. Nothing could have been better than this project for the Marking of Mid Semester Examination because the physical examination is currently not possible. This innovative idea of DTU is really appreciated by other Universities too. I am thankful to study in DTU which makes great decisions in ever condition and features the students with top quality education.

**PHONEBOOK Application**

**Abstract**

Phonebook is a console application without graphics. Phonebook is a very simple project that can help to understand the basic concepts of functions, file handling, classes and objects. This application will teach how to add, list, modify or edit, search and delete data to/from the file.

Adding new records, listing them, modifying them and updating, search for contacts saved, and deleting the phonebook records are the basic functions which make up the main menu of this Phonebook application. It becomes easy for the user to store complete information (i.e. e-mail id, address) about his contact. It is easy for the user to just search his required contact number by just typing name of the contact.

Personal information such as name, sex, father’s name, phone number, citizenship number, email and address are asked while adding a record into the Phonebook. These records can then be modified, listed, searched for and removed. This program is very useful now-a-days to store complete information under single contact number. This program also has options to deletion and modification of the entered contact number.

**Table of Contents Page No.**

**Introduction................................................................. 06**

**Features...................................................................... 06-07**

**Approach..................................................................... 07-10**

**Technical Documents................................................... 10-22**

**Results......................................................................... 23-26**

**Conclusion................................................................... 26**

**References.................................................................. 26-27**

**Introduction:**

Phonebook is a very simple project that can help you understand the basic concepts of functions, file handling and data structure. This application will teach you how to add, list, modify or edit, search and delete data to/from the file.

Adding new records, listing them, modifying them and updating, search for contacts saved, and deleting the phonebook records are the basic functions which make up the main menu of this Phonebook application.

Personal information such as name, sex, father’s name, phone number, citizenship number, email and address are asked while adding a record into the Phonebook. These records can then be modified, listed, searched for and removed.

**Features:**

**1. Add phone record**: This features adds a new phone record to the file. It asks for the phone number and name of the person whose record is to be created. Upon successful creation of a new phone record, the phonebook management system program displays the message – “Phone details successfully stored.” If the new phone record already exists in the file, it displays the message – “Phone id already exists! Try again…”

**2. Search phone records**: This function is very simple, and with it users can search for a phone record either by providing the phone number or by the name of a particular person whose record has already been added in the file.

**3. Show phone records**: In phonebook management system, this feature allows user to list all the phone records stored in file. The information displayed here are phone record number, phone number and the person’s name.

**4. Modify phone record**: This simple feature allows user to edit the phone number and person name corresponding to a particular phone record. Upon successful modification, the program displays the message – “Record modified successfully!”

**5. Delete phone record**: This feature deletes added phone record from the file. The user needs to provide the phone number to be deleted from phonebook management system. Upon successful deletion, it displays the message – “Record deleted!”. If the phone number provided is not found in file, the program displays the message – “Record doesn’t exist! Try again…”

**APPROACH:**

I preferred C++ programming language because I wanted to learn a system-oriented programming language. Java and C# are not system-oriented and can therefore be hacked more easily through reverse engineering. Also, I didn't choose C# because I don't use Windows except at work.

I also find C++ very interesting due to its universal applicability, because due to its proximity to the system, every C++ program runs on every device with a few adjustments. I also wanted to take on this challenge, as C++ is very complex and has a steep learning curve. I see more benefit for me personally in learning a complex programming language like C++ than an easy-to-learn programming language.

When choosing the programming language, neither trends nor the TIOBE index played a role. Because if everyone were to choose the programming language that is currently very popular and sought after, then the diversity would be lost.

I have used many functions in this project. These functions are easy to understand as their name only signifies their respective operations.

* void menu() – This function is used to display the main menu.
* void start() – This functions calls the menu function mentioned above.
* void got() – This function is use defined input function.
* void back() – This function is used to go back to start.
* void addrecord() – It adds a new Phonebook record.
* void listrecord() – This function is used to view list of added records in file.
* void modifyrecord() – This function is used to modify added records.
* void deleterecord() – It deletes record from file.
* void searchrecord() – It searches for added record by name.

