CO101 Project Report on

PHONEBOOK

submitted towards the partial fulfillment of
the requirement for the award of the degree of

Bachelor of Technology
in
SOFTWARE ENGINEERING
Submitted by

RAGHAV KUMAR JHA 2K20/B9/04
SANDALI SINGH 2K20/B9/16

Under the Supervision
of
Sir Ashish Girdhar

Department of Computer Science and Engineering
Delhi Technological University
Bawana Road. Delhi -110042

DECLARATION

We, (RAGHAV KUMAR JHA 2K20/B9/04, SANDALI SINGH 2K20/B9/16) students of B. Tech. (Software Engg.) hereby declare that the project titled ”PHONEBOOK” in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology and submitted to the Department of Computer Science and Engineering, Delhi Technological University, Delhi is carried out under the supervision of our subject teacher Ashsish Girdhar.

RAGHAV KUMAR JHA 2K20/B9/04
SANDALI SINGH 2K20/B9/16
Software Engineering

ACKNOWLEDGEMENT

In performing our major project, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. The completion of this assignment gives us much pleasure. We would like to show our gratitude to Ashish Girdhar, Mentor for the major project. Giving us a good guideline for report throughout numerous consultations. We thank all the people for their help directly and indirectly to complete our assignment. In addition, we would like to thank the Department of Computer Science and Engineering, Delhi Technological University for giving us the opportunity to work on this topic.

ABSTRACT

Phonebook is a very simple mini project in C. The concept of **file management, Data Structures and functions** are used in this project. It **adds new records, lists them, modifies them, searches them and deletes them.** These are the basic functions which make up the main menu of this phonebook application. Personal information such as **Name, Phone No., Email, Date of Birth, Gender, Mother's name, Father's name, Nationality and Address** are required to add a Record to the phonebook.It is easy to understand and simple to use. This Program is very useful nowadays to store complete information under a single contact number. This mini phonebook design allows you to perform simple tasks in your phonebook, such as mobile phones.

CONTENT

Name	Type	Size
phonebook	C source file	14 KB
phonebook	Application	62 KB
phonebook.o	O File	10 KB
project	File	3 KB

INTRODUCTION

To develop a 'PHONEBOOK' application using C programming.
Phonebook is a very simple mini project in C . The concepts of file handling, data structures and functions are extensively used in this project.Personal information like Name, Phone No, Email, Date of Birth, Gender, Mother's Name, Father's Name, Nationality and Address are required to add a record to the phonebook.You can do operations like insert a record , modify, list as well as you can also do a search and delete operations. There is an exit option as well.

DETAILS

This phonebook application is coded and made using Dev C and Code Blocks and GCC compiler. The application size is 61Kb and size of source file is 10Kb.

DESIGN

The present program consists of the following modules: Preprocessor commands, Structure, Function, Variables, Pointers, Array, Iterative Control Instructions i.e. Loops, Decision Control instructions i.e If-else, Switch-case Control Instructions, Statements and expressions and File Handling.

ALGORITHM

1. START
2. Print "WELCOME TO PHONEBOOK" and "Menu" on the screen.

a. Add contact, go to (3)
b. List contact, go to (4)
c. Modify contact, go to (5)
d. Search contact, go to (6)
e. Delete contact, go to (7)
f. Exit phonebook, to STOP

Take input from the user.

3. For Addrecord(), ask the user to Enter name, phone number, email id, Date of birth, gender, mother's name, father's name, nationality, address and write it in the file project. Enter any key to go to the menu.

4. For Listrecords(), arranging data as

a. If unable to open file, print error.

b. Else, display name, phone number, email id, DOB, gender, mother's name, father's name, nationality, address from the file.

Then, print enter any key to go to the main menu.

5. For Modifyrecord(),

a. If unable to open file, print error.

b. Else, Enter contact's name to modify

(I) If contact found in file

i. Input name, phone number, email id, date of birth, gender, mother's name, father's name, nationality, address from user. Rewrite the input data in place of the contact to be modified in the file.

ii. Then, display your data is modified

(II) Else, Print data is not found

Display enter any key to go to the main menu.

