VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

A Data Structure Mini Project Report on

TELEPONE DIRECTORY

Submitted in partial fulfillment of the requirements as a part of the Data structure Lab for the III Semester of degree of ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi*

Submitted by

SAURAV KUMAR
1RN19IS140

SEEMA SADIYA R
1RN19IS141

SHEIKH SAIFULLA
1RN19IS142

SHARANYA RP
1RN20IS143

SHARATH KUMAR BS
1RN20IS144

Under the Guidance of
Faculty Incharge
Mrs. SUNITHA K
Assistant Professor
Dept. of ISE, RNSIT

Department of Information Science and Engineering
RNS Institute of Technology
Channasandra, Dr. Vishnuvardhan Road, RR Nagar
Post,Bengaluru – 560 098

2021 – 2022

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of this mini-project would be incomplete without the mention of the people who made it possible through constant guidance and encouragement.

We would take this opportunity to express our heart-felt gratitude to **Dr. M.K. Venkatesha**, Principal, RNS Institute of Technology for providing the necessary infrastructure to complete this mini-project.

We wish to express our deepest gratitude and thanks to **Dr. Suresh L**, Head of the Department, Information Science and Engineering.

We wish to express sincere thanks to my guide **Sunitha K**, Assistant Professor, Department of Information Science and Engineering for helping us throughout and guiding us from time to time.

A warm thanks to all the faculty of Department of Information Science and Engineering, who have helped us with their views and encouraging ideas.

ABSTRACT

The telephone directory is gaining more importance as the number of its users is increasing rapidly. As the number is rising there is a need of effective management of contacts. The purpose of telephone directory is that it provides facilities to member to search for the required contacts and it will simplify the task and save more paperwork. This will help the user to easily manage the contacts which contains the details of the individual along with their phone numbers .Anyone can add contacts and their details and it allows update information, of the existing record, and delete a particular contact too. Functions of this system can be easily accessed by administrators, students and applicants.

TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION

What is Telephone Directory?

A telephone directory, often known as a phone book, telephone address book, and phone book is an account-based system for viewing, adding, modifying, and deleting phone records based on names and phone numbers.

What is the telephone directory used for?

Its objective is to locate a subscriber's phone number based on his or her name and address.

Why Telephone Directory is important?

- When looking for phone numbers or postal addresses of individuals or organizations, a telephone directory is a useful resource.
- It is a highly powerful and straightforward beginner's understanding.
- telephone directory is a project is to help us in a technical assignment o store contacts using simple c code
- It helps user to manage contacts based on the names using this system
- If no such record is available, then a proper error message will be displayed as per the user input provided to the system.

CHAPTER 2. DATA STRUCTURES

Which data structures are used?

- **Files** and **structures** are the two main data structures which are used in this project.
- They are easy to code and understand.

Why are we using these data structures?

->Structures

•Structure is a user defined data type in C language which allows us to combine data of different types together.

•Structure helps to construct a complex data type which is more meaningful.

•Structure can store data of any type, which is practical and more useful.

•We use structure to group several related things in one place.

•The structure puts the information in one place, with easy access. Which can pass around a single user to functions.

->Files

•A file represents a sequence of bytes on the disk where a group of related data is stored.

•File is created for permanent storage of data in the external memory.

•File is a ready-made structure.

•We use a structure pointer of file type to declare a file.

```
FILE *fp;
```

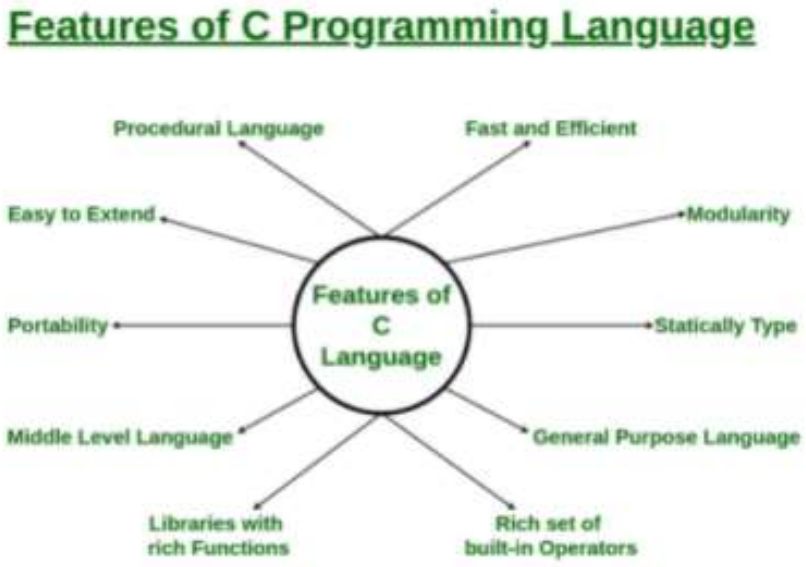
•It provides a number of functions that helps to perform basic file operations.

->Following are the functions:

Function	description
fopen()	create a new file or open a existing file
fclose()	closes a file
getc()	reads a character from a file
putc()	writes a character to a file
fscanf()	reads a set of data from a file
fprintf()	writes a set of data to a file
getw()	reads a integer from a file
putw()	writes a integer to a file
fseek()	set the position to desire point
ftell()	gives current position in the file
rewind()	set the position to the begining point

CHAPTER 3. SALIENT FEATURES OF C PROGRAMMING

- Procedural Language
- Fast and Efficient
- Modularity
- Statically Type
- General-Purpose Language
- Rich set of built-in Operators
- Libraries with rich Functions
- Middle-Level Language
- Portability
- Easy to Extend



Modularity:

It is the concept of storing C programming language code in the form of libraries for further future uses.

Rich set of built-in Operators:

It is a diversified language with a rich set of built-in operators which are used in writing complex or simplified C programs.

Libraries with rich Functions:

Robust libraries and functions in C help even a beginner coder to code with ease.

Middle-Level Language:

As it is a middle-level language so it has the combined form of both capabilities of assembly language and features of the high level language.

CHAPTER 4. DESIGN

ALGORITHM

Step 1: Start.

Step 2: Display operations on the phonebook.

Step 3: Get the user input.

Step 4: Generate the output based on the selected operation.

a) For Adding a record, get the data in particular order and save the record.

b) For Displaying a record, if the file is empty display an error message else print the contents.