To make output screen of PhoneBook application attractive, I used **system("color 5f")** to change the background screen to Purple.

Different other colors can be used as: -

Color attributes are specified by TWO hex digits -- the first  
 corresponds to the background; the second to the foreground. Each digit  
 can be any of the following values:  
 0 = Black  
 8 = Gray  
 1 = Blue  
 9 = Light Blue  
 2 = Green  
 A = Light Green  
 3 = Aqua  
 B = Light Aqua  
 4 = Red  
 C = Light Red  
 5 = Purple  
 D = Light Purple  
 6 = Yellow  
 E = Light Yellow  
 7 = White  
 F = BringWhite

The Header files used are: -

**<iostream>** - The C++ <iostream> header file declares a set of functions for standard Input/Output. It also defines I/O stream objects such as cin, cout, clog, etc.

**<cstdio>** - Input and Output operations can also be performed in C++ using the C Standard Input and Output Library (cstdio, known as stdio. h in the C language). This library uses what are called streams to operate with physical devices

**<conio.h>** - The conio. h is a non-standard header file used in C and C++ programming. This file contains console input-output functions which are mostly used by MS-DOS compilers.

**<cstring>**  - The C++ <cstring> header file declares a set of functions to work with C style string (null terminated byte strings).

**<windows.h>**  - h is a Windows-specific header file for the C++ programming languages which contains declarations for all of the functions in the Windows API, all the common macros used by Windows programmers, and all the data types used by the various functions and subsystems.

**<cstlib> -** This header defines several general-purpose functions, including dynamic memory management, random number generation, communication with the environment, integer arithmetics, searching, sorting and converting.

**got() function:-**

Why I used got() over cin() :

* Cin() stop reading a string after space but got() continues to read string even after space.
* Got() stops reading the string only after seeing a new line character.
* The entered character is immediately returned without waiting for the enter key.
* Got() does not use any buffer to store the input character.
* The entered character does not show up on the console.

Got() function uses input from **getch()**, which stores all type of data from keyboard in form of ASCII codes. There are two ASCII codes used in defining the got() function.

ASCII value 8 = Backspace

ASCII value 13 = Carriage Return (reset the position to beginning)

**Putch()** function is also used to display the value stored in got() input function.

*While I was searching for a new idea of input, I found the input process of* ***passwords*** *we commonly use in our daily life and I implemented that idea in my phonebook application.*

**Technical Documents:**

The C++ program for PhoneBook Application is: -

#include <iostream>

#include <cstdio>

#include <conio.h>

#include <cstring>

#include <cstlib>

#include <windows.h>

using namespace std;

class person

{

public:

char name[35];

char address[50];

char father\_name[35];

char mother\_name[30];

long int mble\_no;

char sex[8];

char mail[100];

char citision\_no[20];

};

void menu();

void got(char \*name);

void start();

void back();

void addrecord();

void listrecord();

void modifyrecord();

void deleterecord();

void searchrecord();

int main()

{

system("color 5f");

start();

return 0;

}

void back()

{

start();

}

void start()

{

menu();

}

void menu()

{

system("cls");

cout<<"\t\t\*\*\*\*\*\*\*\*\*\*WELCOME TO PHONEBOOK\*\*\*\*\*\*\*\*\*\*\*\*\*";

cout<<"\n\n\t\t\t MENU\t\t\n\n";

cout<<"\t1.Add New \t2.List \t3.Exit \n\t4.Modify \t5.Search \t6.Delete";

switch(getch())

{

case '1': addrecord();

break;

case '2': listrecord();

break;

case '3': exit(0);

break;

case '4': modifyrecord();

break;

case '5': searchrecord();

break;

case '6': deleterecord();

break;

default:

system("cls");

cout<<"\nEnter 1 to 6 only";

cout<<"\n Enter any key";

getch();

menu();