6. For searchrecord(),

a. If unable to open file, print error.

b. Else, input name of contact to be searched from user and

i. If contact found in file, display name, phone number, email id, date of birth, gender, mother's name, father's name, nationality, address of the contact

ii. Else, display contact not found.

Then, print enter any key to go to main menu

7. Now for deleterecord(),

a. If unable to open file, print error.

b.Else, input name of contact to be deleted from user

Initiate a loop from start to end of file project

i. If contact found, flag=1. Display the info of the contact.

ii. Else, copy the record to file temp

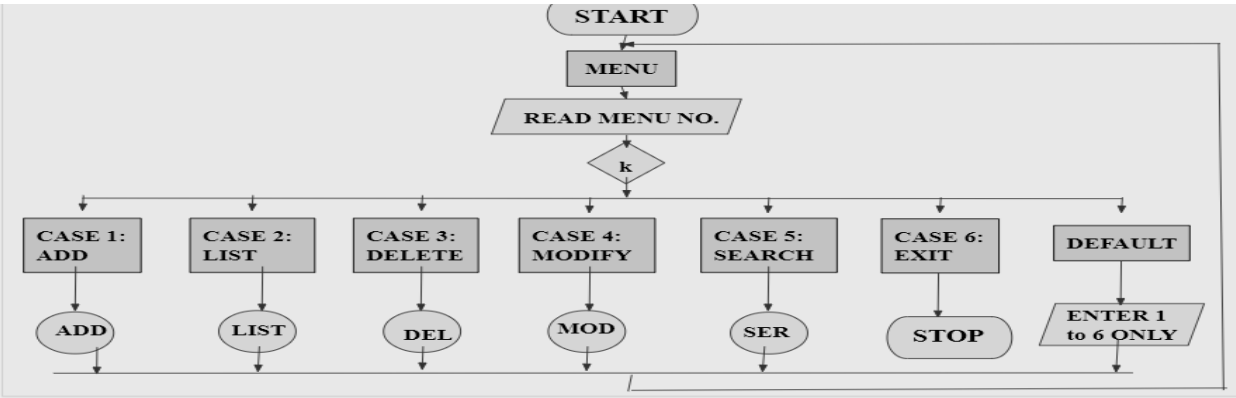
c. if (flag!=1)

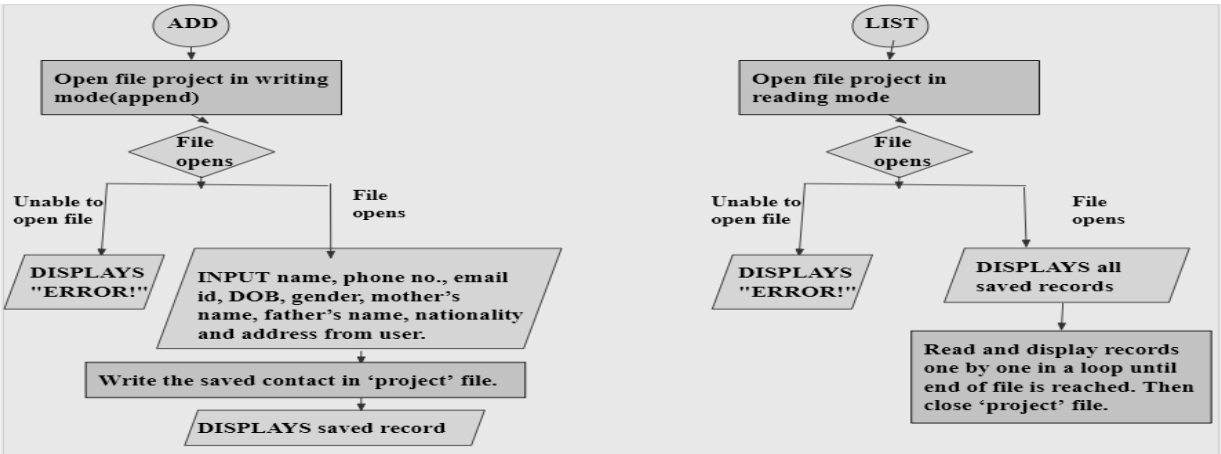
i. Print no contact's record to be deleted.

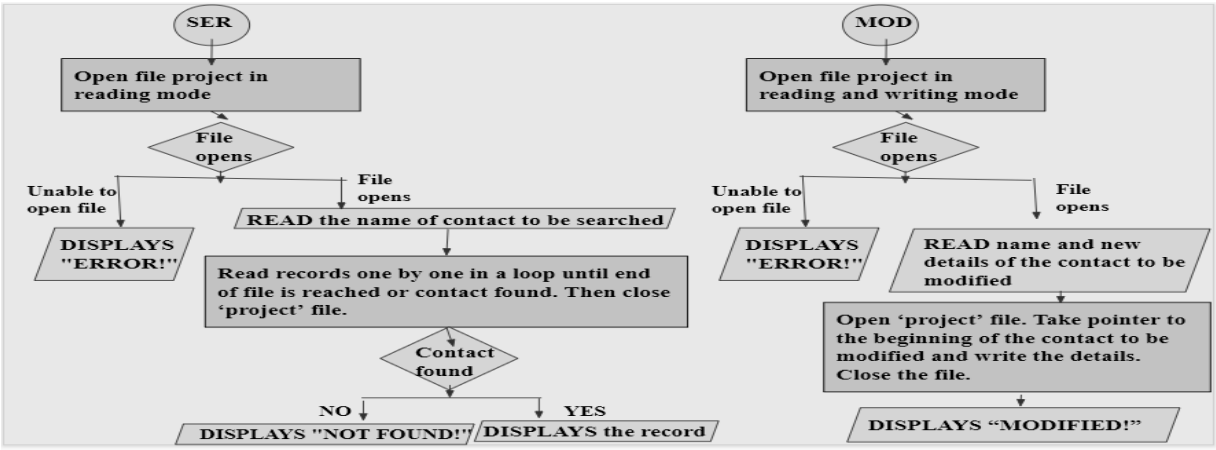
ii. Else, Print record deleted successfully. Delete the file project and rename temp to project file.

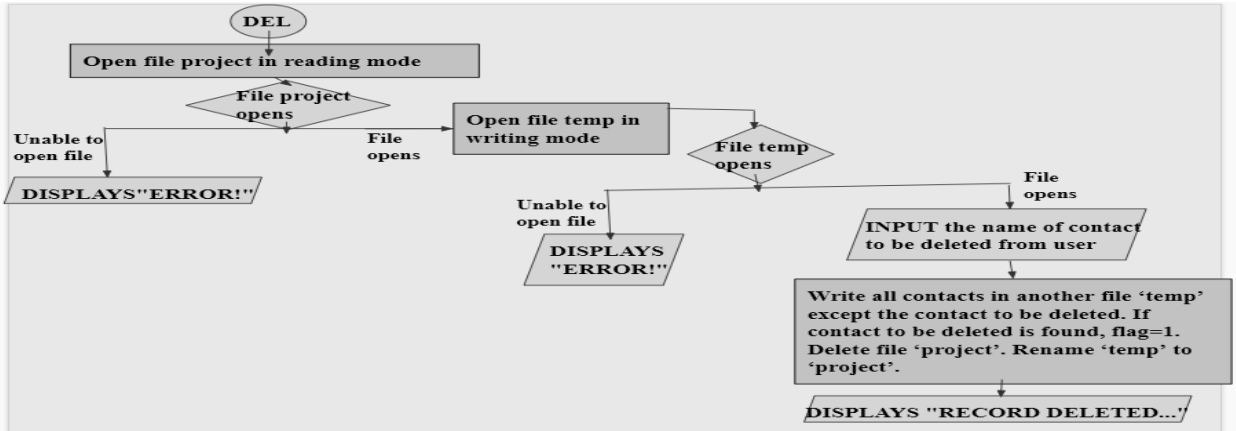
d. Then, ask the user to press any key to go to the menu.

