Data Structures

Lab Project (Semester-2)

WORKING PHONEBOOK

Members:

DARSHAN JAIN (B120022)

DEEPAK YADAV (B120023)

International Institute of Information Technology

Bhubaneswar, India

Contents

Title of Project

WORKING PHONEBOOK

Description of Project

The aim of the project was to create the working phonebook which can perform various operations. Initially the phonebook has hundred contacts stored in it. The stored contacts are in separate binary file and that file only is used for any further operations like adding new contact or deleting some contact. Overall, after any operation is performed the binary file is modified.

Operations which are there in phonebook are

1. Add the new contact

2.Update the existing contact

3.Display all the contacts

4.Display the contacts sorted name wise

5.Display the contacts sorted phone number wise

6.Search contact by name

7.Search contact by phone number

8.Delete existing contact

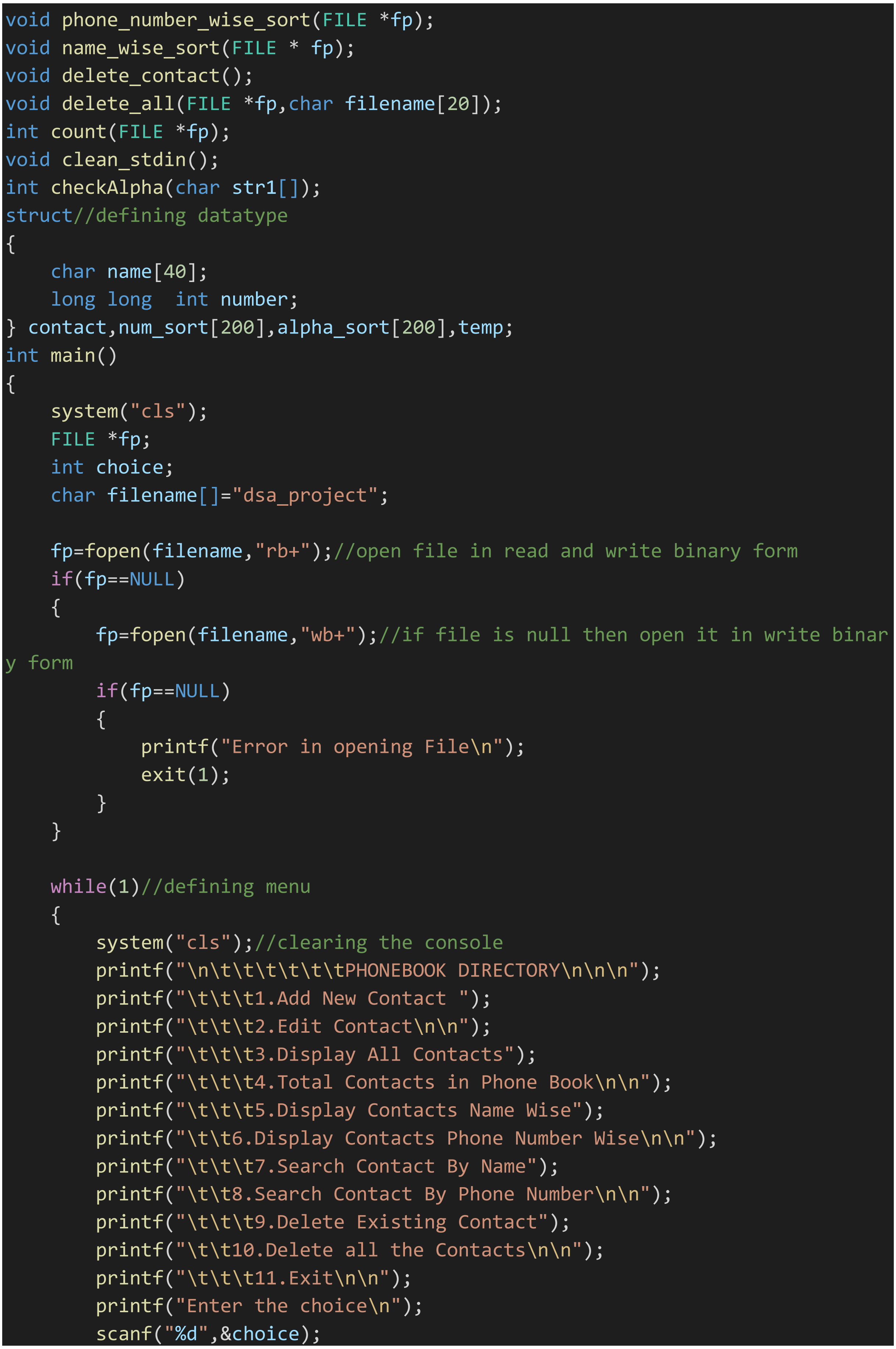
9.Delete all the contacts

10.Total number of contacts in phonebook

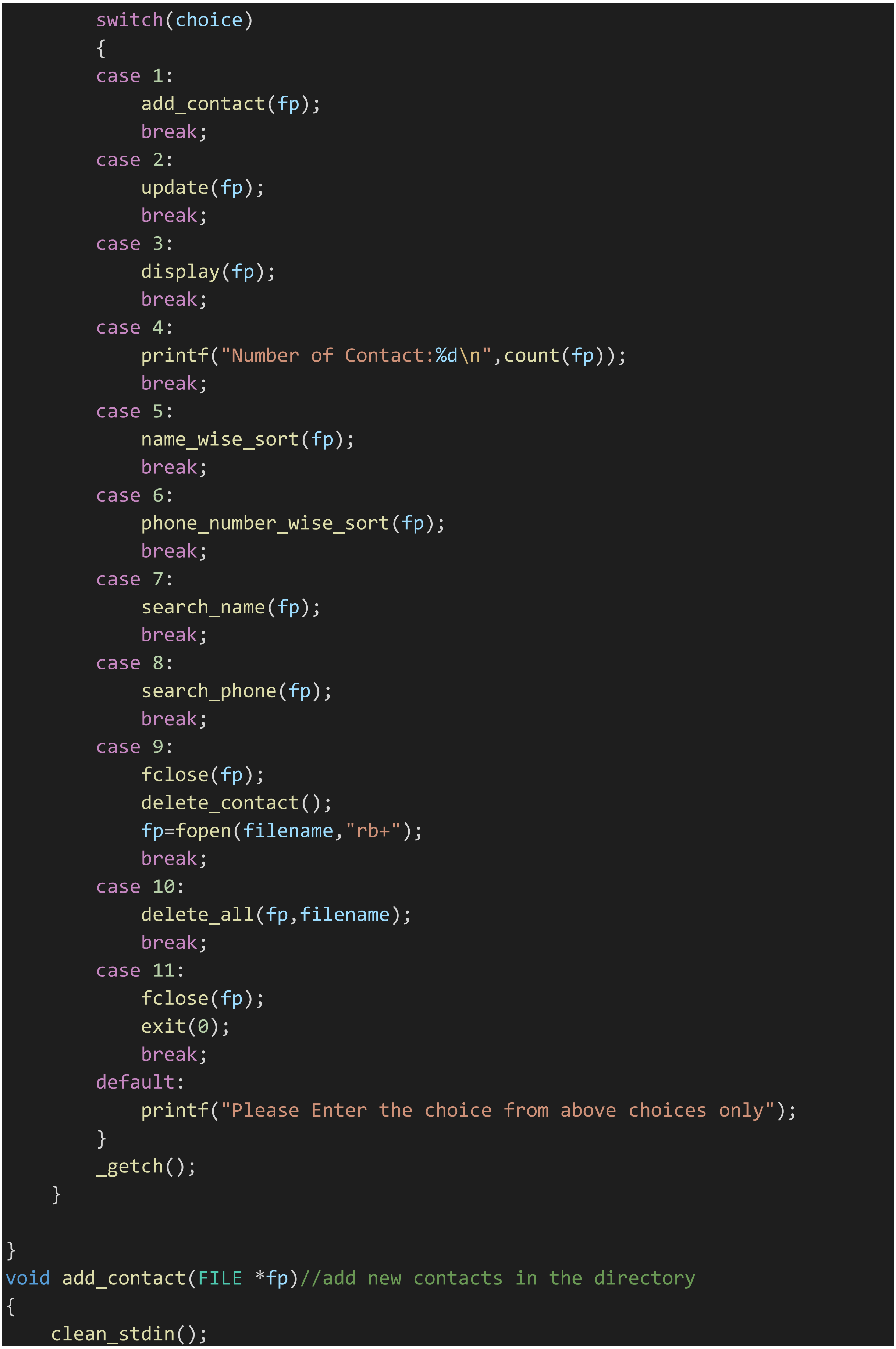
The concepts or topics that are used in the project are structures, file handling, strings and sorting and some basic c. In the file handling, the widely used functions are fread and fwrite. And along with them some other include fseek, rename and remove. And in sorting bubble sorting is used for creating the list name wise and phone number wise.

Code

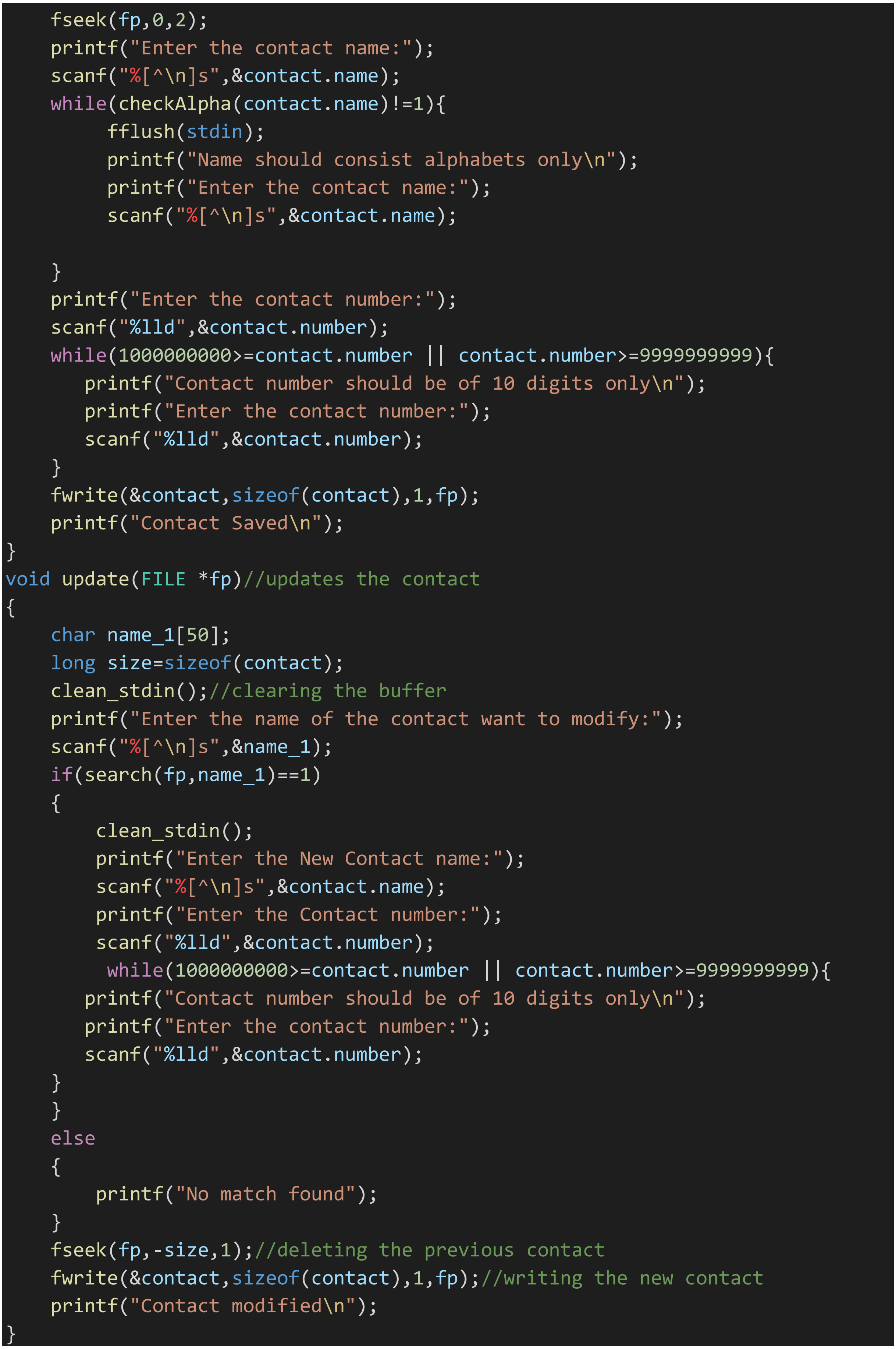
```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include<conio.h>
#include <ctype.h>

void add_contact(FILE *fp);//declaring functions prototype
void update(FILE *fp);
int search(FILE *fp,char name_1[]);
void search_name(FILE *fp);
void search_phone(FILE *fp);
void display(FILE *fp);
```