}

}

void addrecord()

{

system("cls");

FILE \*f;

class person p;

f=fopen("project","ab+");

cout<<"\n Enter name: ";

got(p.name);

cout<<"\nEnter the address: ";

got(p.address);

cout<<"\nEnter father name: ";

got(p.father\_name);

cout<<"\nEnter mother name: ";

got(p.mother\_name);

cout<<"\nEnter phone no.:";

cin>>p.mble\_no;

cout<<"Enter sex:";

got(p.sex);

cout<<"\nEnter e-mail:";

got(p.mail);

cout<<"\nEnter citizen no:";

got(p.citision\_no);

fwrite(&p,sizeof(p),1,f);

fflush(stdin);

cout<<"\nrecord saved";

fclose(f);

cout<<"\n\nEnter any key";

getch();

system("cls");

menu();

}

void listrecord()

{

class person p;

FILE \*f;

f=fopen("project","rb");

if(f==NULL)

{

cout<<"\nCONTACT'S DATA NOT ADDED YET.";

}else {

while(fread(&p,sizeof(p),1,f)==1)

{

cout<<"\n\n\n YOUR RECORD IS\n\n ";

cout<<"\nName="<<p.name<<"\nAdress="<<p.address<<"\nFather name="<<p.father\_name<<"\nMother name="<<p.mother\_name<<"\nMobile no="<<p.mble\_no<<"\nSex="<<p.sex<<"\nE-mail="<<p.mail<<"\nCitizen no="<<p.citision\_no;

getch();

system("cls");

}

}

fclose(f);

cout<<"\n Enter any key";

getch();

system("cls");

menu();

}

void searchrecord()

{

class person p;

FILE \*f;

char name[100];

f=fopen("project","rb");

if(f==NULL)

{

cout<<"\nCONTACT'S DATA NOT ADDED YET.";

}

cout<<"\nEnter name of person to search\n";

got(name);

while(fread(&p,sizeof(p),1,f)==1)

{

if(strcmp(p.name,name)==0)

{

cout<<"\n\tDetail Information About"<<name;

cout<<"\nName="<<p.name<<"\nAdress="<<p.address<<"\nFather name="<<p.father\_name<<"\nMother name="<<p.mother\_name<<"\nMobile no="<<p.mble\_no<<"\nSex="<<p.sex<<"\nE-mail="<<p.mail<<"\nCitizen no="<<p.citision\_no;

}

else

cout<<"\n file not found";

}

fclose(f);

cout<<"\n Enter any key";

getch();

system("cls");

menu();

}

void deleterecord()

{

class person p;

FILE \*f,\*ft;

int flag;

char name[100];

f=fopen("project","rb");

if(f==NULL)

{

cout<<"\nCONTACT'S DATA NOT ADDED YET.";

}

else

{

ft=fopen("temp","wb+");

if(ft==NULL)

cout<<"\nfile opaning error";

else

{

system("cls");

cout<<"\nEnter CONTACT'S NAME:";

got(name);

fflush(stdin);

while(fread(&p,sizeof(p),1,f)==1)

{

if(strcmp(p.name,name)!=0)

fwrite(&p,sizeof(p),1,ft);

if(strcmp(p.name,name)==0)

flag=1;

}

fclose(f);

fclose(ft);

if(flag!=1)

{

cout<<"\nNO CONACT'S RECORD TO DELETE.";

remove("temp.txt");

}

else

{

remove("project");

rename("temp.txt","project");

cout<<"\nRECORD DELETED SUCCESSFULLY.";

}

}

}

cout<<"\n Enter any key";

getch();

system("cls");

menu();

}

void modifyrecord()

{

int c;

FILE \*f;

int flag=0;

class person p,s;

char name[50];

f=fopen("project","rb+");

if(f==NULL)

{

cout<<"\nCONTACT'S DATA NOT ADDED YET.";