FLOWCHART


START
MENU
READ MENU NO.
k
CASE 1: ADD
CASE 2: LIST
CASE 3: DELETE
CASE 4: MODIFY
CASE 5: SEARCH
CASE 6: EXIT
DEFAULT
ADD
LIST
DEL
MOD
SER
STOP
ENTER 1 to 6 ONLY


ADD
Open file project in writing mode(append)
File opens
Unable to open file
File opens
DISPLAYS "ERROR!"
INPUT name, phone no., email id, DOB, gender, mother's name, father's name, nationality and address from user.
Write the saved contact in 'project' file.
DISPLAYS saved record
LIST
Open file project in reading mode
File opens
Unable to open file
File opens
DISPLAYS "ERROR!"
DISPLAYS all saved records
Read and display records one by one in a loop until end of file is reached. Then close 'project' file.


SER
Open file project in reading mode
File opens
Unable to open file
File opens
DISPLAYS "ERROR!"
READ the name of contact to be searched
Read records one by one in a loop until end of file is reached or contact found. Then close 'project' file.
Contact found
NO
YES
DISPLAYS "NOT FOUND!"
DISPLAYS the record
MOD
Open file project in reading and writing mode
File opens
Unable to open file
File opens
DISPLAYS "ERROR!"
READ name and new details of the contact to be modified
Open 'project' file. Take pointer to the beginning of the contact to be modified and write the details. Close the file.
DISPLAYS "MODIFIED!"



USE OF HEADER FILES

- **stdio.h**- Standard input and output header file. It has definitions of functions like printf() and scanf(). It takes an input from the user and displays output either in a file or console. We also used fflush(), fclose(), fopen(),and remove().
- **string.h**- It contains macro definitions, constants, and declaration of functions and types used not only for string handling but also memory handling functions. We have used strcmp() to compare the input name and saved records in searchrecord() function.
- **windows.h**- It is a windows specific header file for the c and c++ programming languages which contains declarations for all the functions in windows API, all the common macros used by windows programmers, and all the data types used by the various functions and subsystems. We have used setcontrolcursorpostion() to display MENU at the center of the output screen.
- **conio.h**- We have used getch() in our program.
- **stdlib.h**- We have used exit() to stop the program when the user asks.

```
struct person
{
        char name[40];
        char mble_no[15];
        char mail[100];
        char DOB[15];
        char sex[8];
        char mother_name[40];
        char father_name[40];
        char nationality[10];
        char address[100];
};
```

```
int passcheck();  //PASSWORD

int menu();  //START MENU

int input(struct person *);  //Inputs record

int output(struct person);  //Displays record

int addrecord();  //ADDS RECORDS TO A FILE

int listrecord();  //LISTS THE SAVED RECORDS

int modifyrecord();  //MODIFIES A RECORD IF FOUND

int deleterecord();  //DELETES A RECORD IF FOUND

int searchrecord();  //SEARCHES A RECORD
```