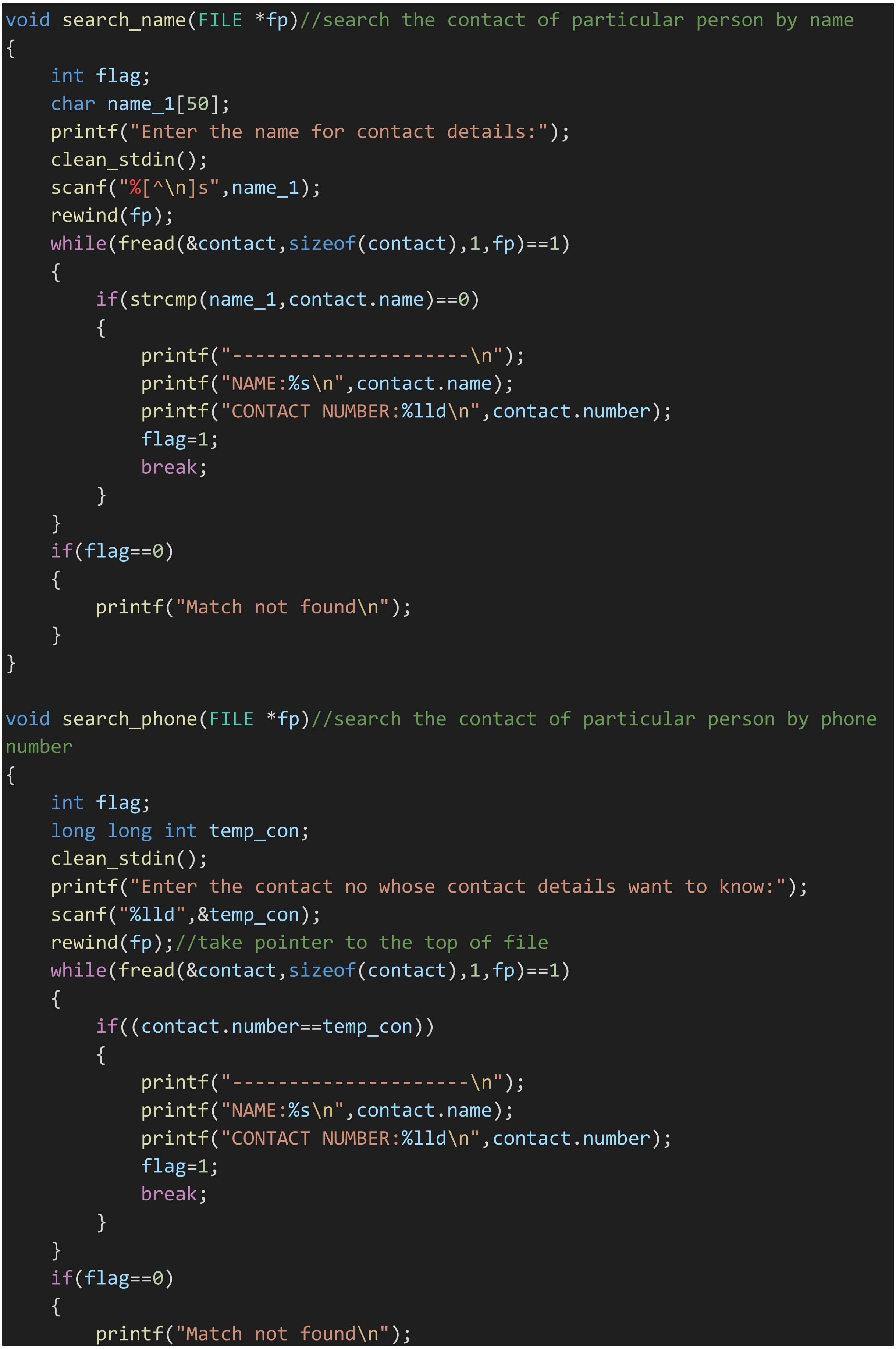
```c
void phone_number_wise_sort(FILE *fp);
void name_wise_sort(FILE * fp);
void delete_contact();
void delete_all(FILE *fp,char filename[20]);
int count(FILE *fp);
void clean_stdin();
int checkAlpha(char str1[]);
struct//defining datatype
{
    char name[40];
    long long  int number;
} contact,num_sort[200],alpha_sort[200],temp;
int main()
{
    system("cls");
    FILE *fp;
    int choice;
    char filename[]="dsa_project";

    fp=fopen(filename,"rb+");//open file in read and write binary form
    if(fp==NULL)
    {
        fp=fopen(filename,"wb+");//if file is null then open it in write binary form
        if(fp==NULL)
        {
            printf("Error in opening File\n");
            exit(1);
        }
    }

    while(1)//defining menu
    {
        system("cls");//clearing the console
        printf("\n\t\t\t\t\t\tPHONEBOOK DIRECTORY\n\n\n");
        printf("\t\t\t1.Add New Contact ");
        printf("\t\t\t2.Edit Contact\n\n");
        printf("\t\t\t3.Display All Contacts");
        printf("\t\t\t4.Total Contacts in Phone Book\n\n");
        printf("\t\t\t5.Display Contacts Name Wise");
        printf("\t\t6.Display Contacts Phone Number Wise\n\n");
        printf("\t\t\t7.Search Contact By Name");
        printf("\t\t8.Search Contact By Phone Number\n\n");
        printf("\t\t\t9.Delete Existing Contact");
        printf("\t\t10.Delete all the Contacts\n\n");
        printf("\t\t\t11.Exit\n\n");
        printf("Enter the choice\n");
        scanf("%d",&choice);
```