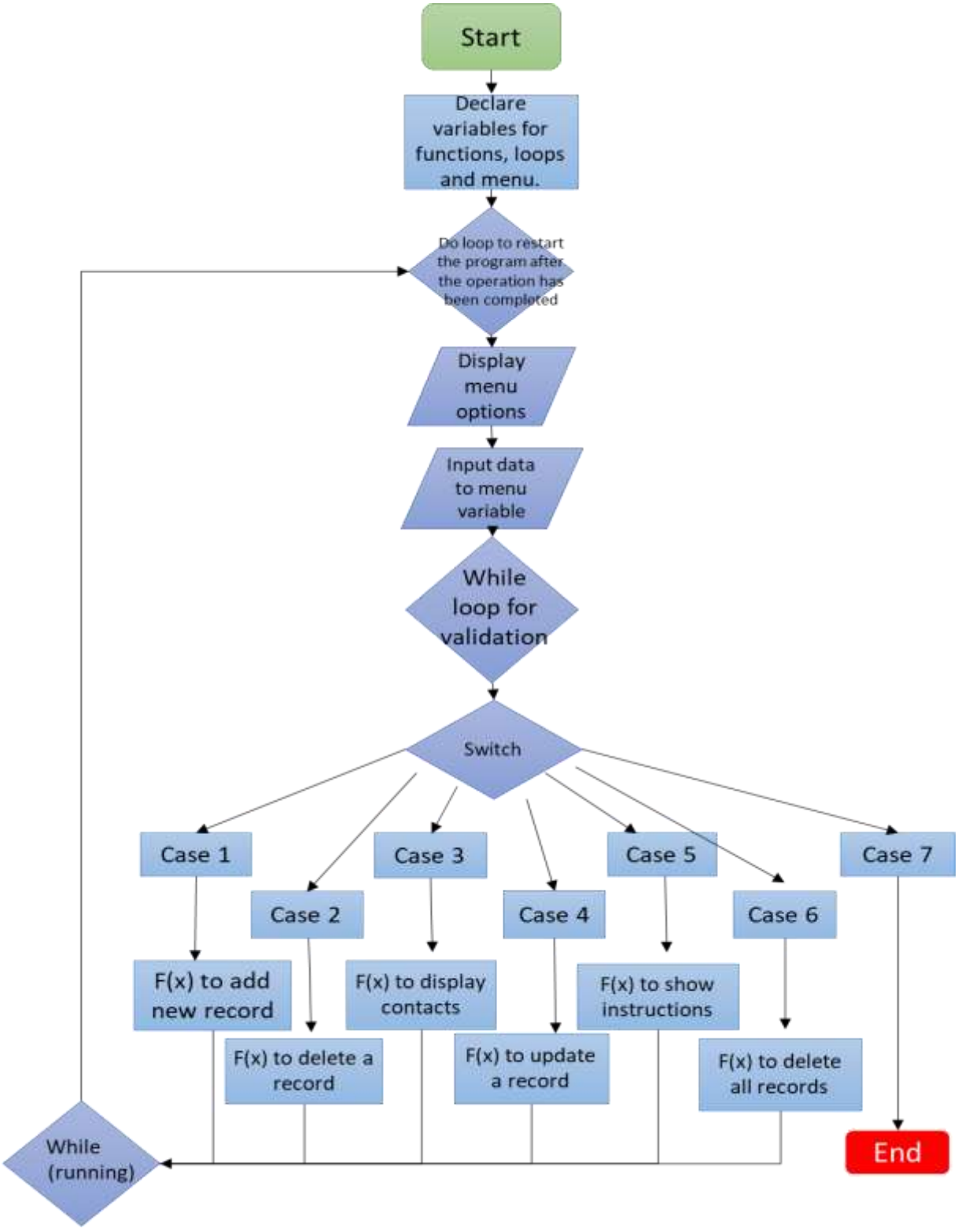
c) For Deleting a record, if the file is empty display an error message, else if file doesn't exist print suitable message, else delete the record and close the respective file.

d) For Updating a record, if record doesn't exist in the file print an error message, else search the record in the file and update the details.

e) For Deleting all records, remove/close the file.

Step 5: After the exit option is selected, display suitable message and terminate the program.

FLOWCHART


Start
Declare variables for functions, loops and menu.
Do loop to restart the program after the operation has been completed
Display menu options
Input data to menu variable
While loop for validation
Switch
Case 1
Case 2
Case 3
Case 4
Case 5
Case 6
Case 7
F(x) to add new record
F(x) to delete a record
F(x) to display contacts
F(x) to update a record
F(x) to show instructions
F(x) to delete all records
While (running)
End

CHAPTER 5. SYSTEM REQUIREMENTS

Hardware Requirements

- **Processor:** Intel Core2 Quad @ 2.4Ghz on Windows® Vista 64-Bit / Windows® 7 64-Bit / Windows® 8 64-Bit / Windows® 8.1 64-Bit.
- **RAM:** 2GB of RAM
- **Memory:** 256GB Hard drive
- **Keyboard:** MS compatible keyboard
- **Mouse:** MS compatible mouse

Software Requirements

- **Operating system:** Windows® Vista 64-Bit / Windows® 7 64-Bit / Windows® 8 64-Bit / Windows® 8.1 64-Bit.
- **Front end Programming language :** C
- **IDE:** VISUAL STUDIO CODE or CODEBLOCKS

CHAPTER 6. IMPLEMENTATION

MAIN FUNCTIONS:

1. void add();