SOURCE CODE

```c
#include<stdio.h> //printf(), scanf(), remove(), fopen(), fclose()
#include<string.h>  // strcmp(),strlen()
#include<windows.h>  //setcursorposition()
#include<conio.h>  //getch()
#include<stdlib.h>  //exit()


struct person
{
        char name[40];  //Name of person
        char mble_no[15];  //Phone number
        char mail[100];  //Email ID
        char DOB[15];  //Date Of Birth
        char sex[8];  //Gender
        char mother_name[40];  //Mother's Name
        char father_name[40];  //Father's Name
        char nationality[10];  //Nationality
        char address[100];   //Address
};

int passcheck();  //PASSWORD

int menu();  //START MENU

int input(struct person *);  //Inputs record

int output(struct person);  //Displays record

int addrecord();  //ADDS RECORDS TO A FILE

int listrecord();  //LISTS THE SAVED RECORDS

int modifyrecord();  //MODIFIES A RECORD IF FOUND

int deleterecord();  //DELETES A RECORD IF FOUND

int searchrecord();  //SEARCHES A RECORD


int main()
{
    system("color 87");  //SETS TEXT COLOR BLACK AND SCREEN COLOR GREY
    passcheck();
    menu();  //CALLS MENU
    return 0;
}


int menu()
{
    system("cls");  //CLEARS SCREEN

    COORD p;
    p.X=30;
    p.Y=5;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),p);

    printf("\t\tWELCOME TO PHONEBOOK");
    printf("\n\n\n\t\t\t\t\t\tMENU\n");
    printf("\n\t1.Add New \t2.List   \t3.Modify \t4.Search \t5.Delete \t6.Exit \t:\t");  //MENU

    int k;
    scanf("%d",&k);

    switch(k)
    {
      case 1: addrecord(); //TO ADD A RECORD
              break;
```

```c
        case 2: listrecord();   //TO OPEN THE LIST OF SAVED RECORDS
                  break;
        case 3: modifyrecord();   //TO MODIFY A RECORD
                  break;
        case 4: searchrecord();   //TO SEARCH A RECORD
                  break;
        case 5: deleterecord(); //TO DELETE A RECORD
                  break;
        case 6: exit(0);   //TO EXIT PHONEBOOK
                  break;
        default:     printf("\nWRONG INPUT!\t");
                     getch();
                     menu();
    }

    return 0;
}

int addrecord()
{
        system("cls");

        FILE *f, *fp;
        struct person p,k;
        f=fopen("project","r+");   //OPENS A FILE PROJECT IN READING MODE
       // fseek(f,0,SEEK_SET);   //POINTER TAKEN TO THE BEGINNING
        fp=fopen("temp","w");

        if(f==NULL)   //DISPLAYS AN ERROR MESSAGE IF PROJECT FILE DOESNT OPEN
        {
            printf("ERROR!\a");
            exit(1);
        }

```

```c
    if(fp==NULL)  //DISPLAYS AN ERROR MESSAGE IF TEMP FILE DOESNT OPEN
    {
        printf("ERROR!\a");
        exit(1);
    }

    printf("\nADDING A CONTACT");
    input(&p);  //inputs contact from user

    {
        int flag=0,m;

        while(m=fread(&k,sizeof(k),1,f))  //read one record at a time from project file
        {
            printf("1");
            if(strcmp(p.name,k.name)>0)  //p.name comes after k.name
              {
                  fwrite(&k,sizeof(k),1,fp);  //writing the read record in temp file
              }

            else if(((strcmp(p.name,k.name)<0)||(strcmp(p.name,k.name)==0))&&(flag==0))  //p.name and k.name are same or p.r
            {
                fwrite(&p,sizeof(p),1,fp);  //writing the input record in temp file
                flag=1;
                printf("\nSAVED...");
                fwrite(&k,sizeof(k),1,fp);  //writing the read record record in temp file
            }

            else if((strcmp(p.name,k.name)<0)&&(flag==1))  //when k.name comes after p.name and input record has been writte
              fwrite(&k,sizeof(k),1,fp);  //writing the rest of the records in temp file

        }

        if(m==0 && flag!=1)
```

```
137                 {
138                     fwrite(&p,sizeof(p),1,fp);
139                     printf("\nSAVED...");
140                 }
141         }
142
143                 fclose(fp);   //closes temp file
144                 fclose(f);   //CLOSES FILE PROJECT
145
146                 remove("project");   //REMOVES PROJECT FILE
147                 rename("temp","project");   //RENAMES TEMP FILE TO PROJECT FILE
148
149
150             fflush(stdin);   //CLEARS THE BUFFER
151             printf("\n\nEnter any key");
152             getch();
153
154             system("cls");   //CLEARS SCREEN
155             menu();   //OPENS MENU
156
157             return 0;
158     }
159
160     int listrecord()
161     {
162         struct person p;
163         FILE *f;
164         f=fopen("project","rb"); //OPENS A FILE PROJECT IN READING MODE
165
166         if(f==NULL)
167         {
168             printf("\nERROR!");
169             exit(1);   //EXITS IF FILE DOESNT OPEN
170         }
```

```
171
172         int recordno=0;
173
174         printf("\nLISTING CONTACTS");
175
176         while(fread(&p,sizeof(p),1,f)) //READS A RECORD FROM FILE
177         {
178             system("cls");   //CLEARS SCREEN
179
180             recordno++;
181             printf("\n\n\nRECORD NO:\t%d\n\n ",recordno);   //DISPLAYS INFO OF EACH RECORD ONE BY ONE
182             output(p);
183
184             getch();
185         }
186
187         if(recordno==0)
188          printf("\nEMPTY!");
189
190         fclose(f);   //CLOSES FILE PROJECT
191         printf("\n Enter any key");
192         getch();
193         system("cls");   //CLEARS SCREEN
194         menu();   //OPENS MENU
195
196         return 0;
197     }
198
199     int searchrecord()
200     {
201         struct person p;
202         FILE *f;
203         char name[100];
204
```

```c
    f=fopen("project","rb");  //OPENS A BINARY FILE PPROJECT IN READING MODE
    if(f==NULL)
    {
        printf("\nERROR!\a\a\a\a");   //DISPLAYS ERROR IF FILE DOESNT OPEN
        exit(1);

    }

    printf("\nSEARCH FOR A RECORD");

    fflush(stdin);
    printf("\nEnter name of person to search\n");
    gets(name);

    int flag=0;
    while(fread(&p,sizeof(p),1,f))
    {
       if(strcmp(p.name,name)==0)  //IF FOUND A MATCH THEN DISPLAYS THE INFO OF THE SEARCHED RECORD
       {
        output(p);
        flag=1;
        break;
       }
    }

    if(flag!=1)  //DISPLAYS A MESSAGE IF SEARCHED RECORD NOT FOUND
        printf("\nNOT FOUND!");

    fclose(f);  //CLOSES FILE PROJECT

    fflush(stdin);
    printf("\n Enter any key");
    getch();