}

else

{

system("cls");

cout<<"\nEnter CONTACT'S NAME TO MODIFY:\n";

got(name);

while(fread(&p,sizeof(p),1,f)==1)

{

if(strcmp(name,p.name)==0)

{

cout<<"\n Enter name: ";

got(s.name);

cout<<"\nEnter the address: ";

got(s.address);

cout<<"\nEnter father name: ";

got(s.father\_name);

cout<<"\nEnter mother name: ";

got(s.mother\_name);

cout<<"\nEnter phone no.:";

cin>>s.mble\_no;

cout<<"Enter sex:";

got(s.sex);

cout<<"\nEnter e-mail:";

got(s.mail);

cout<<"\nEnter citizen no:";

got(s.citision\_no);

fseek(f,sizeof(p),SEEK\_CUR);

fwrite(&s,sizeof(p),1,f);

flag=1;

break;

}

fflush(stdin);

}

if(flag==1)

{

cout<<"\n your data id modified";

}

else

{

cout<<" \n data is not found";

}

fclose(f);

}

cout<<"\n Enter any key";

getch();

system("cls");

menu();

}

void got(char \*name)

{

int i=0,j;

char c,ch;

do

{

c=getch();

if(c!=8&&c!=13)

{

\*(name+i)=c;

putch(c);

i++;

}

if(c==8)

{

if(i>0)

{

i--;

}

system("cls");

for(j=0;j<i;j++)

{

ch=\*(name+j);

putch(ch);

}

}

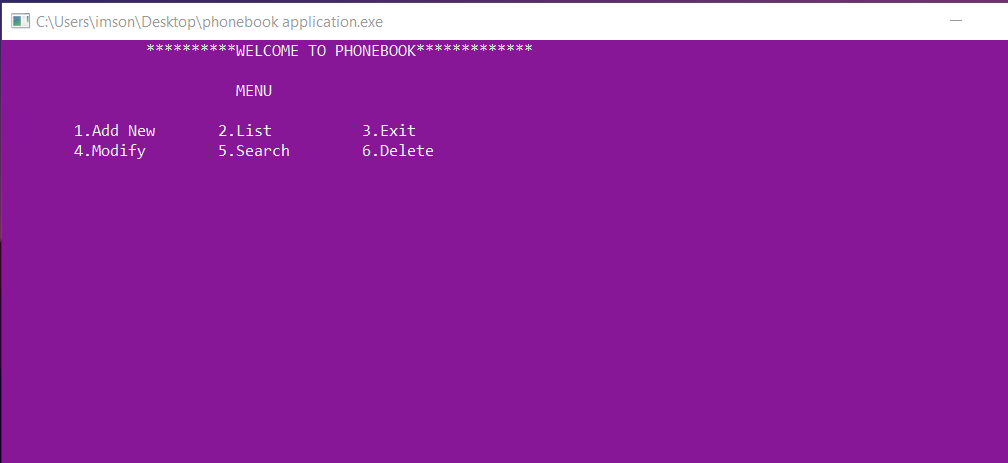
}while(c!=13);

\*(name+i)='\0';