CODE:

```
void add() {
 struct contact c;
 int total_contact=0;
 total_contact=total_contacts();
 FILE *fp;
 printf("Enter name : ");
 scanf("%s",c.name);
 printf("Enter number : ");
 scanf("%s",c.number);
 while (!(atoi(c.number) / 2)) //The atoi function returns the integer representation of
a string
 {
    error_message("Please enter a valid number");
    printf("Enter number : ");
    scanf("%s",c.number);
 }
 fp=fopen("tel.txt","a");
 if ( fp == NULL )
 {
    printf( "Error" );
 } else {
    fprintf(fp, "%d. %s %s \n", total_contact+1,c.name, c.number);
    printf("----------------------------------------\n");
    printf("* Successfully created contact. *\n");
    printf("* Name : %s | Number : %s *\n",c.name,c.number);
    printf("----------------------------------------\n");
 }
 fclose(fp);
}
```

2. void delete();

CODE:
```
void delete() {
char details[50],det[50];
FILE *fp,*temp;
int count=0,id_to_delete,found=0,temp_count=1;
show();
printf("Enter id to delete : ");
scanf("%d",&id_to_delete);
if (id_to_delete<=0 || id_to_delete>=(total_contacts()+1)) {
   error_message("Please enter a valid id");

   return;
}
fp=fopen("tel.txt","r");
temp=fopen("tel_temp.txt","w");
if ( fp == NULL )
{       printf( "Error" ) ;    }
else {
   while (fgets(details, 50, fp) != NULL) {
      count++;
      if(count!=id_to_delete) {
         for (int i=0;i<strlen(details);i++) {
            if (details[i]=='.') {
               found=i+2;
               for (int j=0;j<50;j++) {
                  det[j]=details[found];
                  found++;  }
               found=0;  }
       }
            fprintf(temp, "%d. %s", temp_count,det);
            for (int j=0;j<50;j++) {
               det[j]=' ';
               found++;
            }
            temp_count++;
       }
   }
   fclose(fp);
   fclose(temp);
   printf("---------------------------------\n");
   printf("* Successfully Deleted *\n");
   printf("---------------------------------\n");
   remove("tel.txt");
   rename("tel_temp.txt", "tel.txt");
}
}
```

3. void update();

```
CODE:  void update() {
           FILE *fp, *temp;
           int count=0,id;
           char details[50],name[50],number[11];
           struct contact c;
           show();
           printf("Enter id to update : ");
           scanf("%d", &id);
           if (id<=0 || id>=(total_contacts()+1)) {
              error_message("Please enter a valid id");
              update();
              return;
           }
           printf("Enter new name : ");
           scanf("%s",c.name);
           printf("Enter new number : ");
           scanf("%s",c.number);
           while (!(atoi(c.number) / 2)) {
              error_message("Please enter a valid Number");
              printf("Enter new Number : ");
              scanf("%s",c.number);
           }
           fp = fopen("tel.txt", "r");
           temp = fopen("tel_temp.txt", "w");
           if (fp == NULL) {
              printf("Error");
           } else {
              while (fgets(details, 50, fp) != NULL) {
                 count++;
                 if (count == id) {
                    // Writing updated details to the file
                    fprintf(temp, "%d. %s %s \n", count,c.name, c.number);
                    printf("----------------------------------------------\n");
                    printf("* Successfully Updated *\n");
                    printf("* Name : %s | Number : %s *\n",c.name,c.number);
                    printf("----------------------------------------------\n");
                 } else {
                    fprintf(temp, "%s",details);      }
              }
              fclose(fp);
              fclose(temp);
              remove("tel.txt");
              rename("tel_temp.txt", "tel.txt");
         }
       }
```

4. void show();

```
CODE:  void show() {
       FILE *fp;
       char details[50];
       fp=fopen("tel.txt","r");
       if(total_contacts()==0) {
          printf("-----------------------------\n");
          printf("* No Contacts Added *\n");
          printf("-----------------------------\n");
          return;
       }
       else if ( fp == NULL )
       {   printf( "Error" ) ;    }
       else {
          printf("-------------------------\n");
          printf("ID Name\tNumber\n");
          while (fgets(details, 50, fp) != NULL) {
             printf("%s", details);
          }
          printf("-------------------------\n");
          fclose(fp) ;
         }
      }
```

