TRIBHUVAN UNIVERSITY

INISTITUTE OF ENGINEERING

PULCHOWK CAMPUS

PULCHOWK, LATLITPUR

Project Report

STUDENT DATABASE MANAGEMENT

SUBMITTED BY:

Sardul Khanal (077BCT077)

Subham Shrestha (077BCT082)

Sudeep Subedi (077BCT083)

Sudhan Pandey (077BCT084)

Ujjwal Jha (077BCT092)

SUBMITTED TO:

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING

September, 2021

COPYRIGHT

ABSTRACT

In the present era, the use of computer application has vastly diverse around the globe. From small scale to large each and every sector, the use of computer has been an integral part of their respective system. Mainly in the management field the role of computer program for management is very crucial. So, we made this project related to database management of school by the use of C programming language.

This project aims to bring basic of database management in simple format. Basically the project mainly focus about search mechanism of information in database system. This project explores a new dimension in the database search engine in a simpler and understandable way. It is mainly applicable in management of student data in a school. Also it can be modified for other infinite application in database management.

ACKNOWLEDGEMENT

We would like to express our gratitude and appreciation to all those who gave us the possibility to complete this report. A special thanks to our respected teacher Ganesh Gautam sir whose help, stimulating suggestions and encouragement, helped us to coordinate our mini C project especially in debugging and finding the error in this report, without whose support our project might not be started .This step could be our first attempt to improve our programming skills in C programming. Also we gained a lot of knowledge regarding our subject matter. We would also like to acknowledge with much appreciation the crucial role of our seniors, who consistently gave precious advice during the completion of the project.

A special thanks goes to our classmates, CR, VCR who continuously helped us for the completion of this report. Any suggestion for the improvement of this mini project would be appreciable.

TABLE OF CONTENTS

Background information

1. Introduction

This is a c-project report from Sardul Khanal, Subham Shrestha, Sudeep Subedi, Sudhan Pandey and Ujjwal Jha, students of IOE PULCHOWK Bachelor in Computer Engineering batch 2077. This project is submitted on behalf of 1st year 1st semester C programming course offered by Department of Computer, Electronics and Communication course code CT 401.

Project Title: Student Database management

Compiler used: Dev. C++ and Code Blocks

Language used: C

2. Methodology and Basic Theory used in the program

This program is based on high level language i.e. c programming. In this project we use important parts of c programming which are control statement, looping, function, array, structure, pointer, data file.

2.1 C programming language

C is a procedural programming language. It was initially developed by Dennis Ritchie in the year 1972. It was mainly developed as a system programming language to write an operating system. Structured programming refers to programming that produce program with clean flow, clear and a degree of modularity or hierarchical structure is a simple, contained, versatile, excellent, efficient, fast general purpose language. C programs have ability to write concise programs even though a large number of operators are used. C includes a number of extensive library functions to enhance the basic instructions. It also avails a programmer to write additional functions of their own.

Some specific characteristics of Care:-

• Non-nest able function definitions.

• Variables may be hidden in nested blocks.

• A preprocessor for macro definition, source code file inclusion, and conditional compilation.

• Complex functionality such as I/O, string manipulation, and mathematical functions consistently delegated to library routines.

• A relatively small set of reserved keywords.

• Array indexing as a secondary notion, defined in terms of pointer arithmetic.

• Low-level access to computer memory by converting machine addresses to typed pointers.

• C-programs are highly portable.

2.2 Structure of a C program

Any C program consists of 6 main sections. They are:

2.2.1 Documentation Section

This part comprises of remark lines which incorporate the name of the program, the name of the developer and different subtleties like time and date.

Documentation area assists anybody with getting an outline of the program.

2.2.2 Link Section

The header files for the functions that are utilized in the program are included in the link section.

2.2.3 Definition Section

The defining section contains all of the symbolic constants.

2.2.4 Global Declaration Section

In the global declaration section, global variables that can be used anywhere in the program are declared. The user-defined functions are also declared in this section (function prototyping).

2.2.5 Main Function Section

A declaration part and an Execution part. The declaration part is the part where all the variables are declared. The execution part begins with the curly brackets and ends with the curly close bracket.

2.2.6 Subprogram Section

If the program is a multifunction program, the subprogram section will contain all of the user-defined functions that are called in the main () function. User-defined functions are usually inserted right after the main () function, but they can be placed wherever.

2.3 Control Statement

Control statements control the flow of execution of the statements of a program. The various types of control statements in C language are as under:-

2.3.1 Sequential

2.3.2 Selective

2.3.3 Iteration

2.3.1 Sequential

In sequential control, the C program statements are executed sequentially i.e., one after the from beginning to end.

2.3.2 Selective

In branching or selective structure a program can take different courses of action depending upon different circumstances.

2.3.2.1 If statement

2.3.2.2 If-else statement

2.3.2.3 else-if ladder

2.3.2.4 Switch case

2.3.2.1 If statement

This is the simplest form of decision control statement. In this form, a set of statements are executed only if the condition given with if evaluates to true. Its general syntax is as under:-

```
if(test expression)

{

True-block of Statement-block;

}

Statement x;
```

2.3.2.2 If-else statement

This is a bi-directional control statement. This statement is used to test a condition and take one of the two possible actions. If the condition evaluates to true then one statement (or block of statements) is executed otherwise other statement (or block of statements) is executed. The syntax of if-else statement is as under:-

```
if (condition)

{

block of statements;

}

else

{

block of statements;

}
```

2.3.2.3 Else if ladder

This is a type of nesting in which there is an if-else statement in every else part except the last else part. This type of nesting is called else if ladder. The general syntax of else if ladder is as follows:- If(condition1)

```
{
statementA;
}
else if(condition2)
{
statementB;
}
else if(condition3)
statementC;
}
else
{
statementD;
}
```

2.3.2.4 Switch case

Switch case statements are a substitute for long if statements and has more flexibility and a clearer format than else-if ladder. This statement is used to select one out of the several numbers of alternatives present in a block. This selection statement successively tests the value of an expression against a list of integer or character constants. When a match is found, the statements associated with that constant are executed. The general syntax of switch case statement is as follows:-

```
switch(expression)
{
case constant1:
statements;
break;
case constant2:
```

```
statements;
break;
case constant3:
statements;
break;
………………………
 ………………………
 case constantN:
statements;
break;

default :
statement;
}
```

2.3.3 Iteration

It repeats the set of statements until the condition for termination is met. Iteration statements in C are for, while and do-while.

2.3.3.1 for loop

The general syntax of for loop consists of three expressions separated by semicolons. It is given as follows:-

```
for(expression1;expression2;expression3)
{
Statement(s);
}
```

2.3.3.2 While loop

It is suited for the problems where it is not known in advance that how many times a statement or block of statements will be executed. The general syntax of while loop is as under:-

```
while(condition)
{
```

Statement(s);

}

2.3.3.3 Do while loop

It is similar to while loop and is used for the problems where it is not known in advance that how many times the statement or block of statements will be executed. However, unlike while loop, in case of do-while, firstly the statements inside the loop body are executed and then the condition is evaluated. As a result of which this loop is executed at least once even if the condition is initially false. The general syntax of do while loop is as under:-

do{

Statement(s);

}while(condition);

Conditional operator

It is a ternary operator and is used to perform simple conditional operations. It is used to do operations similar to if-else statement. The general syntax of conditional operator is as under:-
Test expression? expression1:expression2

2.4 Library functions

These function are the built-in functions i.e., they are predefined in the library of the C. These are used to perform the most common operations like calculations, updatation, etc. Some of the library functions are printf, scanf, etc. To use this functions in the program the user have to use associate header file associated to the corresponding function in the program.

2.5 User defined Function

C allows you to define functions according to your need. These functions are known as user-defined functions. A function is a block of code which performs a particular task. A function can be re used by a programmer in a given program any number of times.

2.6 Return statement

A function may or may not send back any value to the calling function .If it does it is through the return statement. The called function can only return only one value per call at most. The syntax of return statement is as follows:

Return;

2.7 Recursion

A function defined in c can call itself. This is called recursion. A function calling itself is also called ‘recursive’ function.

2.8 Array

An array is a way to store a fixed number of items of the same data type under a single name. Each data item of the array can be accessed by using a number called an “index” or “subscript”.

Syntax for array declaration:

dataType array Name[array size];

2.9 String

A string in C (also known as C string) is an array of characters, followed by a NULL character. A string is any series of characters that are interpreted literally by a script. For example, "hello world" and "LKJH019283" are both examples of strings.

2.9.1 String handling functions

strlen( ) Function: is used to find the length of a character string.

strcat( ) function: In C language concatenates two given strings. It concatenates source string at the end of destination string.

strcat (str2, str1); (str1 is concatenated at the end of str2.)

strcpy( ) function: Copies contents of one string into another string.

strcpy (str1, str2); (It copies contents of str2 into str1.)

strncat( ) function: In C language concatenates (appends) portion of one string at the end of another string.

strncat (str2, str1, 3); (First 3 characters of str1 is concatenated at the end of str2.)

2.10 Pointer

A pointer in C is a variable used to store the address of another variable. A pointer can also be used to refer to another pointer function. The pointer can increase/decrease, that is, point to the next/previous storage unit. Pointers are designed to save memory space and speed up execution time.

Declaration of pointer:

dataType *ptr;

Int *j; (declares the variable j of type int in pointer)

2.11 Structure

A structure is a user-defined data type in C that allows us to combine data of different types. The structure helps create a more complex data type. It is similar to an array, but multiple type of data is stored in structure. Due to their structure, data is stored in the form of data records. The syntax for structure is as follows:-

```
struct structure_name {
Data_type variable_name 1;
Data_type variable_name 2;
Data_type variable_name 3;
...................................
Data_type variable_name n;
};
```

2.12 Files

A file is an assortment of connected data. File programming is totally different from Console I/O. In console I/O knowledge may be browse and displayed onto the screen however don't seem to be keep for future use whereas in file I/O, knowledge may be saved onto files in order that the data will be retrieved for the long run usage.