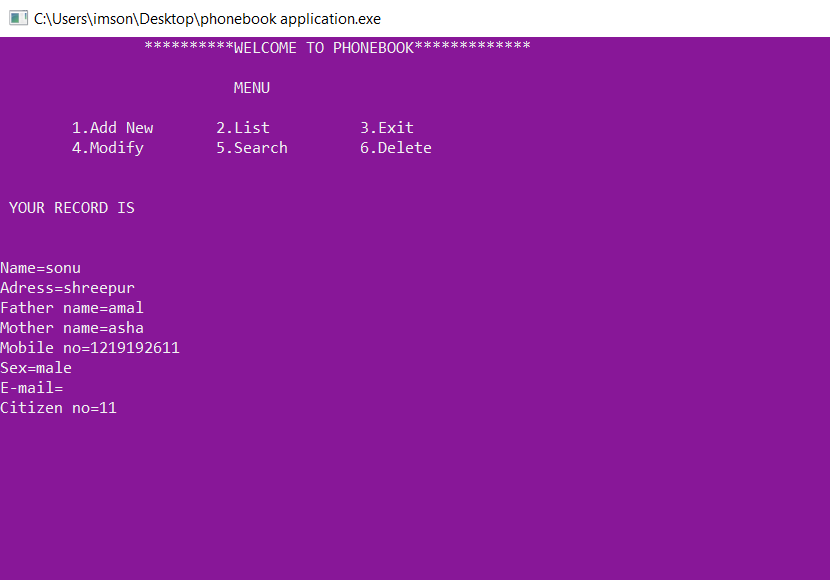
}

**RESULT:**

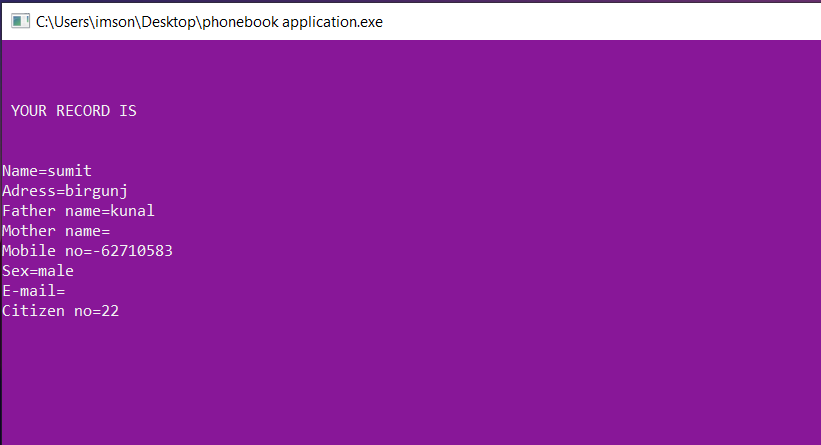
The Outputs of this PhoneBook Application program are: -



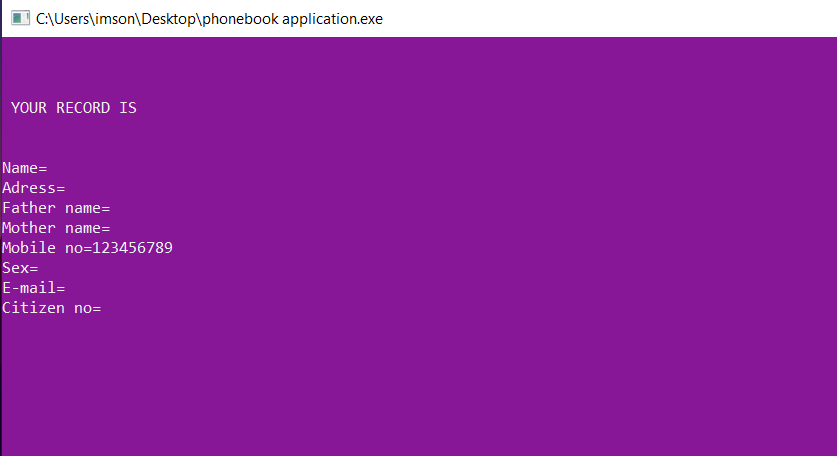
This is the **Menu** of the PhoneBook Application.



Added the first contact information with name **sonu**, and listed it.