5. int delete_all();

```
CODE:  int delete_all()
       {
          FILE *fp,*temp;
          remove("FILE tel.txt");
          rename("tel_temp.txt", "tel.txt");
             if (remove("tel.txt") == 0) {
             printf("All the contacts have been erased\n");
          } else {
             error_message("Error");    }
          return 0;
       }
```

SUPPORTIVE FUNCIONS

- void error_message(char msg[]);
- int total_contacts();
- void showInstructions();
- void delay(int milliseconds);

CHAPTER 7. RESULT

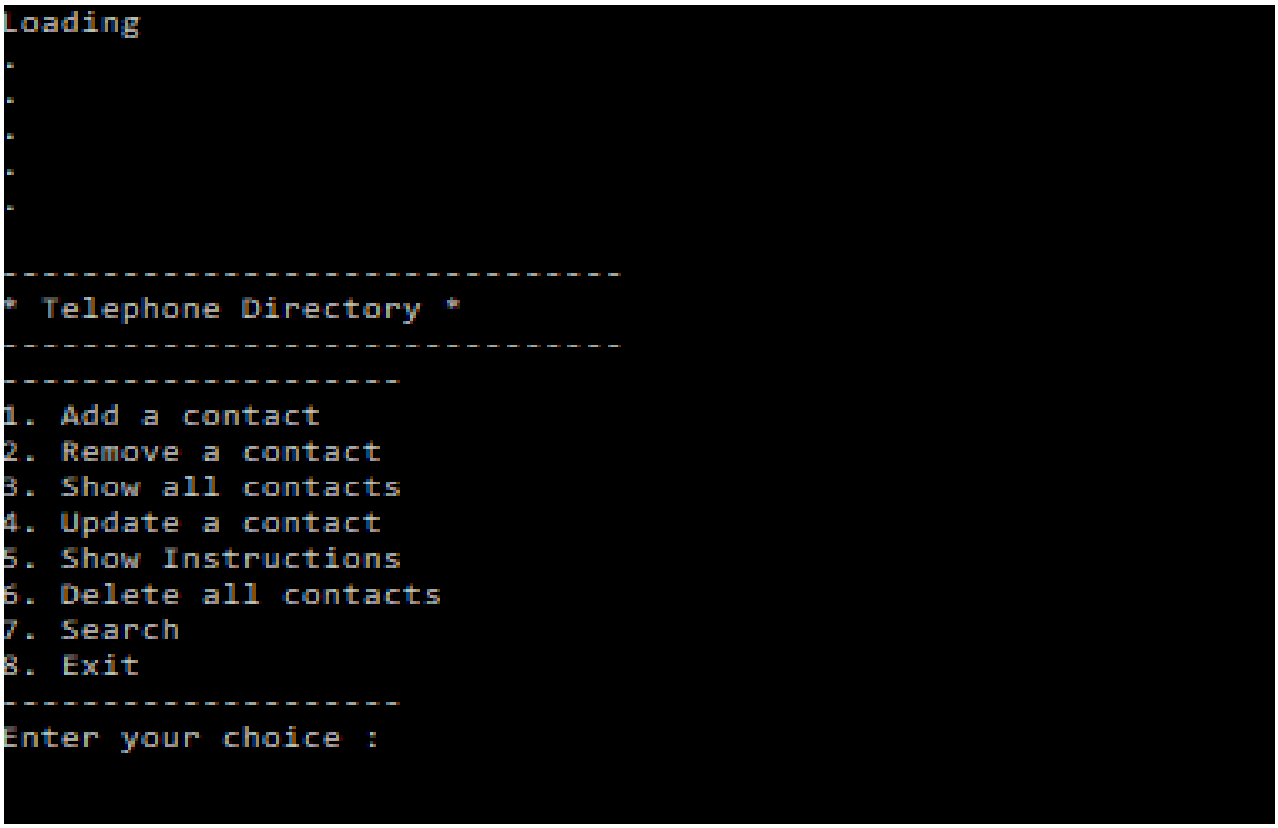
SNAPSHOTS OF OUTPUT SCREENS

Beginning screen

The following snapshot contains the welcome screen of the directory.