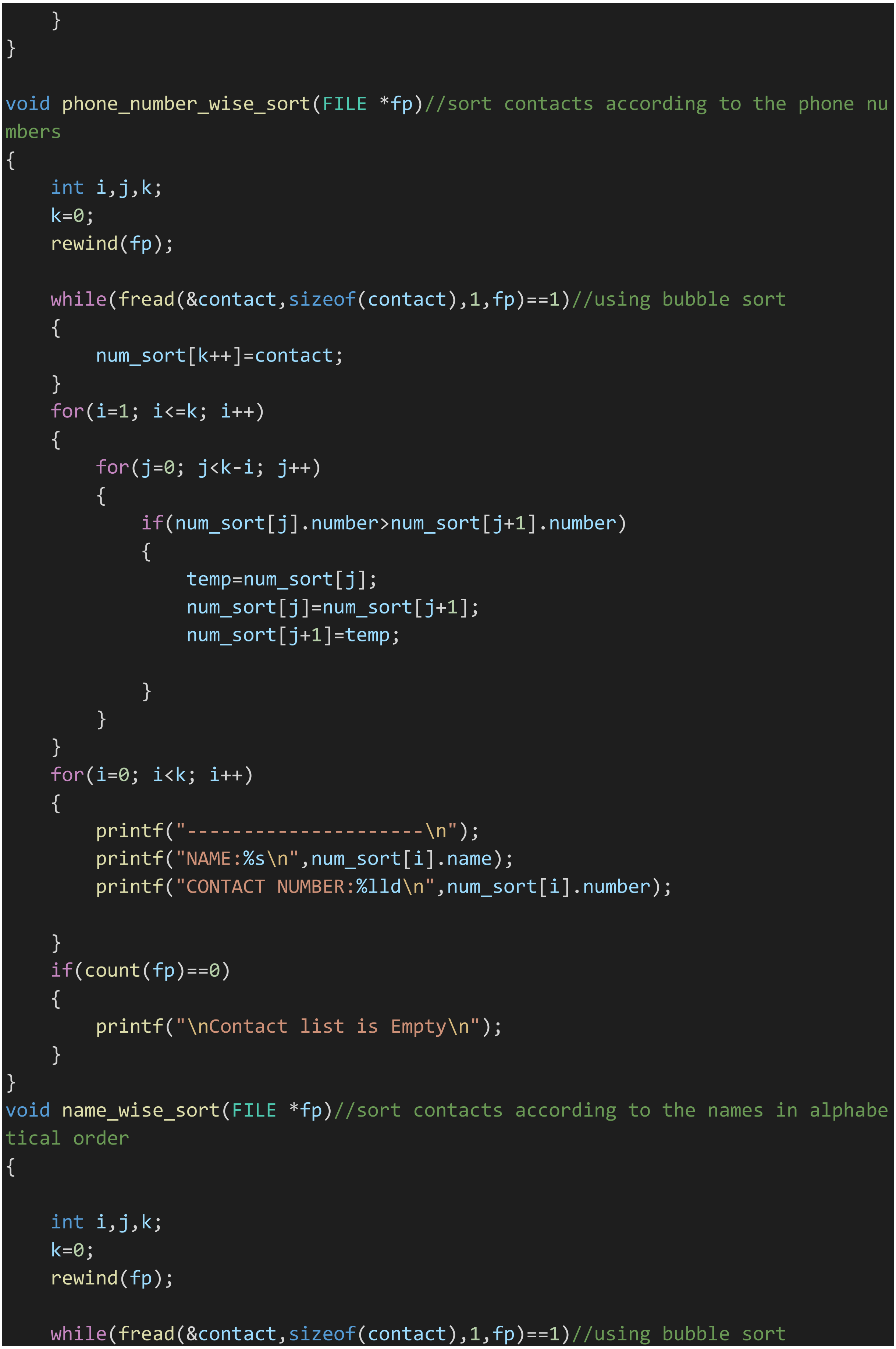
```
        switch(choice)
        {
        case 1:
            add_contact(fp);
            break;
        case 2:
            update(fp);
            break;
        case 3:
            display(fp);
            break;
        case 4:
            printf("Number of Contact:%d\n",count(fp));
            break;
        case 5:
            name_wise_sort(fp);
            break;
        case 6:
            phone_number_wise_sort(fp);
            break;
        case 7:
            search_name(fp);
            break;
        case 8:
            search_phone(fp);
            break;
        case 9:
            fclose(fp);
            delete_contact();
            fp=fopen(filename,"rb+");
            break;
        case 10:
            delete_all(fp,filename);
            break;
        case 11:
            fclose(fp);
            exit(0);
            break;
        default:
            printf("Please Enter the choice from above choices only");
        }
        _getch();
    }

}
void add_contact(FILE *fp)//add new contacts in the directory
{
    clean_stdin();
```