```

```c
    system("cls");  //CLEARS SCREEN
    menu();

    return 0;
}
int deleterecord()
{
    struct person p;
    FILE *f,*ft;
    int flag;
    char name[100];

    f=fopen("project","rb");  //OPENS FILE PROJECT IN READING MODE

    if(f==NULL)  //DISPLAYS AN ERROR MESSAGE IF PROJECT FILE DOESNT OPEN
        {
            printf("ERROR!");
        }

    else
        {
            ft=fopen("temp","wb");  //OPENS A TEMPORARY FILE IN WRITING MODE
            if(ft==NULL)  //DISPLAYS ERROR IF TEMP FILE DOESNT OPEN
               printf("ERROR");
            else
            {
                printf("\nDELETION");

                fflush(stdin);
                printf("Enter CONTACT'S NAME:");
                gets(name);  //INPUTS NAME OF CONTACT TO BE DELETED
              //  fflush(stdin);  //CLEARS BUFFER

                while(fread(&p,sizeof(p),1,f))  //READS RECORDS
```

```c
                {
                    if(strcmp(p.name,name)!=0)  //IF THE READ RECORD IS NOT THE ONE TO BE DELETED
                        fwrite(&p,sizeof(p),1,ft);  //IT IS WRITTEN IN TEMP FILE
                    if(strcmp(p.name,name)==0)  //IF THE READ RECORD IS THE ONE TO BE DELETED
                        {
                            flag=1;  //IT IS NOT WRITTEN IN TEMP FILE
                            printf("\nRecord to be deleted found...");
                            output(p);
                        }
                }

                fclose(f);  //CLOSES PROJECT FILE
                fclose(ft);  //CLOSES TEMP FILE

                if(flag!=1)  //IF RECORD TO BE DELETED NOT FOUND
                {
                    printf("\nNOT FOUND!");
                    remove("temp.txt");  //REMOVES TEMP FILE
                }
                else
                {
                    remove("project");  //REMOVES PROJECT FILE
                    rename("temp","project");  //RENAMES TEMP FILE TO PROJECT FILE
                    printf("RECORD DELETED...");
                }
            }
        }

        fflush(stdin);
        printf("\n Enter any key");
        getch();

        system("cls");  //CLAERS SCREEN
        menu();  //OPENS MENU
```

```c
    return 0;
}

int modifyrecord()
{
    FILE *f;

    f=fopen("project","rb+");  //OPEN A BINARY FILE PROJECT IN READING AND WRITING MODE

    if(f==NULL)  //DISPLAYS ERROR IF FILE DOESNT OPEN
        {
            printf("ERROR!");
            exit(1);  //EXITS PHONEBOOK
        }
    else
       {
           printf("\nMODIFICATION");

            system("cls");  //CLEARS SCREEN

            int flag=0;
            struct person p;
            char  name[50];

            fflush(stdin);
            printf("\nEnter CONTACT'S NAME TO MODIFY:\n");
            gets(name);

            while(fread(&p,sizeof(p),1,f))  //READS RECORDS ONE BY ONE TILL THE END OF FILE UNLESS THE CONTACT TO BE MODIFIES I
            {
                if(strcmp(name,p.name)==0)  //IF RECORD TO BE MODIFIED IS FOUND
                {
                    printf("\nContact to be modified found...");
```

```c
                    input(&p);  //INPUT THE NEW CONTACT INFO
                    fseek(f,-sizeof(p),SEEK_CUR);  //POINTER TAKEN TO THE BEGINNING OF THE RECORD TO BE MODIFIED
                    fwrite(&p,sizeof(p),1,f);  //MODIFIED RECORD WRITTEN
                    flag=1;
                    break;
                }
                fflush(stdin);  //CLEARS BUFFER
            }
            if(flag==1)
            {
                printf("\nMODIFIED!");

            }
            else  //DISPLAYS A MESSAGE IF RECORD TO BE MODIFIED NOT FOUND
            {
                    printf(" \nNOT FOUND");

            }
        }

        fclose(f);  //CLOSES PROJECT FILE
        printf("\n Enter any key");
        getch();
        system("cls");  //CLEARS SCREEN
        menu();  //OPENS MENU

        return 0;

}
int input(struct person *p)
{
        fflush(stdin);
        printf("\n\nEnter name:\t");  //INPUT NAME
        gets(p->name);

        //VALID PHONE NUMBER INPUT
          do
          {
                fflush(stdin);
                printf("\nEnter phone no.:\t");  //INPUT PHONE NO
                gets(p->mble_no);

                int mblelen=strlen(p->mble_no);

                if(mblelen!=10)  //checking number of digits
                {
                   printf("\nInvalid phone number!");
                   continue;  //asks for phone no again if input is not a 10 digit
                }

                int k=0,flag=0;

                while(k<mblelen)
                {
                      if(p->mble_no[k]<48 || p->mble_no[k]>57)  //if not a digit [ '0' =48 and '9' =57 ]
                      {
                           printf("\nInvalid phone number!");
                           flag=1;  //asks for phone no again if input is not a 10 digit number
                           break;
                      }
                      k++;
                }
              if(flag==1)
                   continue;
              else
                  break;
          }while(1);