Many applications require that information be written to or read from an auxiliary memory device. Such information is stored on the memory device in the form of a data file. The data files allow us to store information permanently and to access and alter that information whenever necessary.

Mode	Meaning of mode
w	Opening for write. If the file with specified filename currently exists it will be destroyed and new file is created in its place.
r	Opening for read
a	Open an existing file for appending. A new file will be created if the file with the specific filename does not exit.
r+	Opening for both reading and writing
w+	Open a new file for both reading and writing
a+	Open an existing file for reading and writing. A new file will be created if the file with the specified filename does not exist.

Opening a file:

We can use the fopen( ) function to create a new file or to open an existing file. This call will initialize an object of the type FILE, which contains all the information necessary to control the stream. The prototype of this function call is as follows:

file_pointer = fopen("path_string_for_file", "mode_string"); Closing a file: To close a file, use the fclose( ) function. The prototype of this function is: fclose(file_pointer);

In C programming language, record IO operations can be performed on a file using functions like fwrite() and fread() using following syntax:

fwrite (&structure_or_array_variable,sizeof_structure_or_array_variable,number_of structure_or_array,file_pointer);

The function fwrite() writes the contents of the structure or array into the file pointed by file pointer.

fread (&structure_or_array_variable,sizeof_structure_or_array_variable,number_of structure_or_array,file_pointer);

The function fread() reads the contents of the file pointed by the file pointer and saves the read values into the structure variable.

3. ALGORITHM

Algorithm for ***main ()*** function.

```
1.  Start
2.  Ask for the pin from the admin.
3.  Is the Pin correct?
            If yes: go to step 4
            If no: display “wrong pin and quit”
4.  Display different options in the menu.
5.  Ask for code for each option.
6.  Is code = 1?
            If yes: call addUser function.
            If no: goto step 7
7.  Is code = 2?
            If yes: call listUser function.
            If no: goto step 8
8.  Is code = 3?
            If yes:call searchUserWithFirstNames function.
            If no: goto step 9
9.  Is code = 4?
            If yes: call searchUserWithLastName function.
            If no: goto step 10
10. Is code = 5?
            If yes: call searchUserWithRegistrationNumber function.
            If no: goto step 11
11. Is code = 6?
            If yes: call EdituserWtihReg function.
            If no: goto step 12
12. Do you want to continue or quit?
13. Ask for value.
14. Is value = 0?  If yes: go to step 15
                   If no: goto step 4
15. End program
```

Algorithm for ***addUser ()*** function.

```
1. Start
2. Declare name, address, age, gender, phonenumber, grade, section,reg number.
3. Ask for details: name, address, age, gender, phonenumber, grade, section,reg number.
4. Open a file in read and write able format
5. Write the details in to the file.
6. End
```

Algorithm for ***listUser()*** function.

```
1. Start
2. Declare name, address, age, gender, phonenumber, grade, section,reg number.
3. Open a file in read format
4. Is it the end of the file?
      a. If no : go to step 5
      b. If yes: go to step 8
5. Read  details from file: name, address, age, gender, phonenumber, grade, section,reg
   number, until end of file
6. Display all the records.
7. Goto step 4
8. End
```

Algorithm for ***searchUserWithFirstName()*** function.

```
1. Start
2. Declare nameToSearch.
3. Declare name, address, age, gender, phonenumber, grade, section,reg number.
4. Ask nameToSearch.
5. Open a file in read format
6. Is it the end of the file?
      a. If no : go to step 7
      b. If yes: go to step 10
7. Read  details from file: name, address, age, gender, phonenumber, grade, section,reg
   number, until end of file
8. Is name = nameToSearch?
      a. If no : go to step 9
      b. If yes: display records and increase counter.
9. Go to step 6
10. Is counter = 0?   If no : go to step 11.
                      If yes: display no resuls
11. End
```

Algorithm for ***searchUserWithLastName ()*** function.

```
1.  Start
2.  Declare lastnameToSearch.
3.  Declare name, address, age, gender, phonenumber, grade, section,reg number.
4.  Ask nameToSearch.
5.  Open a file in read format
6.  Is it the end of the file?
        a.  If no : go to step 7
        b.  If yes: go to step 10
7.  Read  details from file: name, address, age, gender, phonenumber, grade, section,reg
    number, until end of file
8.  Is lastname = lastnameToSearch?
        a.  If no : go to step 9
        b.  If yes: display records and increase counter.
9.  Go to step 6
10. Is counter = 0?
        a.  If no : go to step 11.
        b.  If yes: display no results
11. End
```

Algorithm for ***searchUserWithRegistrationNumber ()*** function.

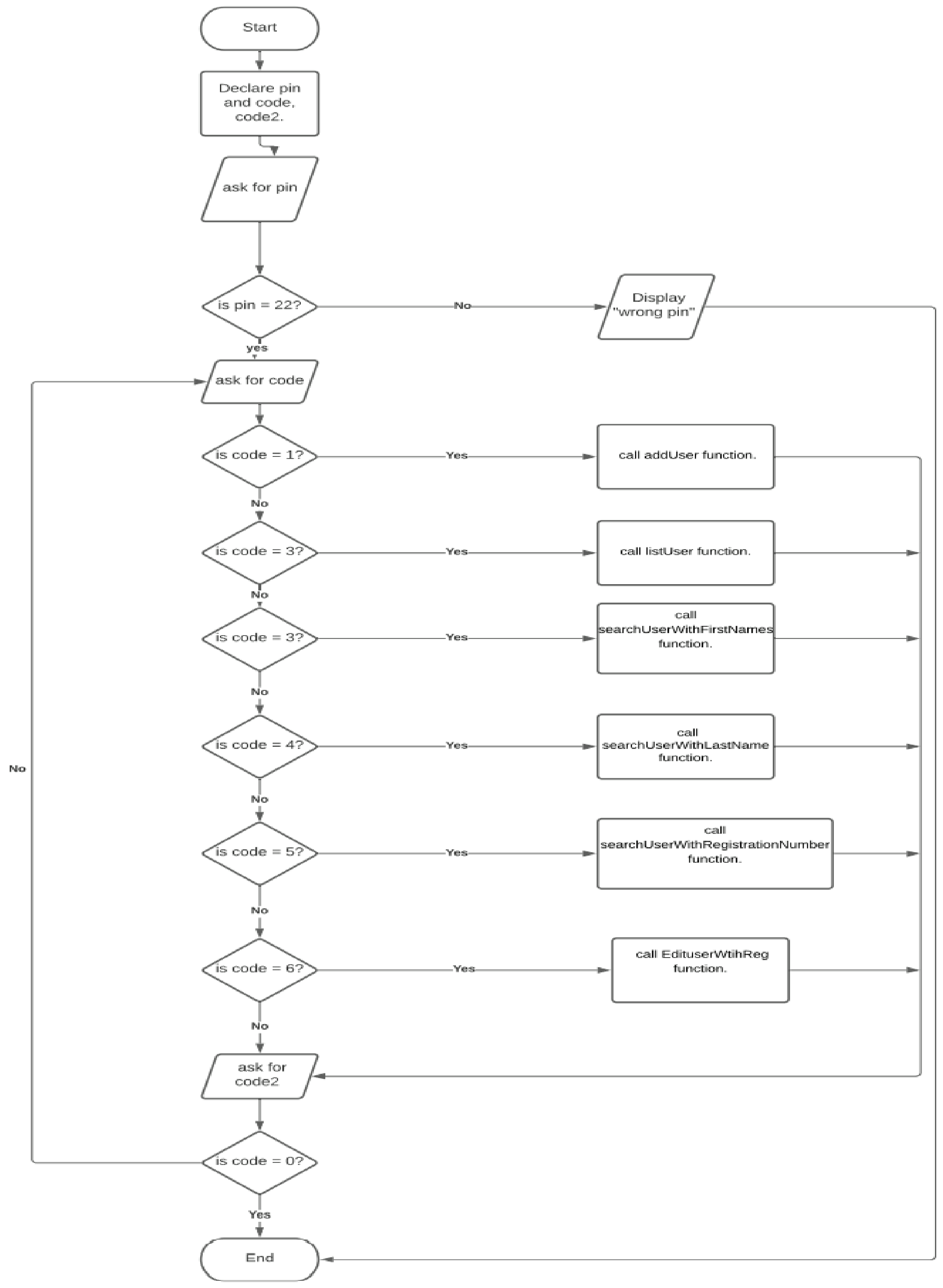
```
1.  Start
2.  Declare RegN.
3.  Declare name, address, age, gender, phonenumber, grade, section,regnumber.
4.  Ask nameToSearch.
5.  Open a file in read format
6.  Is it the end of the file?
        a.  If no : go to step 7
        b.  If yes: go to step 10
7.  Read  details from file: name, address, age, gender, phonenumber, grade,
    section,regnumber, until end of file
8.  Is RegN = regnumber?
        a.  If no : go to step 9
        b.  If yes: display records and increase counter.
9.  Go to step 6
10. Is counter = 0?
        a.  If no : go to step 11.
        b.  If yes: display no results
11. End
```

Algorithm for ***EdituserWtihReg()*** function.