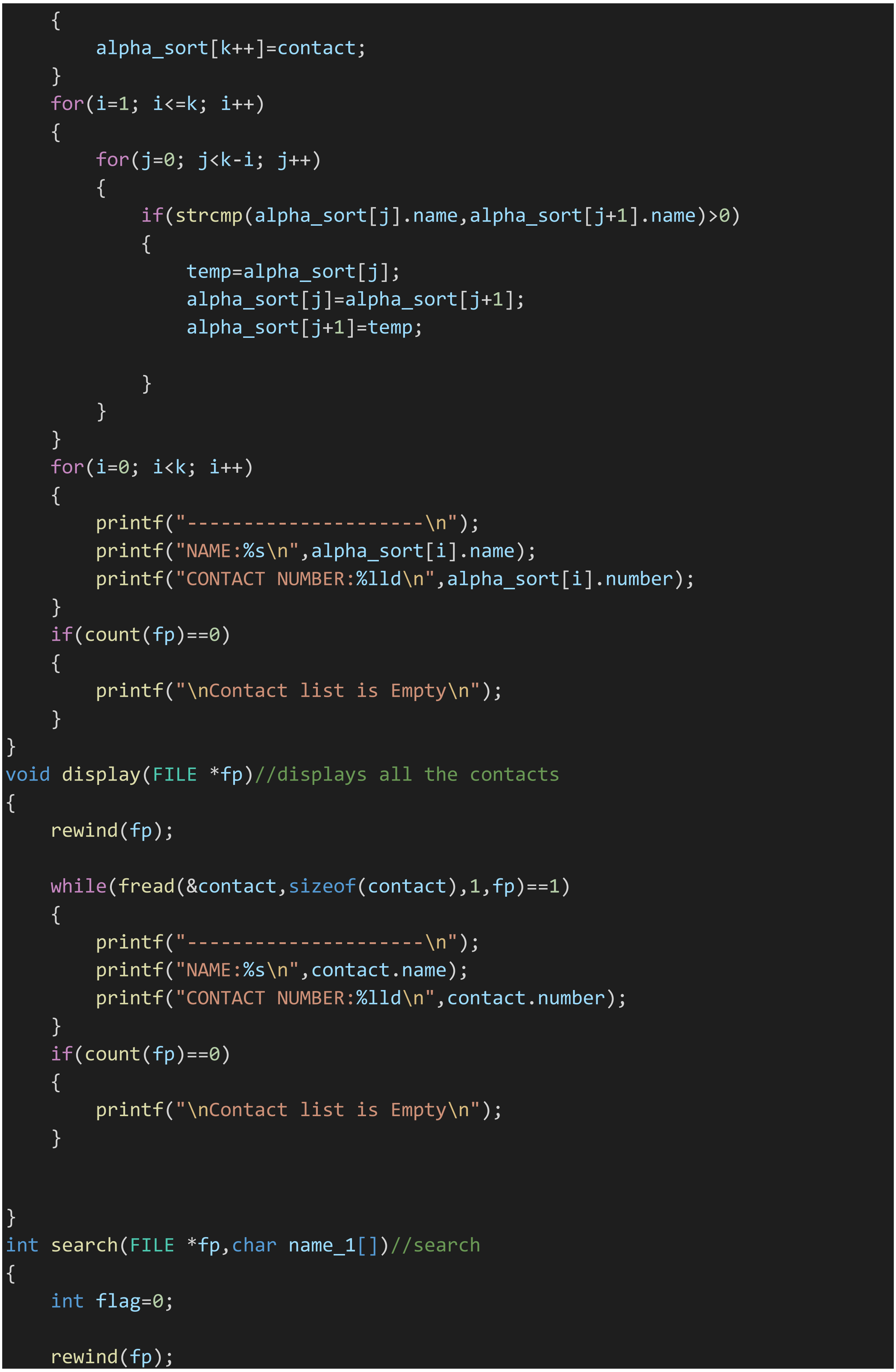
```
    fseek(fp,0,2);
    printf("Enter the contact name:");
    scanf("%[^\n]s",&contact.name);
    while(checkAlpha(contact.name)!=1){
         fflush(stdin);
         printf("Name should consist alphabets only\n");
         printf("Enter the contact name:");
         scanf("%[^\n]s",&contact.name);

    }
    printf("Enter the contact number:");
    scanf("%lld",&contact.number);
    while(1000000000>=contact.number || contact.number>=9999999999){
       printf("Contact number should be of 10 digits only\n");
       printf("Enter the contact number:");
       scanf("%lld",&contact.number);
    }
    fwrite(&contact,sizeof(contact),1,fp);
    printf("Contact Saved\n");
}
void update(FILE *fp)//updates the contact
{
    char name_1[50];
    long size=sizeof(contact);
    clean_stdin();//clearing the buffer
    printf("Enter the name of the contact want to modify:");
    scanf("%[^\n]s",&name_1);
    if(search(fp,name_1)==1)
    {
        clean_stdin();
        printf("Enter the New Contact name:");
        scanf("%[^\n]s",&contact.name);
        printf("Enter the Contact number:");
        scanf("%lld",&contact.number);
         while(1000000000>=contact.number || contact.number>=9999999999){
       printf("Contact number should be of 10 digits only\n");
       printf("Enter the contact number:");
       scanf("%lld",&contact.number);
    }
    }
    else
    {
        printf("No match found");
    }
    fseek(fp,-size,1);//deleting the previous contact
    fwrite(&contact,sizeof(contact),1,fp);//writing the new contact
    printf("Contact modified\n");
}
```