```

```c
        fflush(stdin);
        printf("\nEnter e-mail:\t");   //INPUT EMAIL ID
        gets(p->mail);

        fflush(stdin);
        printf("\nEnter Date Of Birth(dd/mm/yyyy):\t");   //INPUT DATE OF BIRTH
        gets(p->DOB);

        fflush(stdin);
        printf("\nEnter sex:\t");   //INPUT GENDER
        gets(p->sex);

        fflush(stdin);
        printf("\nEnter mother name:\t");   //INPUT MOTHERS NAME
        gets(p->mother_name);

        fflush(stdin);
        printf("\nEnter father name:\t");   //INPUT FATHERS NAME
        gets(p->father_name);

        fflush(stdin);
        printf("\nEnter nationality:\t");   //INPUT NATIONALITY
        gets(p->nationality);

        fflush(stdin);
        printf("\nEnter the address:\t");   //INPUT ADDRESS
        gets(p->address);

        return 0;
}

int output(struct person p)
{
        printf("\nName=\t");
```

```c
        puts(p.name);
        printf("\nPhone No.=\t");
        puts(p.mble_no);
        printf("\nEmail id=\t");
        puts(p.mail);
        printf("\nDOB=\t");
        puts(p.DOB);
        printf("\nGender=\t");
        puts(p.sex);
        printf("\nMother's Name=\t");
        puts(p.mother_name);
        printf("\nFather's Name=\t");
        puts(p.father_name);
        printf("\nNationality=\t");
        puts(p.nationality);
        printf("\nAddress=\t");
        puts(p.address);

        return 0;
}

int passcheck()
{
    system("cls");

    char pass[]="DTU2024",password[10];

    COORD P;
    P.X=40;
    P.Y=5;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),P);

    printf("PASSWORD:\t");
    int p=0;
```

```c
    do{
        password[p]=getch();
        if(password[p]!='\r'){
            printf("*");
        }
        p++;
    }while(password[p-1]!='\r');
    password[p-1]='\0';

    if(strcmp(password,pass)==0)
        return 0;
    else
        {
            printf("\a");   //Beeps when input password is wrong
            passcheck();
        }
}
```

OUTPUT

```
PASSWORD:        *******
```

```
                    WELCOME TO PHONEBOOK

                              MENU
1.Add New      2.List      3.Modify      4.Search      5.Delete      6.Exit
```

```
ADDING A CONTACT

Enter name:     Amitabh Bachchan

Enter phone no.:        9988776655

Enter e-mail:   bachchan@gmail.com

Enter Date Of Birth(dd/mm/yyyy):        11/11/1996

Enter sex:      Male

Enter mother name:      Abc Bachchan

Enter father name:      Def Bachchan

Enter nationality:      Indian

Enter the address:      Bachchan Mansion, Mumbai, India

SAVED...

Enter any key
```

```
Enter CONTACT'S NAME TO MODIFY:
Amitabh Bachchan

Contact to be modified found...

Enter name:     Amitabh Bachchan

Enter phone no.:        9988776655

Enter e-mail:   bachchan@gmail.com

Enter Date Of Birth(dd/mm/yyyy):        11/11/1996

Enter sex:      Male

Enter mother name:      Abc Bachchan

Enter father name:      Def Bachchan

Enter nationality:      Indian

Enter the address:      123 Street, Washington, U.S.A.

MODIFIED!
 Enter any key
```

```
LISTING CONTACTS

RECORD NO:      1

Name=   Amitabh Bachchan

Phone No.=      9988776655

Email id=       bachchan@gmail.com

DOB=    11/11/1996

Gender= Male

Mother's Name=  Abc Bachchan

Father's Name=  Def Bachchan

Nationality=    Indian

Address=        123 Street, Washington, U.S.A.
```

```
SEARCH FOR A RECORD
Enter name of person to search
Amitabh Bachchan

Name=   Amitabh Bachchan

Phone No.=      9988776655

Email id=       bachchan@gmail.com

DOB=    11/11/1996

Gender= Male

Mother's Name=  Abc Bachchan

Father's Name=  Def Bachchan

Nationality=    Indian

Address=        123 Street, Washington, U.S.A.

 Enter any key
```

```
DELETIONEnter CONTACT'S NAME:Amitabh Bachchan

Record to be deleted found...
Name=   Amitabh Bachchan

Phone No.=      9988776655

Email id=       bachchan@gmail.com

DOB=    11/11/1996

Gender= Male

Mother's Name=  Abc Bachchan

Father's Name=  Def Bachchan

Nationality=    Indian

Address=        123 Street, Washington, U.S.A.
RECORD DELETED...
 Enter any key
```

ADVANTAGES

- It becomes easy for the user to store complete information (i.e. Email id, Address, etc.) about his contact.
- It is easy for the user to just search his required contact number by just typing the name of the contact.
- If there are two or more contacts with the same name, when searched the program shows both the records.

DISADVANTAGES

- Sometimes it becomes difficult to store more contacts (i.e. over 150).
- To search contact by phone no.

CONCLUSION

This program makes the user simple to connect to his contact. The contact personal information and family information is stored under a single number this would benefit the user to easily search and locate his required contact. This program deals with five operations of adding contact, listing contact, modifying a contact, searching according to the user's choice and deleting them. Each operation made as an individual function and so control enters a different structure and all the data added, modified or deleted going to be stored in a .txt file using file pointer. With the help of this project we are able to understand the concept of file handling, data structure, control instruction etc. in a more efficient manner. This application can be used in many organisations to store staff and employees personal details.