

**PSP [20ES104] COURSE PROJECT REPORT**

**On**

**“Phone Book Management”**

Developed By:

H.T.NO STUDENT NAME

2203A51365 K.SAI HARSHITH

2203A51333 A.RISHI CHAND

2203A51345 E.PRASANNA KUMAR

2203A51394 U.SIDDHARTHA

Under the Guidance of

Mr. Riyaz Mohammed

Assistant Professor

Submitted to

Department Computer Science and Artificial Intelligence SR University

Ananthasagar(V), Hasanparthy(M), Hanamkonda(Dist.) – 506371

[www.sru.edu.in](http://www.sru.edu.in/)

**June 2023**

**Department of Computer Science and Artificial Intelligence**

**CERTIFICATE**

This is to certify that the PSP course project report entitled **“Phone Book Management”** is a record of bonafide work carried out by the student(s) Sai Harshith, Rishi Chand, Prasanna Kumar, Siddhartha bearing roll number(s) 2203A51365, 2203A51333, 2203A51345, 2203A51394 of Computer Science and Artificial Intelligence department during the academic year 2022-23.

**Supervisor**

(Riyaz Mohammed)

**INDEX**

**Sl. No Title Page No.**

1. Problem statement 1
2. Module-wise description 2
3. Knowledge required to develop the project 3
4. Source code (.c file code followed by .h file code) 4
5. Results 8

# PROBLEM STATEMENT:

The phone book is a very simple C project that will help you understand the core concepts of capacity, record keeping, and data structure. This program will have how to add, list, edit or alter and delete data from a record.

The functionality as below mentioned:

1. Read ‘n’ Members details dynamically such as:

* Name
* Phone Number

1. Sort ‘n’ Members details according to user’s choice:

* Add a contact
* Read the contacts
* Edit the contact
* Delete the contact
* Exit system

1. We can view all the contacts in a systematic manner.
2. Print the ‘n’ members details.

# MODULES:

In this application all variables and structure are declared globally so that these variables and structure members can be accessed throughout the program at any function call. We can choose any function by using function calls which are declared in switch-case. The memory allocation will be done in this program dynamically.

In this application four modules are used.

1. Read/Input

In this module the application asks the person who runs the program to enter n members details. To give n members details for loop is used.

1. Sorting

In this module sorting of data is done according to the chosen wise. In this module there is a sub menu which asks to select the sorting wise by using switch case. The sorting sub menu will be like press 1 to sort by add a contact press 2 to sort by list or read all the contacts press 3 to sort by edit a contact press 4 to sort by delete a contact press 5 to exit the system.

In this module we used another control statement (do while) so that the application asks whether to continue sorting.

1. Searching

In this module searching of data is done according to the chosen wise.

In this module there is a sub menu which asks to select the sorting wise by using switch case. The sorting sub menu will be like press 1 to search by add a contact press 2 to search by list or read all the contacts press 3 to search by edit a contact press 4 to search by delete a contact press 5 to exit the system.

1. Print

In this module all the details of ‘n’ members will be displayed on to the screen systematically. In this module printf function and for loop are used.

**KNOWLEDGE REQUIRED TO DEVELOP THIS APPLICATION**

* + Control Statements (if, if-else, switch)
  + Loop Statements (while/do while, for)
  + Arrays (1D/2D-arrays)
  + Strings (Strings and Table of strings) and its functions (strcpy, strcmp)
  + Functions (Any type of user defined functions)
  + Structure (structures and nested structures)
  + Pointers (pointer to strings and pointers to structures)
  + Dynamic Memory Allocation (malloc/ calloc/ realloc)

**SOURCE CODE [.C FILE]:**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

struct Contact {

char name[50];

char phone[20];

};

int main() {

int choice, i;

long count = 0;

char name[50];

struct Contact contact;

FILE \*fp;

fp = fopen("phonebook.dat", "r+");

if (fp == NULL) {

fp = fopen("phonebook.dat", "a+");

if (fp == NULL) {

printf("Cannot open file");

exit(1);