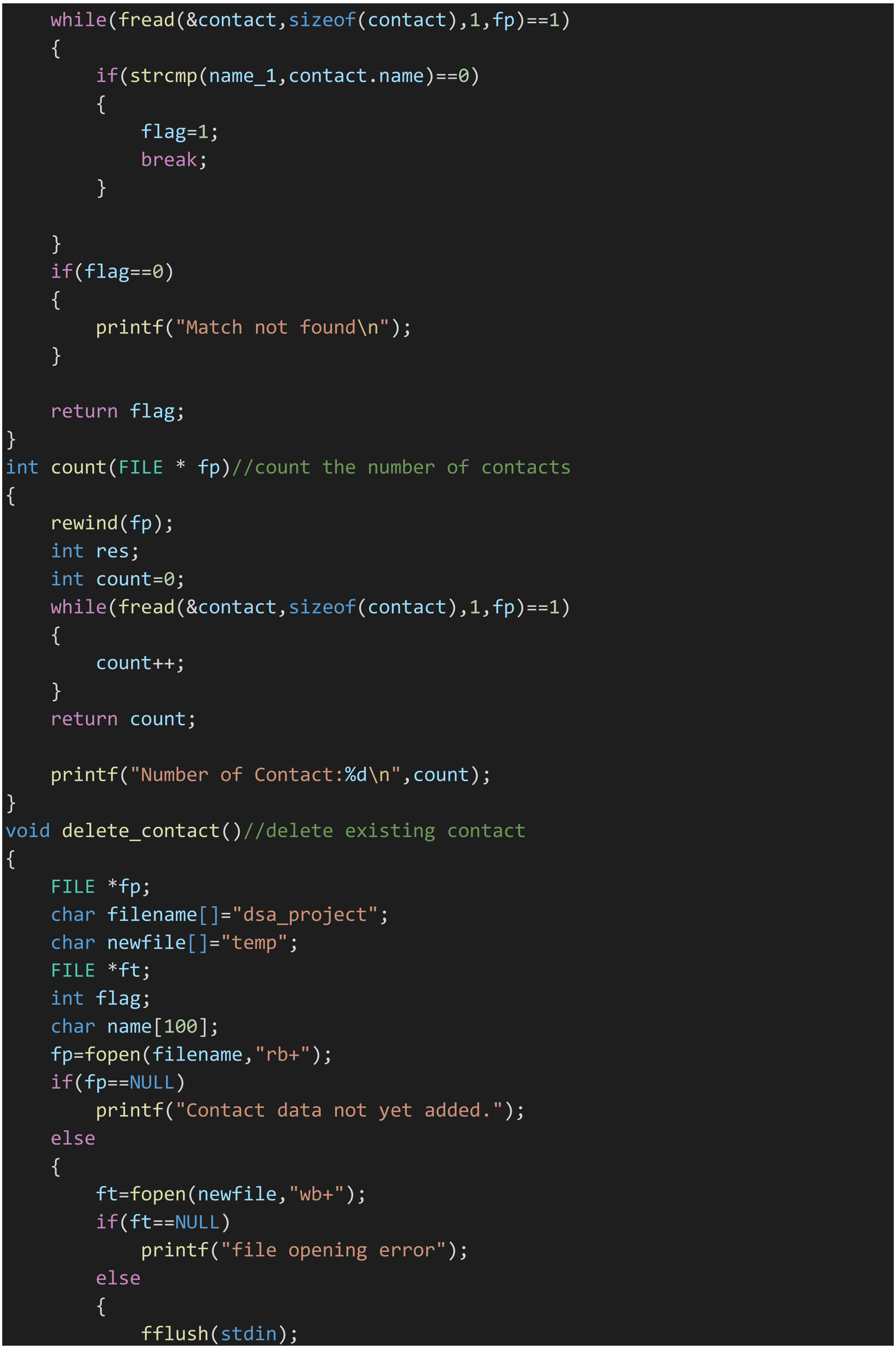
```c
void search_name(FILE *fp)//search the contact of particular person by name
{
    int flag;
    char name_1[50];
    printf("Enter the name for contact details:");
    clean_stdin();
    scanf("%[^\n]s",name_1);
    rewind(fp);
    while(fread(&contact,sizeof(contact),1,fp)==1)
    {
        if(strcmp(name_1,contact.name)==0)
        {
            printf("---------------------\n");
            printf("NAME:%s\n",contact.name);
            printf("CONTACT NUMBER:%lld\n",contact.number);
            flag=1;
            break;
        }
    }
    if(flag==0)
    {
        printf("Match not found\n");
    }
}

void search_phone(FILE *fp)//search the contact of particular person by phone number
{
    int flag;
    long long int temp_con;
    clean_stdin();
    printf("Enter the contact no whose contact details want to know:");
    scanf("%lld",&temp_con);
    rewind(fp);//take pointer to the top of file
    while(fread(&contact,sizeof(contact),1,fp)==1)
    {
        if((contact.number==temp_con))
        {
            printf("---------------------\n");
            printf("NAME:%s\n",contact.name);
            printf("CONTACT NUMBER:%lld\n",contact.number);
            flag=1;
            break;
        }
    }
    if(flag==0)
    {
        printf("Match not found\n");
```