This is the snapshot of the home screen of the telephone directory.

It has all the functions displayed.



Adding contacts into directory

The following snapshot contains the use of the function add.

Here, the contacts or each record can be added into the directory.

It takes two inputs of name and number.

```
7. Search
8. Exit
--------------------
Enter your choice : 1
Enter name : ADITI
Enter number : 8765768795
-----------------------------------------
* Successfully created contact. *
* Name : ADITI | Number : 8765768795 *
-----------------------------------------
Enter your choice : 1
Enter name : GOPI
Enter number : 9936527468
-----------------------------------------
* Successfully created contact. *
* Name : GOPI | Number : 9936527468 *
-----------------------------------------
Enter your choice : 1
Enter name : RAM
Enter number : 9972046178
-----------------------------------------
* Successfully created contact. *
* Name : RAM | Number : 9972046178 *
-----------------------------------------
```

Deleting a contact and Display

The following snapshot contains the use of the function delete and display.

Here, a single contact or record is deleted with the help of the ID which the user enters.

Then all the contacts in the directory are being displayed after deleting a contact.

```
Enter your choice : 3
--------------------------
ID Name Number
1. ADITI 8765768795
2. GOPI 9936527468
3. RAM 9972046178
--------------------------
Enter your choice : 2
--------------------------
ID Name Number
1. ADITI 8765768795
2. GOPI 9936527468
3. RAM 9972046178
--------------------------
Enter id to delete : 2
--------------------------------
* Successfully Deleted *
--------------------------------
Enter your choice : 3
--------------------------
ID Name Number
1. ADITI 8765768795
2. RAM 9972046178
--------------------------
Enter your choice :
```

Updating the existing contacts

The following snapshot contains the use of the function update.

Here, the details of a single contact can be updated using the ID entered by the user.

Both name and the number can be updated.

```
--------------------------
Enter your choice : 4
--------------------------
ID Name Number
1. ADITI 8765768795
2. RAM 9972046178
--------------------------
Enter id to update : 2
Enter new name : AMAR
Enter new number : 9879581045
--------------------------------------------------
* Successfully Updated *
* Name : AMAR | Number : 9879581045 *
--------------------------------------------------
Enter your choice : 3
--------------------------
ID Name Number
1. ADITI 8765768795
2. AMAR 9879581045
--------------------------
Enter your choice :
```

Displaying all the instructions

The following snapshot contains the use of the function update.

Here, all the functions or the instruction which can be performed is again displayed.

```
----------------------------
Enter your choice : 5
--------------------
1. Add a contact
2. Remove a contact
3. Show all contacts
4. Update a contact
5. Show Instructions
6. Delete all contacts
7. Search
8. Exit
--------------------
Enter your choice :
```

Searching for a contact

The following snapshot contains the use of the function search.

Here, a contact details can be searched using the name entered by the user.