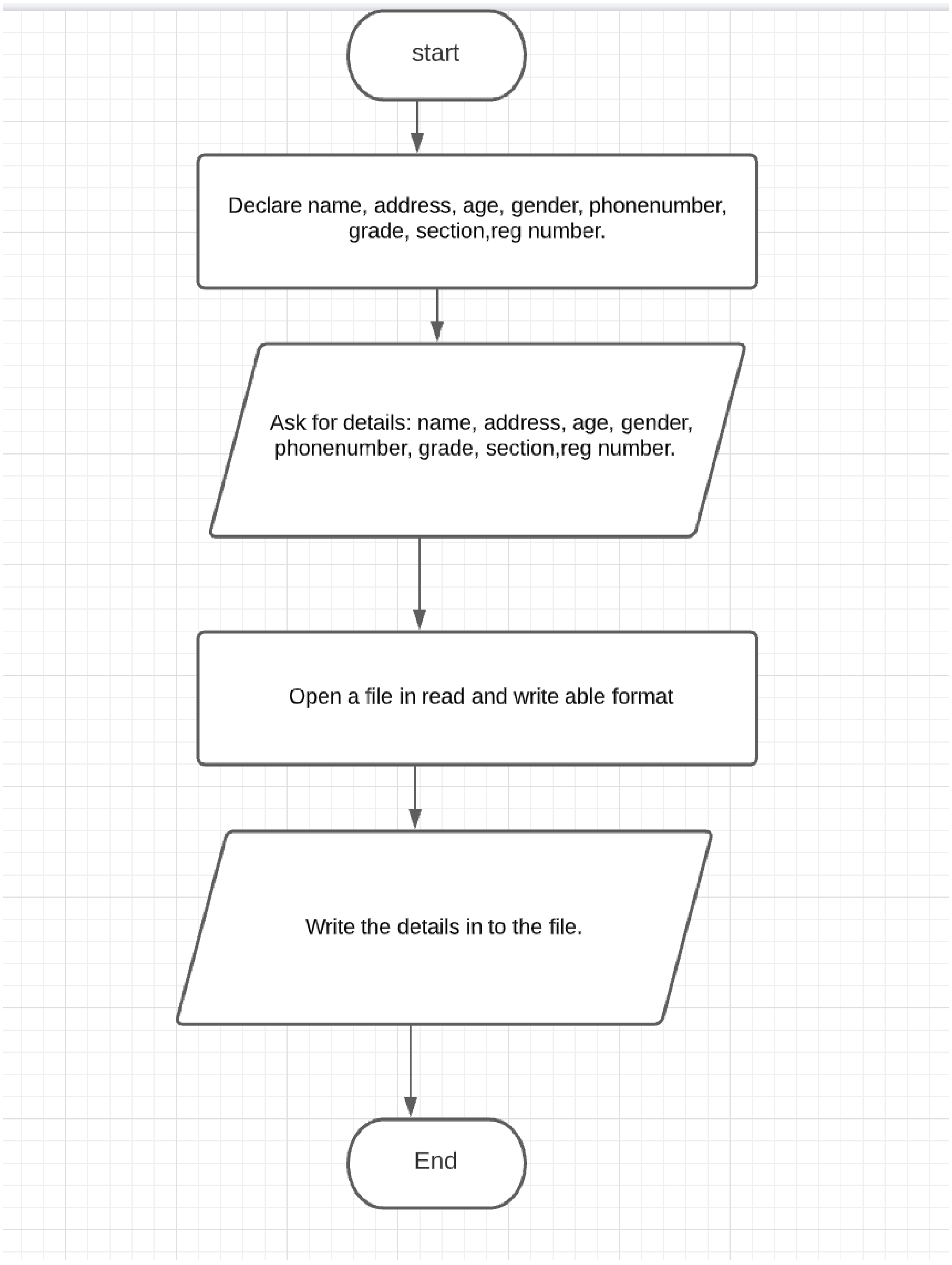
```
1.  Start
2.  Declare RegN.
3.  Declare name, address, age, gender, phonenumber, grade, section,regnumber.
4.  Ask nameToSearch.
5.  Open a file in read and write format
6.  Is it the end of the file?
        a.  If no : go to step 7
        b.  If yes: go to step 12
7.  Read  details from file: name, address, age, gender, phonenumber, grade,
    section,regnumber, until end of file
8.  Is RegN = regnumber?
        a.  If no : go to step 9
        b.  If yes: increase counter and goto step 10
9.  Go to step 6
10. Ask for details: name, address, age, gender, phonenumber, grade, section,regnumber.
11. Edit the details to the file.
12. Is counter = 0?
        a.  If no : go to step 13.
        b.  If yes: display no results found.
13. End
```

4. FLOWCHART

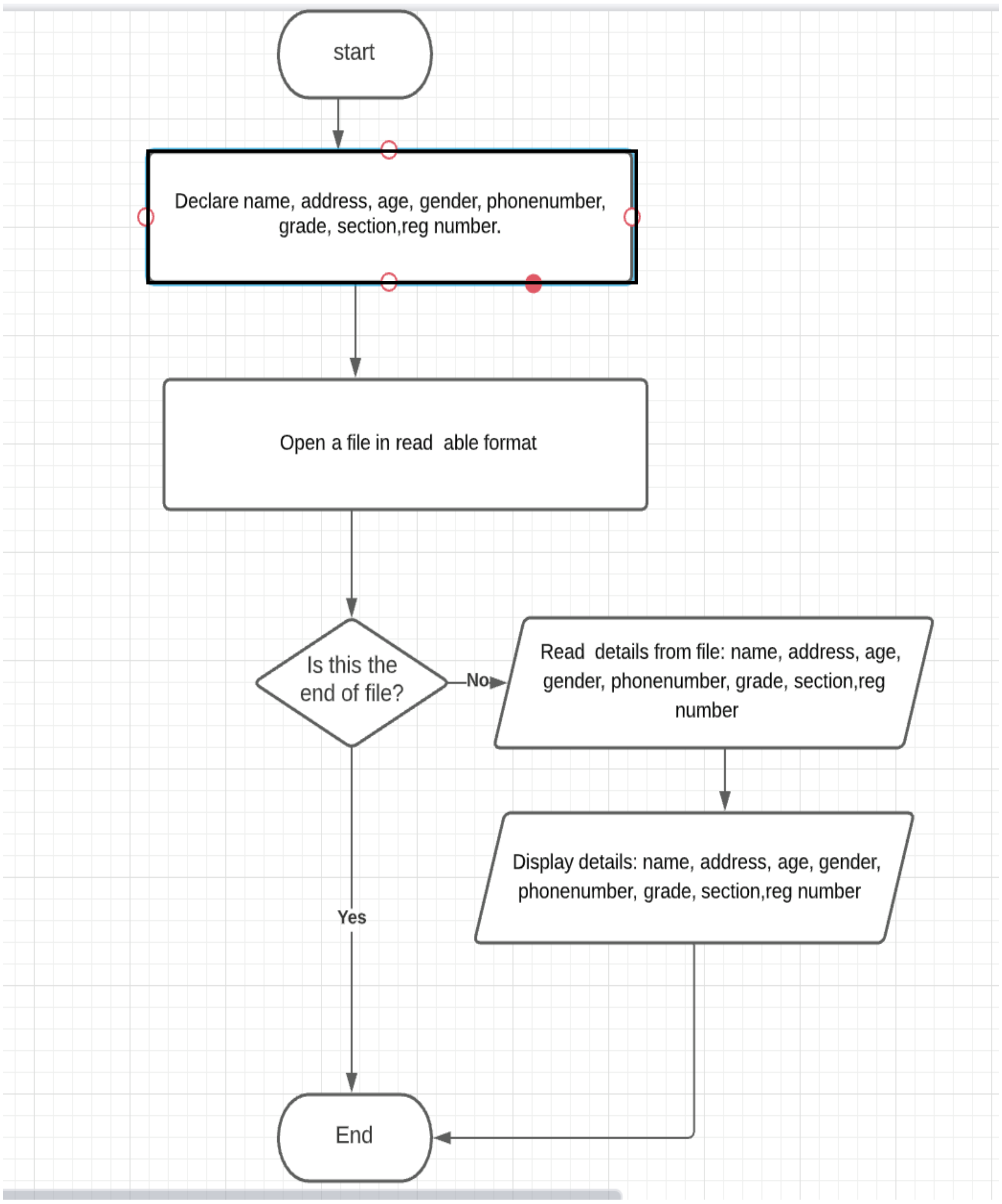
Flowchart of main:



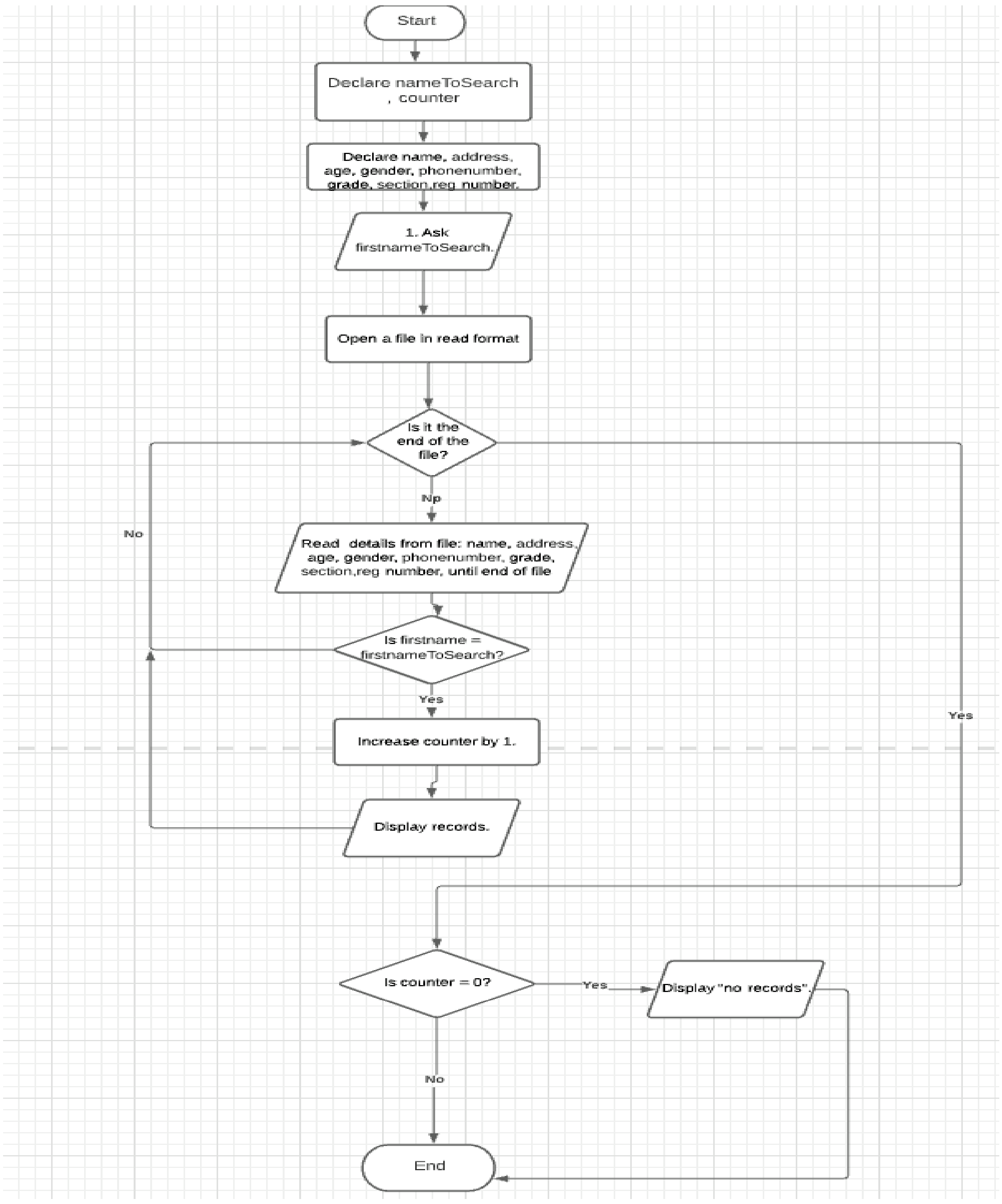
Flowchart for adding user:


start
Declare name, address, age, gender, phonenumber, grade, section,reg number.
Ask for details: name, address, age, gender, phonenumber, grade, section,reg number.
Open a file in read and write able format
Write the details in to the file.
End

Flowchart for listing user:

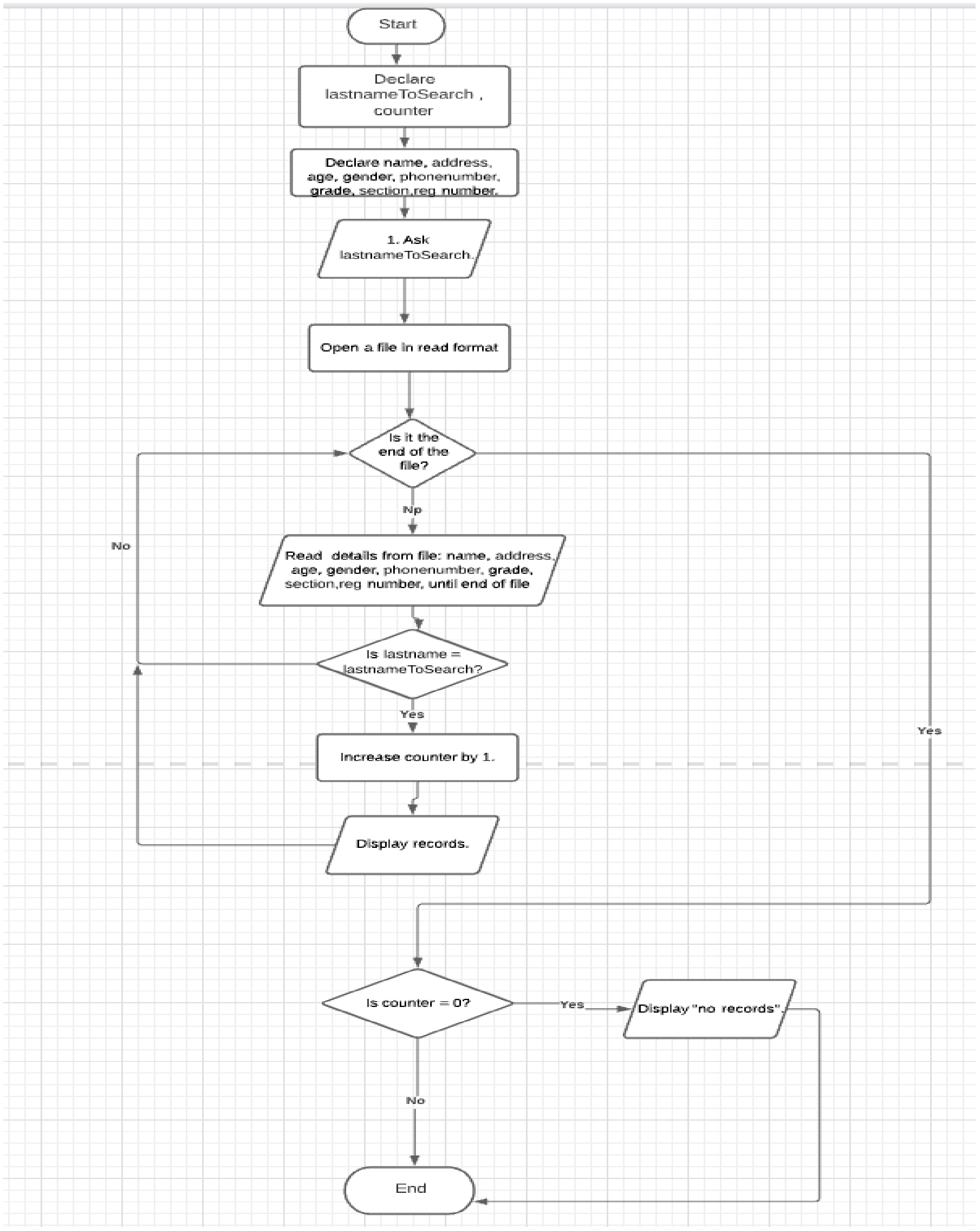

start
Declare name, address, age, gender, phonenumber, grade, section,reg number.
Open a file in read able format
Is this the end of file?
No
Read details from file: name, address, age, gender, phonenumber, grade, section,reg number
Display details: name, address, age, gender, phonenumber, grade, section,reg number
Yes
End

Flowchart for search by first name:



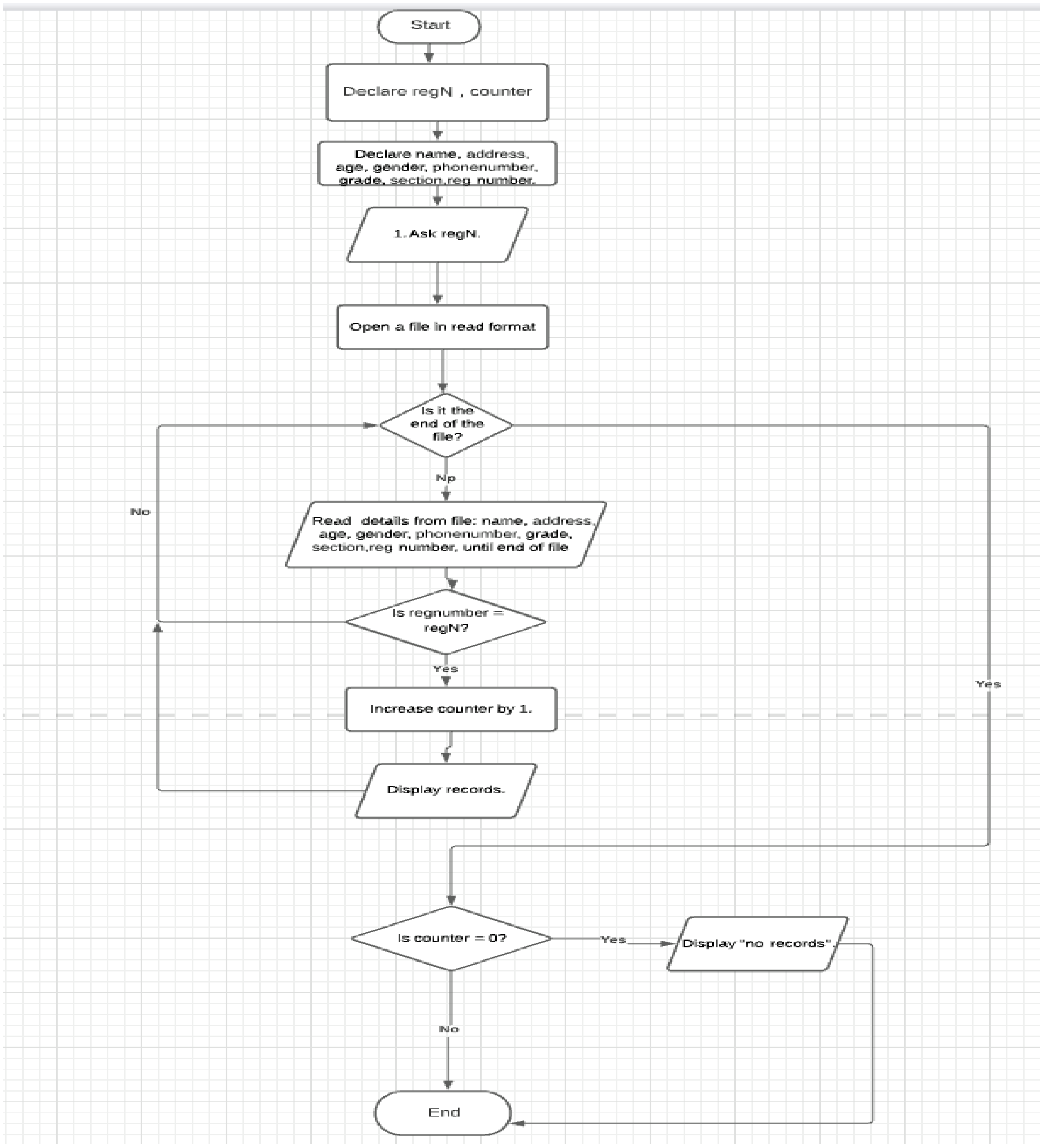
Start
Declare nameToSearch , counter
Declare name, address, age, gender, phonenumber, grade, section,reg number.
1. Ask firstnameToSearch.
Open a file in read format
Is it the end of the file?
No
Read details from file: name, address, age, gender, phonenumber, grade, section,reg number, until end of file
No
Is firstname = firstnameToSearch?
Yes
Increase counter by 1.
Display records.
Yes
Is counter = 0?
Yes
Display "no records".
No
End

Flowchart for search by last name:

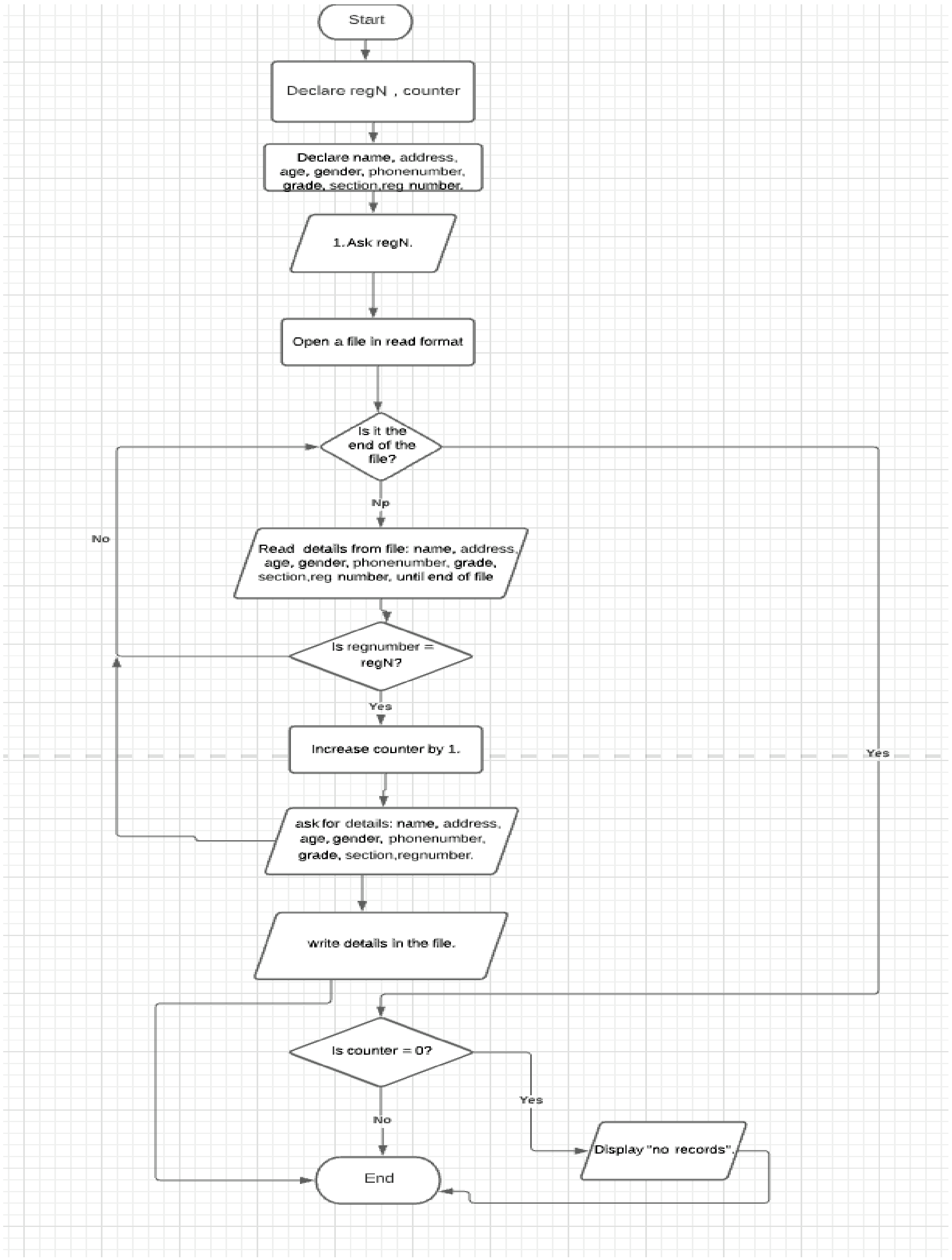


Start
Declare lastnameToSearch , counter
Declare name, address, age, gender, phonenumber, grade, section,reg number.
1. Ask lastnameToSearch.
Open a file in read format
Is it the end of the file?
No
Read details from file: name, address, age, gender, phonenumber, grade, section,reg number, until end of file
Is lastname = lastnameToSearch?
No
Yes
Increase counter by 1.
Display records.
Yes
Is counter = 0?
Yes
Display "no records".
No
End

Flowchart for search by registration number:


Start
Declare regN , counter
Declare name, address, age, gender, phonenumber, grade, section,reg number.
1. Ask regN.
Open a file in read format
Is it the end of the file?
Np
Read details from file: name, address, age, gender, phonenumber, grade, section,reg number, until end of file
No
Is regnumber = regN?
Yes
Increase counter by 1.
Display records.
Yes
Is counter = 0?
Yes
Display "no records".
No
End

Flowchart for editing user:


Start
Declare regN , counter
Declare name, address, age, gender, phonenumber, grade, section,reg number.
1. Ask regN.
Open a file in read format
Is it the end of the file?
Np
No
Read details from file: name, address, age, gender, phonenumber, grade, section,reg number, until end of file
Is regnumber = regN?
Yes
Increase counter by 1.
Yes
ask for details: name, address, age, gender, phonenumber, grade, section,regnumber.
write details in the file.
Is counter = 0?
Yes
No
Display "no records".
End