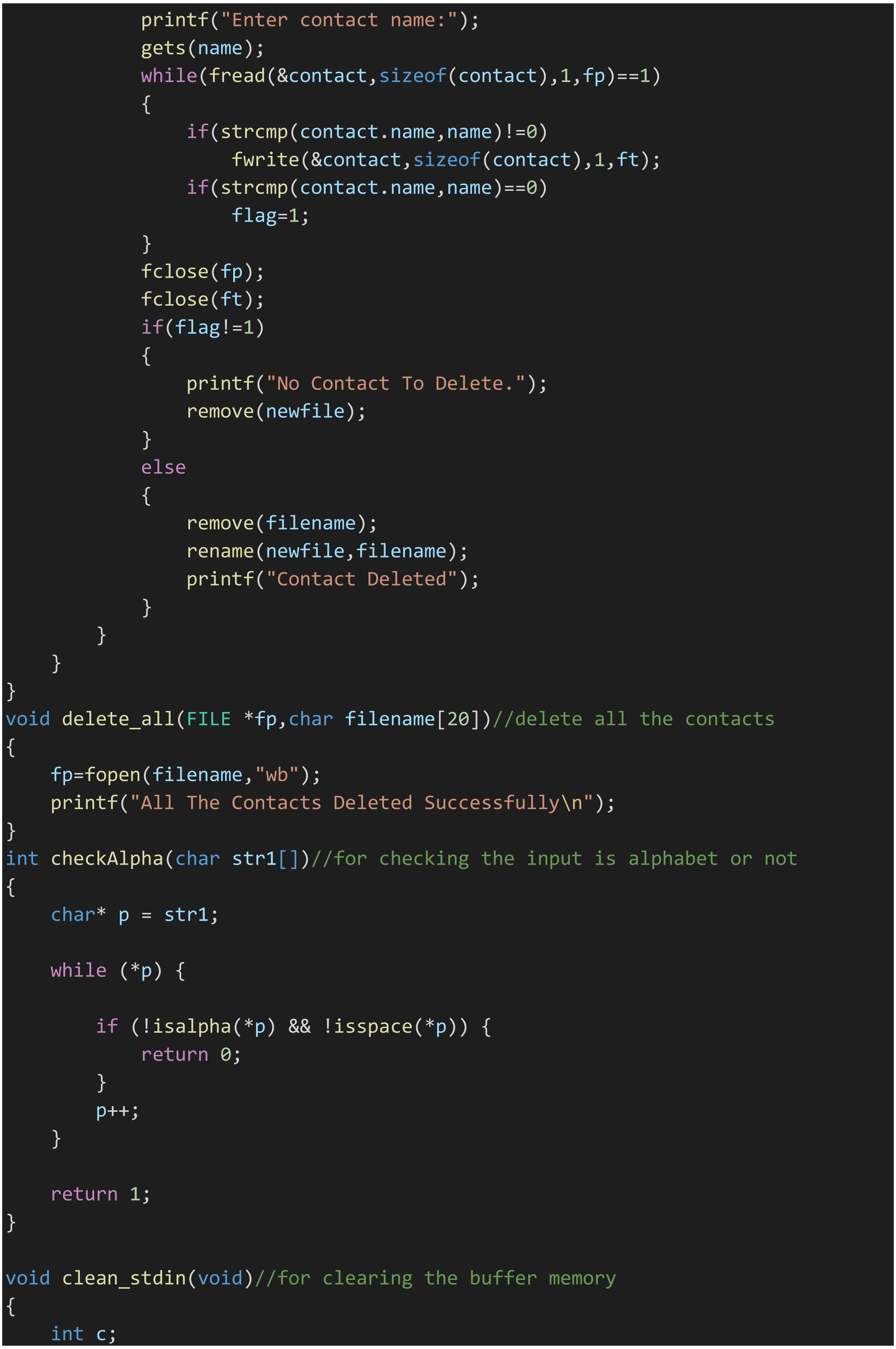
```c
    }
}

void phone_number_wise_sort(FILE *fp)//sort contacts according to the phone numbers
{
    int i,j,k;
    k=0;
    rewind(fp);

    while(fread(&contact,sizeof(contact),1,fp)==1)//using bubble sort
    {
        num_sort[k++]=contact;
    }
    for(i=1; i<=k; i++)
    {
        for(j=0; j<k-i; j++)
        {
            if(num_sort[j].number>num_sort[j+1].number)
            {
                temp=num_sort[j];
                num_sort[j]=num_sort[j+1];
                num_sort[j+1]=temp;

            }
        }
    }
    for(i=0; i<k; i++)
    {
        printf("---------------------\n");
        printf("NAME:%s\n",num_sort[i].name);
        printf("CONTACT NUMBER:%lld\n",num_sort[i].number);

    }
    if(count(fp)==0)
    {
        printf("\nContact list is Empty\n");
    }
}
void name_wise_sort(FILE *fp)//sort contacts according to the names in alphabetical order
{

    int i,j,k;
    k=0;
    rewind(fp);

    while(fread(&contact,sizeof(contact),1,fp)==1)//using bubble sort
```



```
    {
        alpha_sort[k++]=contact;
    }
    for(i=1; i<=k; i++)
    {
        for(j=0; j<k-i; j++)
        {
            if(strcmp(alpha_sort[j].name,alpha_sort[j+1].name)>0)
            {
                temp=alpha_sort[j];
                alpha_sort[j]=alpha_sort[j+1];
                alpha_sort[j+1]=temp;

            }
        }
    }
    for(i=0; i<k; i++)
    {
        printf("---------------------\n");
        printf("NAME:%s\n",alpha_sort[i].name);
        printf("CONTACT NUMBER:%lld\n",alpha_sort[i].number);
    }
    if(count(fp)==0)
    {
        printf("\nContact list is Empty\n");
    }
}
void display(FILE *fp)//displays all the contacts
{
    rewind(fp);

    while(fread(&contact,sizeof(contact),1,fp)==1)
    {
        printf("---------------------\n");
        printf("NAME:%s\n",contact.name);
        printf("CONTACT NUMBER:%lld\n",contact.number);
    }
    if(count(fp)==0)
    {
        printf("\nContact list is Empty\n");
    }


}
int search(FILE *fp,char name_1[])//search
{
    int flag=0;

    rewind(fp);
```

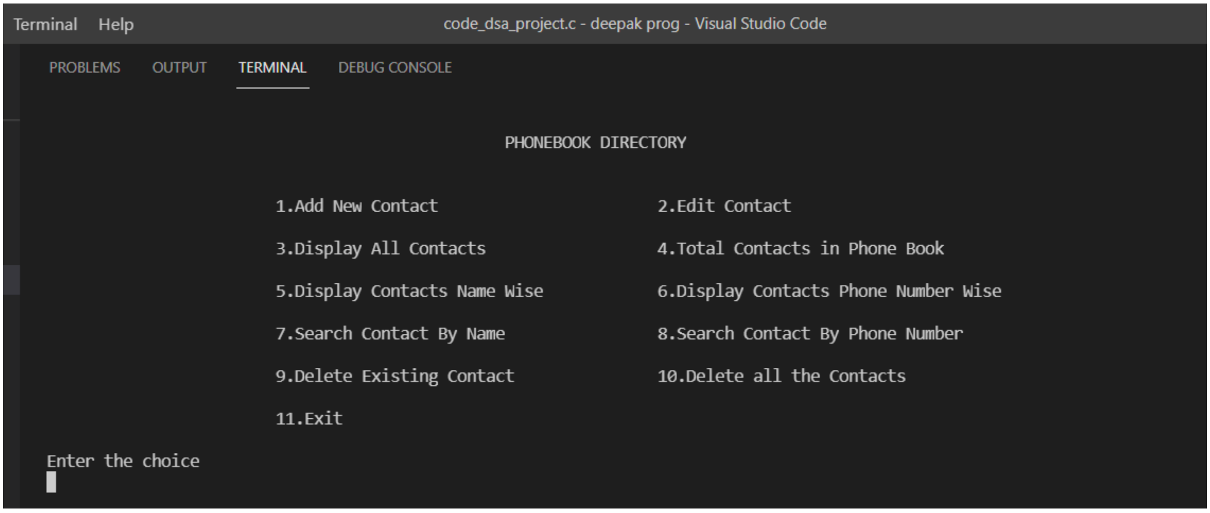


```
    while(fread(&contact,sizeof(contact),1,fp)==1)
    {
        if(strcmp(name_1,contact.name)==0)
        {
            flag=1;
            break;
        }

    }
    if(flag==0)
    {
        printf("Match not found\n");
    }

    return flag;
}
int count(FILE * fp)//count the number of contacts
{
    rewind(fp);
    int res;
    int count=0;
    while(fread(&contact,sizeof(contact),1,fp)==1)
    {
        count++;
    }
    return count;

    printf("Number of Contact:%d\n",count);
}
void delete_contact()//delete existing contact
{
    FILE *fp;
    char filename[]="dsa_project";
    char newfile[]="temp";
    FILE *ft;
    int flag;
    char name[100];
    fp=fopen(filename,"rb+");
    if(fp==NULL)
        printf("Contact data not yet added.");
    else
    {
        ft=fopen(newfile,"wb+");
        if(ft==NULL)
            printf("file opening error");
        else
        {
            fflush(stdin);
```



```
            printf("Enter contact name:");
            gets(name);
            while(fread(&contact,sizeof(contact),1,fp)==1)
            {
                if(strcmp(contact.name,name)!=0)
                    fwrite(&contact,sizeof(contact),1,ft);
                if(strcmp(contact.name,name)==0)
                    flag=1;
            }
            fclose(fp);
            fclose(ft);
            if(flag!=1)
            {
                printf("No Contact To Delete.");
                remove(newfile);
            }
            else
            {
                remove(filename);
                rename(newfile,filename);
                printf("Contact Deleted");
            }
        }
    }
}
void delete_all(FILE *fp,char filename[20])//delete all the contacts
{
    fp=fopen(filename,"wb");
    printf("All The Contacts Deleted Successfully\n");
}
int checkAlpha(char str1[])//for checking the input is alphabet or not
{
    char* p = str1;

    while (*p) {

        if (!isalpha(*p) && !isspace(*p)) {
            return 0;
        }
        p++;
    }

    return 1;
}

void clean_stdin(void)//for clearing the buffer memory
{
    int c;
    do
    {
        c = getchar();
    }
    while (c != '\n');
}
```

Sample Outputs



```
                                        PHONEBOOK DIRECTORY

                      1.Add New Contact                    2.Edit Contact

                      3.Display All Contacts               4.Total Contacts in Phone Book

                      5.Display Contacts Name Wise         6.Display Contacts Phone Number Wise

                      7.Search Contact By Name             8.Search Contact By Phone Number

                      9.Delete Existing Contact            10.Delete all the Contacts

                      11.Exit

Enter the choice
█
```

Contribution by group member

DARSHAN JAIN(B120022):

Code: Add, delete, search

Data Input: 50 Entries

Documentation: Title, Description, Contribution

DEEPAK YADAV(B120023):

Code: Update, sort, main

Data Input: 50 Entries

Documentation: Code, Output,References

References

Stackoverflow.com

Geeksfgorgeeks.com