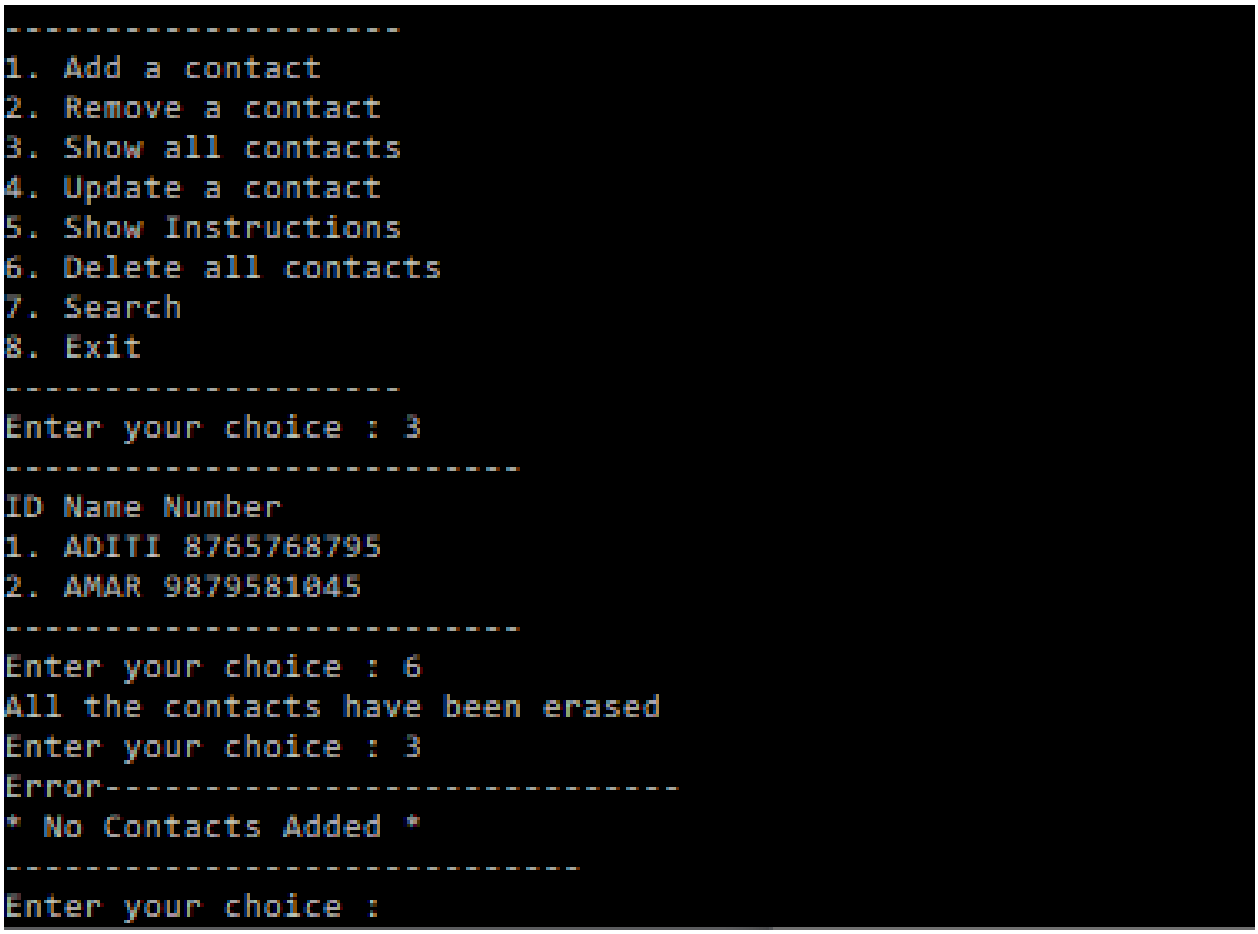
It displays both name and number of the contact.

```
1. Add a contact
2. Remove a contact
3. Show all contacts
4. Update a contact
5. Show Instructions
6. Delete all contacts
7. Search
8. Exit
--------------------
Enter your choice : 7
Enter the name you want to search for
AMAR
Name        Number
AMAR    9879581045
Enter your choice :
```

Deletion of all contacts

The following snapshot contains the use of the function delete_all.

The directory can be made empty by deleting all the contacts at once .



CHAPTER 8. CONCLUSION AND FURTHER ENHANCEMENTS

CONCLUSION

Telephone Directory is only a humble venture to satisfy the needs to manage the work. Several user-friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the telephone directory. It can be used to replace our hard phone book. This will help the user to easily manage the contacts which contains the details of the individual along with their phone numbers. This application can be used by any organization whether big or small that has challenges to overcome and managing the contacts of the directory.

FURTHER ENHANCEMENTS

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding:

- More advance software can be given for the library management system with more facilities
- The platform can be hosted on the online servers to make it accessible worldwide.
- The telephone directory can be further upgraded or enhanced by adding other functions to perform handy operations like storing and maintaining different phonebook using cloud storage.

The above mentioned points are the enhancements which can be done to increase the applicability and usage of this project.