5. SOURCE CODE

APPENDIX:

```
#include <stdio.h>
#include <string.h>
int cont;
int compare(char word1[], char word2[])
{
    int i = 0;
    while (word1[i] != '\0' && word2[2] != '\0')
    {
        if (word1[i] != word2[i])
        {
            return 0;
            break;
        }
        i++;
    }
    return 1;
};
int addUser()
{
    printf("\n");
    FILE *fp;
    fp = fopen("address.txt", "a");

    char firstname[20];
    char lastname[20];
    char address[40];
    long int phonenumber;
    char gender[20];
    char grade[40];
    char section[20];
    int age;
    long int registrationNumber;
    char email[100];

    printf("\t\t\t");
    printf("Enter the firstname.\n");
    printf("\t\t\t");
    scanf("%s", firstname);
    printf("\t\t\t");
    printf("Enter the lastname.\n");
    printf("\t\t\t");
    scanf("%s", lastname);
    printf("\t\t\t");
    printf("Enter the address.\n");
    printf("\t\t\t");
    scanf("%s", address);
    printf("\t\t\t");
    printf("Enter the phonenumber.\n");
    printf("\t\t\t");
    scanf("%ld", &phonenumber);
    printf("\t\t\t");
    printf("Enter the gender (Male/Female/Other).\n");
    printf("\t\t\t");
```

```
    scanf("%s", gender);
    printf("\t\t\t");
    printf("Enter the grade.\n");
    printf("\t\t\t");
    scanf("%s", grade);
    printf("\t\t\t");
    printf("Enter the section.\n");
    printf("\t\t\t");
    scanf("%s", section);
    printf("\t\t\t");
    printf("Enter the age.\n");
    printf("\t\t\t");
    scanf("%d", &age);
    printf("\t\t\t");
    printf("Enter the registrationnumber.\n");
    printf("\t\t\t");
    scanf("%ld", &registrationNumber);
    printf("\t\t\t");
    printf("Enter the email.\n");
    printf("\t\t\t");
    scanf("%s", email);


    fprintf(fp, "Firstname:%s Lastname:%s Address:%s Phonenumber:%ld Gender:%s Grade:%s
Section:%s Age:%d Registrationnumber:%ld Email:%s\n",
            firstname, lastname, address, phonenumber, gender, grade, section, age,
registrationNumber, email);
    fclose(fp);
    printf("\n Successfully created new record! \n");
    return 0;
```

```
}
int listUser()
{
    FILE *fp;
    fp = fopen("address.txt", "r");

    char firstname[20];
    char lastname[20];
    char address[40];
    long int phonenumber;
    char gender[20];
    char grade[40];
    char section[20];
    int age;
    long int registrationNumber;
    char email[100];

    while (fscanf(fp, "Firstname:%s Lastname:%s Address:%s Phonenumber:%ld Gender:%s
Grade:%s Section:%s Age:%d Registrationnumber:%ld Email:%s\n",
                firstname, lastname, address, &phonenumber, gender, grade, section, &age,
&registrationNumber, email) != EOF)
    {
        printf("\t\t\t");
        printf("---------------------------");
        printf("\n\n");
        printf("\t\t\t");
        printf("Firstname:%s \n\t\t\tLastname:%s \n\t\t\tAddress:%s \n\t\t\tPhonenumber:%ld
\n\t\t\tGender:%s \n\t\t\tGrade:%s \n\t\t\tSection:%s \n\t\t\tAge:%d
\n\t\t\tRegistrationnumber:%ld \n\t\t\tEmail:%s",
                firstname, lastname, address, phonenumber, gender, grade, section, age,
registrationNumber, email);

        printf("\n\n");

    }

    return 0;

}

int searchWithFirstname()

{

    char fname[20];

    printf("\t\t\t");


    printf("Enter the firstname.\n");

    printf("\t\t\t");


    scanf("%s", fname);


    FILE *fp;

    fp = fopen("address.txt", "r");


    char firstname[20];

    char lastname[20];

    char address[40];

    long int phonenumber;

    char gender[20];

    char grade[40];

    char section[20];

    int age;

    long int registrationNumber;

    char email[100];
    int results = 0;


    while (fscanf(fp, "Firstname:%s Lastname:%s Address:%s Phonenumber:%ld Gender:%s
Grade:%s Section:%s Age:%d Registrationnumber:%ld Email:%s\n",
                  firstname, lastname, address, &phonenumber, gender, grade, section, &age,
&registrationNumber, email) != EOF)
    {
        // printf("%d",compare(fname,firstname));
        if (compare(fname, firstname) == 1)
        {
            printf("\t\t\t");
            printf("---------------------------");
            printf("\n\n");
            printf("\t\t\t");
            printf("Firstname:%s \n\t\t\tLastname:%s \n\t\t\tAddress:%s \n\t\t\tPhonenumber:%ld
\n\t\t\tGender:%s \n\t\t\tGrade:%s \n\t\t\tSection:%s \n\t\t\tAge:%d
\n\t\t\tRegistrationnumber:%ld \n\t\t\tEmail:%s",
                   firstname, lastname, address, phonenumber, gender, grade, section, age,
registrationNumber, email);
            printf("\n\n");
            results++;
        }
    }


    if (results == 0)
    {
        printf("\t\t\t");


        printf("User with this firstname doesnot exit.\n");
    }
    return 0;
}
int searchWithLastname()
{
    char lname[20];
    printf("\t\t\t");

    printf("Enter the lastname.\n");
    printf("\t\t\t");

    scanf("%s", lname);

    FILE *fp;
    fp = fopen("address.txt", "r");

    char firstname[20];
    char lastname[20];
    char address[40];
    long int phonenumber;
    char gender[20];
    char grade[40];
    char section[20];
    int age;
    long int registrationNumber;
    char email[100];
    int results = 0;

    while (fscanf(fp, "Firstname:%s Lastname:%s Address:%s Phonenumber:%ld Gender:%s
Grade:%s Section:%s Age:%d Registrationnumber:%ld Email:%s\n",
                firstname, lastname, address, &phonenumber, gender, grade, section, &age,
&registrationNumber, email) != EOF)
    {
        // printf("%d",compare(fname,firstname));
        if (compare(lname, lastname) == 1)
        {
            printf("\t\t");
            printf("---------------------------");
            printf("\n\n");
            printf("\t\t");
            printf("Firstname:%s \n\t\t\tLastname:%s \n\t\t\tAddress:%s \n\t\t\tPhonenumber:%ld
\n\t\t\tGender:%s \n\t\t\tGrade:%s \n\t\t\tSection:%s \n\t\t\tAge:%d
\n\t\t\tRegistrationnumber:%ld \n\t\t\tEmail:%s",
                firstname, lastname, address, phonenumber, gender, grade, section, age,
registrationNumber, email);
            printf("\n\n");
            results++;
        }
    }
    if (results == 0)
    {
        printf("\t\t\t");
        printf("User with this lastname doesnot exit.\n");
    }
    return 0;
}
int searchWithReg()
{
    long int regis;
    printf("\t\t\t");
    printf("Enter the registration number.\n");
    printf("\t\t\t");

    scanf("%ld", &regis);

    FILE *fp;
    fp = fopen("address.txt", "r");

    char firstname[20];
    char lastname[20];
    char address[40];
    long int phonenumber;
    char gender[20];
    char grade[40];
    char section[20];
    int age;
    long int registrationNumber;
    char email[100];
    int results = 0;

    while (fscanf(fp, "Firstname:%s Lastname:%s Address:%s Phonenumber:%ld Gender:%s
Grade:%s Section:%s Age:%d Registrationnumber:%ld Email:%s\n",
                  firstname, lastname, address, &phonenumber, gender, grade, section, &age,
&registrationNumber, email) != EOF)
    {
        if (regis == registrationNumber)
        {
            printf("\t\t\t");
            printf("---------------------------");

            printf("\n\n");

            printf("\t\t\t");

            printf("Firstname:%s \n\t\t\tLastname:%s \n\t\t\tAddress:%s \n\t\t\tPhonenumber:%ld
\n\t\t\tGender:%s \n\t\t\tGrade:%s \n\t\t\tSection:%s \n\t\t\tAge:%d
\n\t\t\tRegistrationnumber:%ld \n\t\t\tEmail:%s",

                  firstname, lastname, address, phonenumber, gender, grade, section, age,
registrationNumber, email);

            printf("\n\n");

            results++;

        }

    }

    if (results == 0)

    {

        printf("\t\t\t");


        printf("User with this registration number doesnot exit.\n");

    }

    return 0;

}


int editUser()

{

    long int regis;

    printf("\t\t\t");


    printf("Enter the registration number.\n");

    printf("\t\t\t");
    scanf("%ld", &regis);


    FILE *fp;
    fp = fopen("address.txt", "r+");


    char firstname[20];
    char lastname[20];
    char address[40];
    long int phonenumber;
    char gender[20];
    char grade[40];
    char section[20];
    int age;
    long int registrationNumber;
    char email[100];
    int results = 0;


    while (fscanf(fp, "Firstname:%s Lastname:%s Address:%s Phonenumber:%ld Gender:%s
Grade:%s Section:%s Age:%d Registrationnumber:%ld Email:%s\n",
                  firstname, lastname, address, &phonenumber, gender, grade, section, &age,
&registrationNumber, email) != EOF)
    {



        if (regis == registrationNumber)
        {
            char fnameEdited[20];


            char lnameEdited[20];
            char addressEdited[40];
            long int phonenumberE;
            char genderE[20];
            char gradeE[40];
            char sectionE[20];
            int ageE;

            char emailE[100];
            printf("\t\t\t");
            printf("Enter the firstname.\n");
            printf("\t\t\t");
            scanf("%s", fnameEdited);
            printf("\t\t\t");
            printf("Enter the lastname.\n");
            printf("\t\t\t");
            scanf("%s", lnameEdited);
            printf("\t\t\t");
            printf("Enter the address.\n");
            printf("\t\t\t");
            scanf("%s", addressEdited);
            printf("\t\t\t");
            printf("Enter the phonenumber.\n");
            printf("\t\t\t");
            scanf("%ld", &phonenumberE);
            printf("\t\t\t");
            printf("Enter the gender (Male/Female/Other).\n");
            printf("\t\t\t");
            scanf("%s", genderE);
```

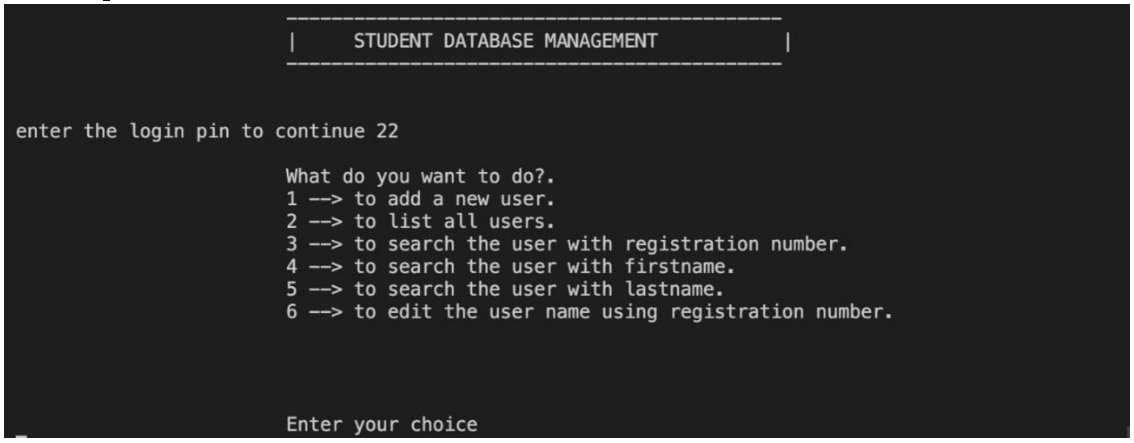
```
printf("\t\t\t");
printf("Enter the grade.\n");
printf("\t\t\t");
scanf("%s", gradeE);
printf("\t\t\t");
printf("Enter the section.\n");
printf("\t\t\t");
scanf("%s", sectionE);
printf("\t\t\t");
printf("Enter the age.\n");
printf("\t\t\t");
scanf("%d", &ageE);
printf("\t\t\t");
printf("Enter the email.\n");
printf("\t\t\t");
scanf("%s", emailE);

strcpy(firstname, fnameEdited);
strcpy(lastname, lnameEdited);
strcpy(address, addressEdited);
strcpy(gender, genderE);
strcpy(grade, gradeE);
strcpy(section, sectionE);
age=ageE;
strcpy(email, emailE);

printf("\t\t");
printf("--------------------------");
```

```
            printf("\n\n");

            printf("\t\t");


            printf("Firstname:%s \n\t\t\tLastname:%s \n\t\t\tAddress:%s \n\t\t\tPhonenumber:%ld
\n\t\t\tGender:%s \n\t\t\tGrade:%s \n\t\t\tSection:%s \n\t\t\tAge:%d
\n\t\t\tRegistrationnumber:%ld \n\t\t\tEmail:%s",

                   firstname, lastname, address, phonenumber, gender, grade, section, age,
registrationNumber, email);

            printf("\n\n");

            results++;

        }

    }

printf("%d", results);

    if (results == 0)

    {

        printf("\t\t\t");

        printf("User with this registration number doesnot exit.\n");

    }

    else

    {printf("Successfully edited!");}


    return 0;

}




int main()

{
    int choice, filteredchoice, pin;
start:
    printf("\n");
    printf("\t\t\t");
    printf("--------------------------------------------\n");
    printf("\t\t\t");
    printf("|     STUDENT DATABASE MANAGEMENT           |");
    printf("\n");
    printf("\t\t\t");
    printf("--------------------------------------------\n");
    printf("\n\n");

    printf("enter the login pin to continue\t");
    scanf("%d", &pin);
    if (pin == 22)
    {

    label:
        printf("\n");
        printf("\t\t\t");
        printf("What do you want to do?.\n");
        printf("\t\t\t");
        printf("1 --> to add a new user.\n");
        printf("\t\t\t");

        printf("2 --> to list all users.\n");
        printf("\t\t\t");
printf("3 --> to search the user with registration number.\n");
printf("\t\t\t");

printf("4 --> to search the user with firstname.\n");
printf("\t\t\t");

printf("5 --> to search the user with lastname.\n");

printf("\t\t\t");

printf("6 --> to edit the user name using registration number.\n");

printf("\n\n\n\n");

printf("\t\t\t");

printf("Enter your choice\n");
scanf("%d", &choice);

if (choice == 1)

{
    addUser();
}

else if (choice == 2)
{
    listUser();
}
else if (choice == 3)
{
   searchWithReg();
}
else if (choice == 4)
{
   searchWithFirstname();
}
else if (choice == 5)
{
   searchWithLastname();
}
else if (choice == 6)
{
   editUser();
}
else
{
   printf("\t\t\t");
   printf("Please choose the correct option.(You can only choose between 1,2 and 3.)\n");
   goto label;
}

printf("\t\t\t");
printf("Press any key to continue interacting or 0 to quit.\n");
printf("\t\t\t");
scanf("%d", &cont);
```

```
        if (cont == 0)
        {
            return 0;
        }
        else
        {
            goto label;
        }
    }
    else
        printf("Wrong pin");
}
```

7. IMPLEMENTATION AND OUTPUT

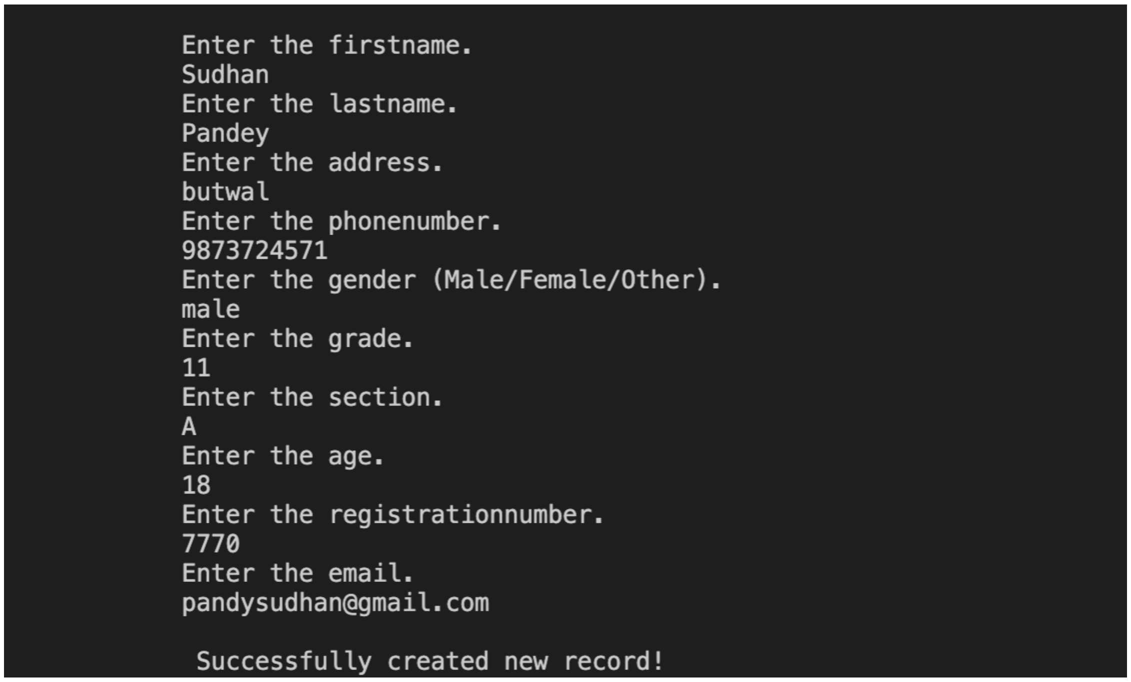
The output screenshot is shown as follows:



```
                        -------------------------------------------
                        |     STUDENT DATABASE MANAGEMENT          |
                        -------------------------------------------


enter the login pin to continue 22

                        What do you want to do?.
                        1 --> to add a new user.
                        2 --> to list all users.
                        3 --> to search the user with registration number.
                        4 --> to search the user with firstname.
                        5 --> to search the user with lastname.
                        6 --> to edit the user name using registration number.



                        Enter your choice
```

Figure 1.1 (main page)

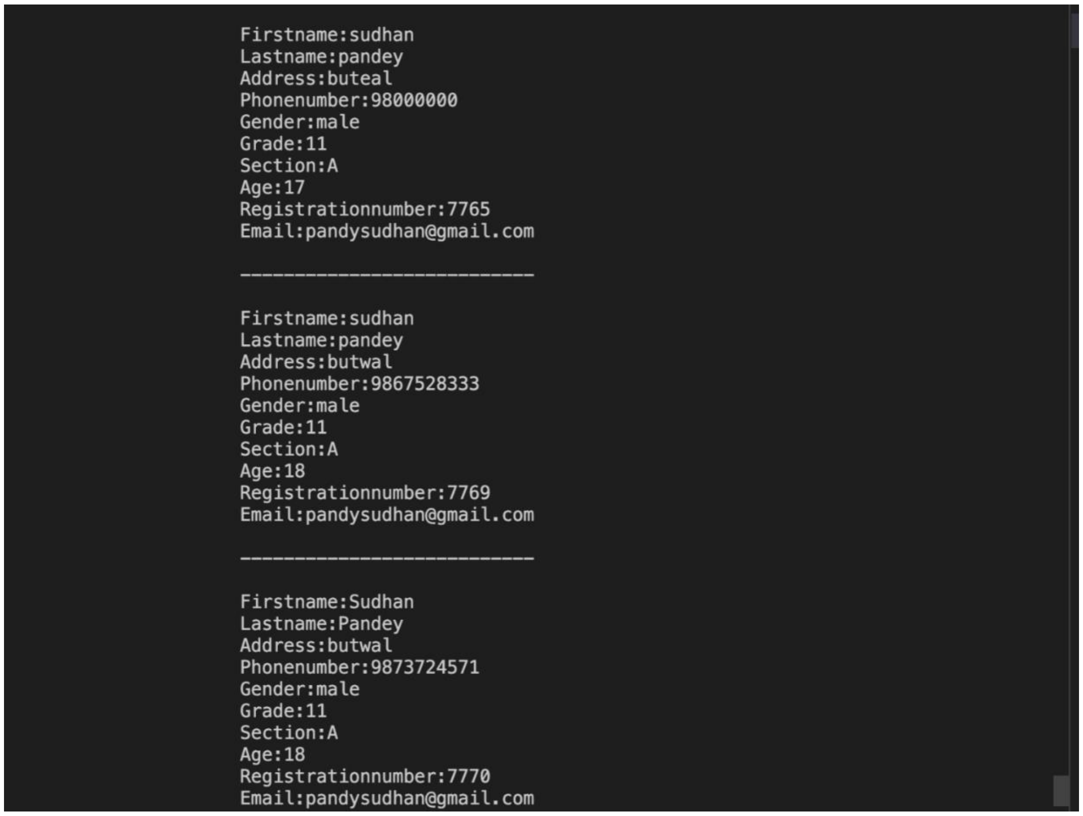
In the above screenshot, user has to enter the pin to continue further. Here this is the main page where user is given the choice as per ones wish.



```
Enter the firstname.
Sudhan
Enter the lastname.
Pandey
Enter the address.
butwal
Enter the phonenumber.
9873724571
Enter the gender (Male/Female/Other).
male
Enter the grade.
11
Enter the section.
A
Enter the age.
18
Enter the registrationnumber.
7770
Enter the email.
pandysudhan@gmail.com

 Successfully created new record!
```

Figure 1.2 (add user data)

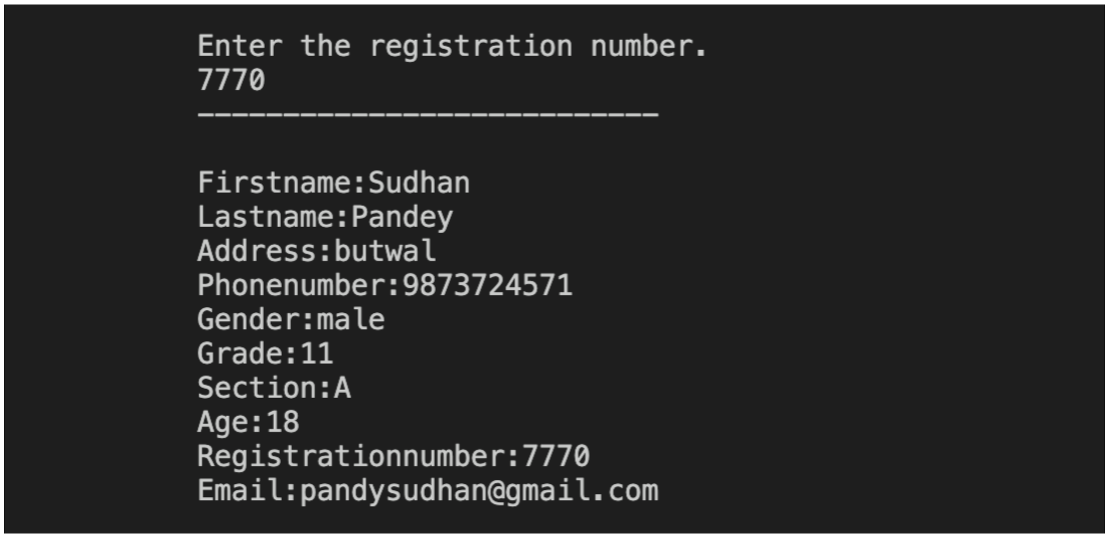
Here, user can add respective information about student.



```
Firstname:sudhan
Lastname:pandey
Address:buteal
Phonenumber:98000000
Gender:male
Grade:11
Section:A
Age:17
Registrationnumber:7765
Email:pandysudhan@gmail.com

--------------------------

Firstname:sudhan
Lastname:pandey
Address:butwal
Phonenumber:9867528333
Gender:male
Grade:11
Section:A
Age:18
Registrationnumber:7769
Email:pandysudhan@gmail.com

--------------------------

Firstname:Sudhan
Lastname:Pandey
Address:butwal
Phonenumber:9873724571
Gender:male
Grade:11
Section:A
Age:18
Registrationnumber:7770
Email:pandysudhan@gmail.com
```

Figure 1.3 (list user data)

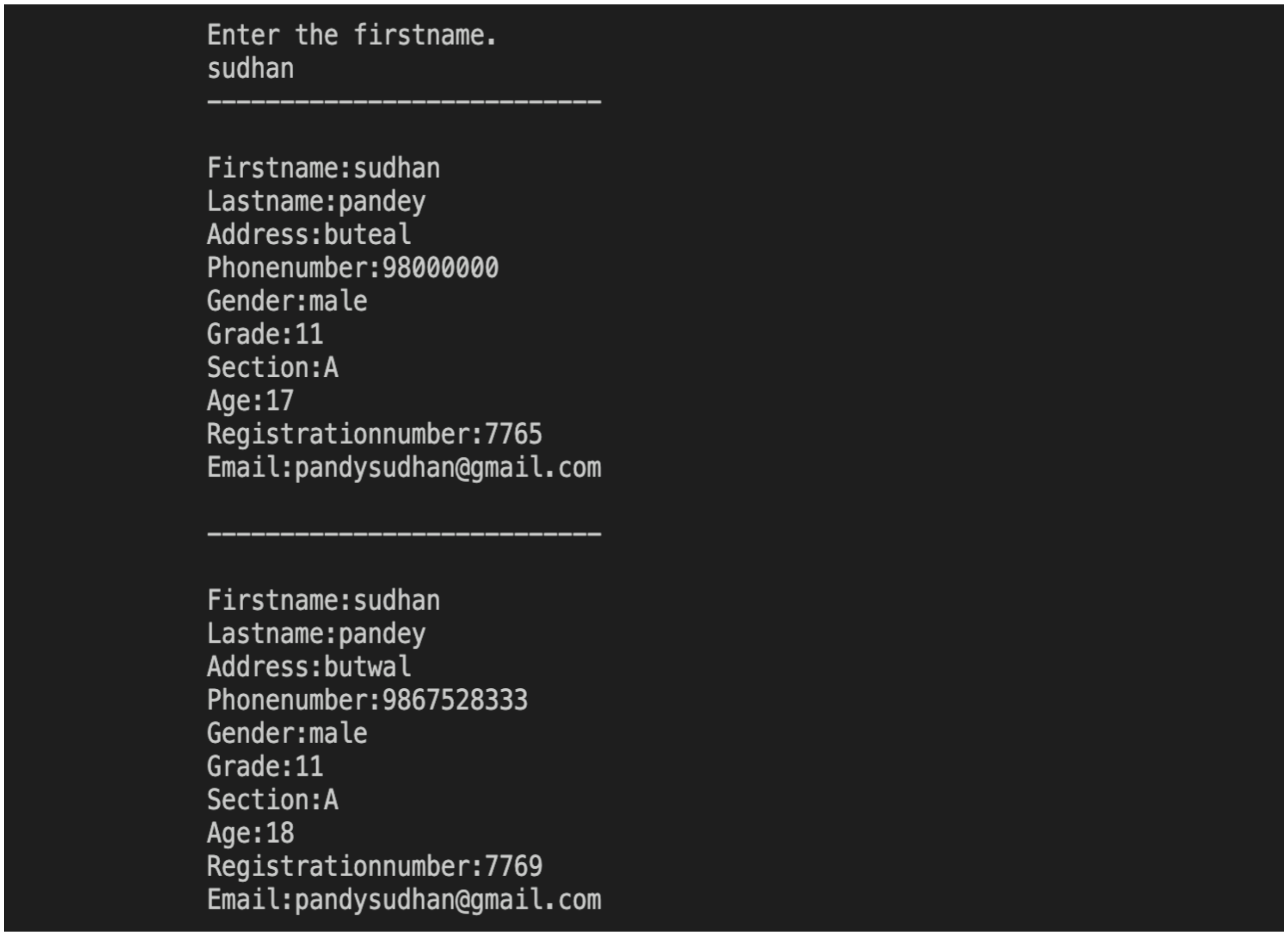
User can use this choice to list all the stored data of student previously recorded in formatted manner.



```
Enter the registration number.
7770
---------------------------

Firstname:Sudhan
Lastname:Pandey
Address:butwal
Phonenumber:9873724571
Gender:male
Grade:11
Section:A
Age:18
Registrationnumber:7770
Email:pandysudhan@gmail.com
```

Figure 1.4 (search by registration number)

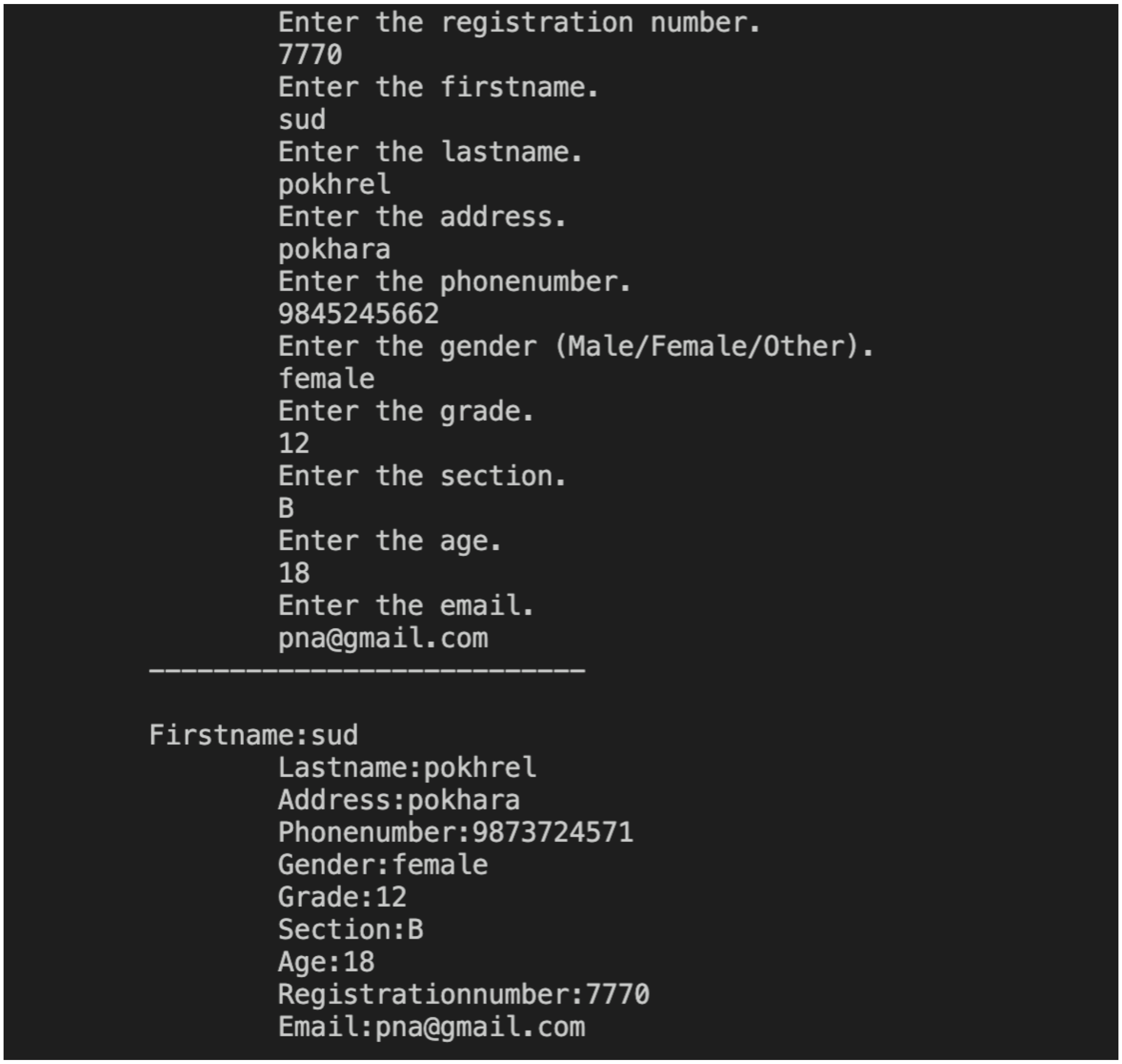
In above figure 1.4, user can search the desired student by entering registration number.



```
Enter the firstname.
sudhan
---------------------------

Firstname:sudhan
Lastname:pandey
Address:buteal
Phonenumber:98000000
Gender:male
Grade:11
Section:A
Age:17
Registrationnumber:7765
Email:pandysudhan@gmail.com

---------------------------

Firstname:sudhan
Lastname:pandey
Address:butwal
Phonenumber:9867528333
Gender:male
Grade:11
Section:A
Age:18
Registrationnumber:7769
Email:pandysudhan@gmail.com
```

Figure 1.5 (search by name)

In the above figure 1.5, user can search the information about a specific student either by help of first name or last name. user have two choice to either search from students first name or last name. The information about student is shown in a formatted manner.



```
          Enter the registration number.
          7770
          Enter the firstname.
          sud
          Enter the lastname.
          pokhrel
          Enter the address.
          pokhara
          Enter the phonenumber.
          9845245662
          Enter the gender (Male/Female/Other).
          female
          Enter the grade.
          12
          Enter the section.
          B
          Enter the age.
          18
          Enter the email.
          pna@gmail.com
--------------------------

Firstname:sud
          Lastname:pokhrel
          Address:pokhara
          Phonenumber:9873724571
          Gender:female
          Grade:12
          Section:B
          Age:18
          Registrationnumber:7770
          Email:pna@gmail.com
```

Figure 1.6 (edit user)

Here, as shown user have option to edit the previously stored data about student. If there is misinformation entered previously, user have the option to edit the information.

7. DISUSSION AND CONLCUSION

The program covers almost all the topic that we studied during the course of our study. The program consists of main concept of file handling, control statement and structures. This program is bit different from usual as we didn’t use function statement hoping to make it easier to understand by any normal user. We have written our program in Code Blocks and Dev C++ as it is convenient and easier for programming.

In addition, we still want to emphasize that the program is not complete by itself. There is still a lot of room for improvement. More function can be added to program to make it dynamic. Graphics may be added to program to make it more attractive as well as interactive. Other variety of function can be added to make database system more diverse.

This is an open source program and therefore everybody is welcome to develop it. Future developers are very welcome to add their ideas to the program and improvise it. Hopefully, all the users of this program will find it useful and entertaining.

8. REFERENCES

i. Class presentation given by Ganesh sir

ii. Learning C by examples by Er.Krishna Kandel

iii. Programming with C by Bryon S.Gottfried

iv. Let us C by Yashavant P. Kanetkar

♦♦ ♦♦ ♦♦ ♦♦ ♦♦ The End ♦♦ ♦♦ ♦♦ ♦♦ ♦♦