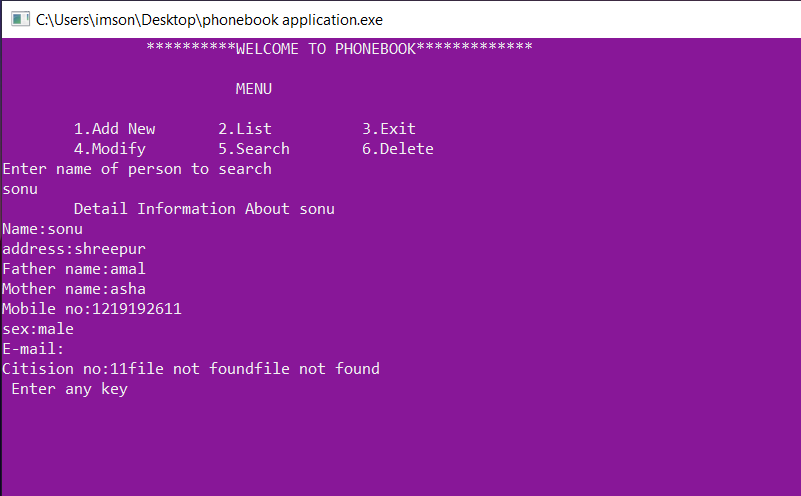


Added second contact information with name **sumit**, and listed it.

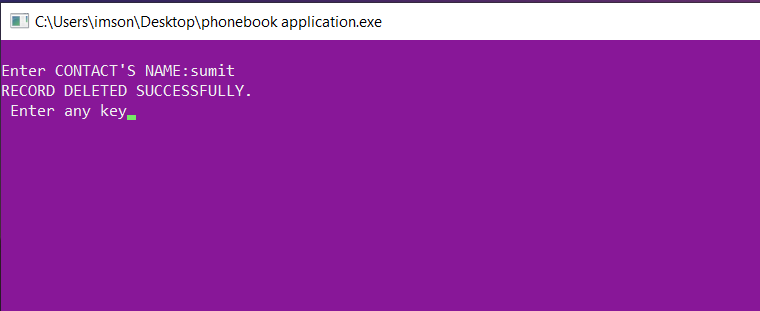


Added third contact information without any name, and listed it.

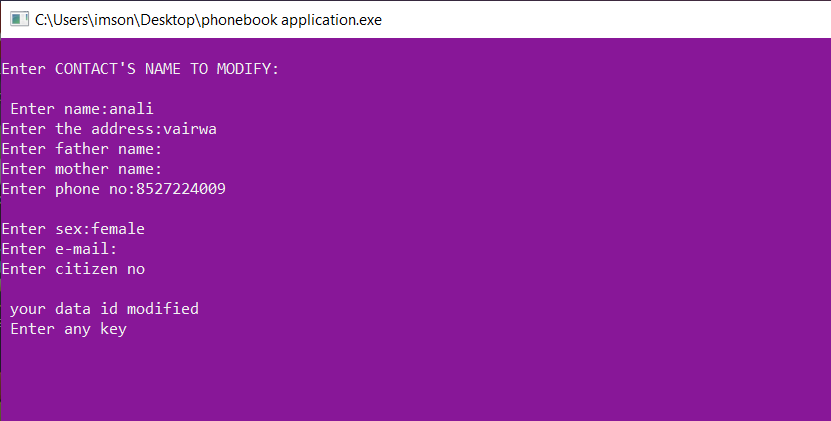
*All the other information can be kept blank except the* ***Phone no.***



Searched for details, giving the input name **sonu.**



Deleted the contact information of **sumit**.



Modified the contact information of “ “(blank) as **anjali**.

At last existed.

**CONCLUSION:**

PhoneBook is an innovative project that helps us to find an easy way to store all of our contacts. We can use it to replace our hard phonebook and even use it as office wide phonebook directory. This helps user to easily search and manage the contacts details. Apart from name and phone number this phonebook application also stores other type of data like address, email, etc. which could be useful to contact the person is phone number doesn't provide required communication.

**REFERRENCES:**

**Websites:**

* [www.slideshare.net](http://www.slideshare.net)
* theasciicode.com.ar
* [www.codesansar.com](http://www.codesansar.com)
* [www.go4expert.com](http://www.go4expert.com)
* [www.codewithc.com](http://www.codewithc.com)
* Narendraalianionline.blogspot.com

**YouTube:**

* Neso Academy
* Naresh i Technologies
* SKill Tech Edu
* Azeem ur Rehman Official