}

}

// Read existing contacts and update count

fseek(fp, 0, SEEK\_END);

count = ftell(fp) / sizeof(struct Contact);

rewind(fp);

while (1) {

printf("\nPhone Book\n");

printf("1. Add a contact\n");

printf("2. List all contacts\n");

printf("3. Edit a contact\n");

printf("4. Delete a contact\n");

printf("5. Exit\n");

printf("Enter your choice: ");

scanf("%d", &choice);

switch (choice) {

case 1:

printf("\nEnter name: ");

scanf("%s", contact.name);

printf("Enter phone number: ");

scanf("%s", contact.phone);

fseek(fp, 0, SEEK\_END);

fwrite(&contact, sizeof(struct Contact), 1, fp);

count++;

break;

case 2:

printf("\nList of contacts\n");

rewind(fp);

for (i = 0; i < count; i++) {

fread(&contact, sizeof(struct Contact), 1, fp);

printf("%s\t%s\n", contact.name, contact.phone);

}

break;

case 3:

printf("\nEnter the name of the contact to edit: ");

scanf("%s", name);

rewind(fp);

for (i = 0; i < count; i++) {

fread(&contact, sizeof(struct Contact), 1, fp);

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

{

printf("Enter new phone number: ");

scanf("%s", contact.phone);

fseek(fp,-sizeof(struct Contact), SEEK\_CUR);

fwrite(&contact, sizeof(struct Contact), 1, fp);

break;

}

}

break;

case 4:

printf("\nEnter the name of the contact to delete: ");

scanf("%s", name);

rewind(fp);

FILE \*temp = fopen("temp.dat", "w");

if (temp == NULL)

{

printf("Cannot open temporary file");

exit(1);

}

int deleted = 0;

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

{

fread(&contact, sizeof(struct Contact), 1, fp);

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

{

printf("%s\t%s\n", contact.name, contact.phone);

printf("Are you sure you want to delete this contact? (Y/N): ");

scanf(" %c", &choice);

if (choice == 'Y' || choice == 'y')

{

deleted = 1;

continue;

}

}

fwrite(&contact, sizeof(struct Contact), 1, temp);

}

fclose(fp);

fclose(temp);

if (deleted)

{

remove("phonebook.dat");

rename("temp.dat", "phonebook.dat");

printf("Contact deleted successfully.");

count--;

}

else

{

remove("temp.dat");

printf("Contact not found.");

}

fp = fopen("phonebook.dat", "r+");

if (fp == NULL) {

printf("Cannot open file");

exit(1);

}

break;

case 5:

fclose(fp);

exit(0);

default:

printf("Invalid choice\n");

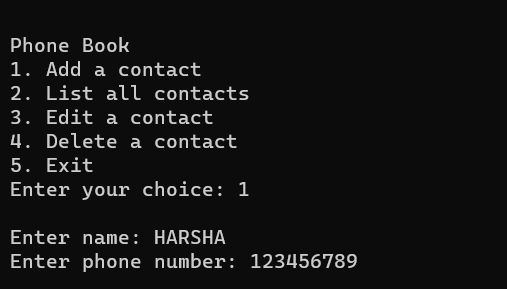
}

}

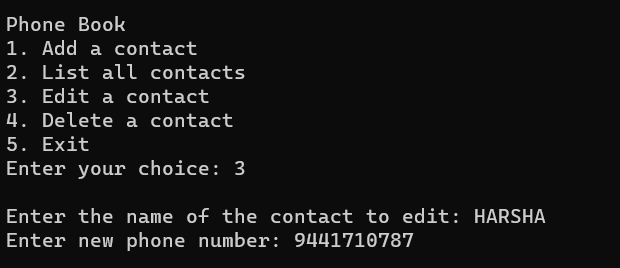
return 0;

**RESULTS:**

After entering choice as 1:

****

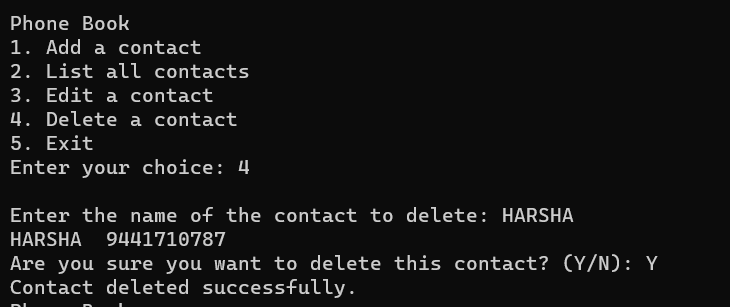
After entering choice as 3:



After entering choice as 2:



After entering choice as 4:



After entering choice as 5 it exits:

