ASSIGNMENT NO.8.

Aim :-   Department maintains a student information. The file contains roll number, name, division and address. Allow user to add, delete information of student. Display information of particular employee. If record of student does not exist an appropriate message is displayed. If it is, then the system displays the student

Objective:- To study the different data structure concepts to implement this program.

Theory:-

Input/output formatting

Writing to or reading from a file is similar to writing onto a terminal screen or reading from a keyboard. Differences are:

- File must be opened with an OPEN statement, in which the unit number and (optionally) the filename are given
- Subsequent writes (or reads) must refer to a known unit number (used for open)
- File should be closed at the end

File opening and closing

The syntax is:

```
OPEN([unit=]lunit,file='name' [,options])

CLOSE([unit=]lunit [,options])
```

For example:

```
OPEN(10, file='output.dat', status='new')

CLOSE(unit=10)
```

- The first parameter is the unit number and the keyword unit= can be omitted.
- The unit numbers 0,5 and 6 are predefined.
- 0 is output for standard (system) error messages

- o 5 is for standard (user) input
- o 6 is for standard (user) output
- o These units are opened by default and should not be re-opened nor closed by users

Some options for opening a file:

- o status: existence of the file ('old', 'new', 'replace', 'scratch', 'unknown')
- o position: offset, where to start writing ('append')
- o action: file operation mode ('write','read','readwrite')
- o form: text or binary file ('formatted', 'unformatted')
- o access: direct or sequential file access ('direct','sequential','stream')
- o iostat: error indicator, (output) integer (non zero only upon an error)
- o err: the label number to jump upon error
- o recl: record length, (input) integer for direct access files only. Be careful, it can be in bytes or words...

Algorithm:-

1. Take the count of number of students from the user

2. Make an array of object of the student class which stores the information of the students

3. Open a file by using the ofstream object

4. Take the information of the student from the user and write it to the file

5. User can perform 1.Search 2.Delete 3.Display operations

6. For Search

1.Input the Roll number to be searched

2. Open the file using Ifstream object in input mode

3. Read the contents of the file in an object sequentially and check it with the roll number to be searched if found Display found message and the details of the students

4. If not found continue till end of file

5. If eof is reached display the message Not found

7.For Delete

1.Input the roll number to be deleted

2. Open the Main file in input mode and a temporary file in output mode

3.Sequentially search through the main file and copy the contents to the temp file except the roll number to be deleted

4. Delete the contents of the Main file

5. Rename the temp file with the name of the main file

 8. For Display

1. Open the file in input mode and display the details of all the students sequentially

Program Code:-

```cpp
#include <iostream>

#include <fstream>

#include <cstring>

#include <iomanip>

#include<cstdlib>

#define max 50

using namespace std;

class Student

{
    char name[max];

    int rollNo;

    int year;

    int division;

    char address[50];

    friend class FileOperations;

    public:    Student()

            {
```

```cpp
                        strcpy(name,"");

                        rollNo=year=division=0;

                        strcpy(address,"");

            }

            Student(char name[max],int rollNo,int year,int division,char address[max])

            {

                        strcpy(this->address,address);

                        strcpy(this->name,name);

                        this->division=division;

                        this->rollNo=rollNo;

                        this->year=year;

            }

            int getRollNo()

            {

                        return rollNo;

            }

            void displayStudentData()

            {


        cout<<endl<<setw(3)<<rollNo<<setw(10)<<name<<setw(3)<<year<<setw(2)<<division<<setw(10)<<address;

            }

};

class FileOperations

{

        fstream file;
```

```cpp
public:FileOperations(char *name)

        {

                //strcpy(this->name,name);

                this->file.open(name,ios::in|ios::out|ios::ate|ios::binary);

        }

        void insertRecord(int rollNo,char name[max],int year,int division,char address[max])

        {

                Student s=Student(name,rollNo,year,division,address);

                file.seekp(0,ios::end);

                file.write((char*)&s,sizeof(Student));

                file.clear();

        }

        void displayAllRecords()

        {

                Student s;

                file.seekg(0,ios::beg);

                while(file.read((char *)&s,sizeof(Student)))

                {

                        s.displayStudentData();

                }

                file.clear();

        }

        void displayRecord(int rollNo)

        {
```

```cpp
        Student s;
        file.seekg(0,ios::beg);
        void *p;
        while(file.read((char *)&s,sizeof(Student)))
        {
                if(s.rollNo==rollNo)
                {
                        s.displayStudentData();
                        break;
                }
        }
        if(p==NULL)
                throw "Element not present";
        file.clear();
    }
    void deleteRecord(int rollNo)
    {
        ofstream newFile("new.txt",ios::binary);
        file.seekg(0,ios::beg);
        bool flag=false;
        Student s;
        while(file.read((char *)&s,sizeof(s)))
        {
                if(s.rollNo==rollNo)
                {
```

```
                        flag=true;

                        continue;

                }

                newFile.write((char *)&s,sizeof(s));

        }

        if(!flag)

        {

                cout<<"Element Not Present";

        }

        file.close();

        newFile.close();

        remove("student.txt");

        rename("new.txt","student.txt");

        file.open("student.txt",ios::in|ios::ate|ios::out|ios::binary);

    }

    ~FileOperations()

    {

        file.close();

        cout<<"Closing file..";

    }

};
int  main()
{
    ofstream newFile("student.txt",ios::app|ios::binary);
```
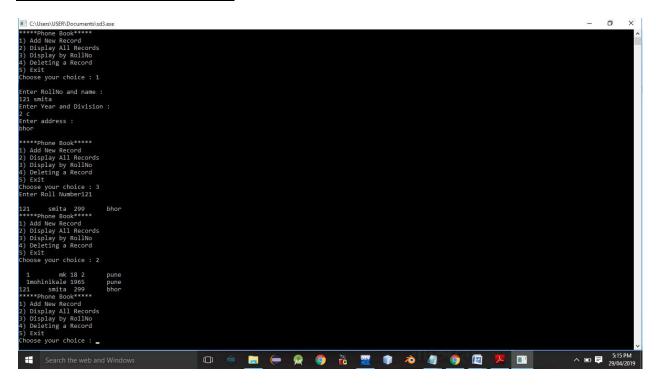
```cpp
        newFile.close();

        FileOperations file((char *)"student.txt");

    int rollNo,year,choice=0;

    char division;

    char name[max],address[max];

    while(choice!=5)

    {

        //clrscr();

        cout<<"\n*****Phone Book*****\n";

        cout<<"1) Add New Record\n";

        cout<<"2) Display All Records\n";

        cout<<"3) Display by RollNo\n";

        cout<<"4) Deleting a Record\n";

        cout<<"5) Exit\n";

        cout<<"Choose your choice : ";

        cin>>choice;

        switch(choice)

        {

            case 1 : //New Record

                        cout<<endl<<"Enter RollNo and name : \n";

                        cin>>rollNo>>name;

                        cout<<"Enter Year and Division : \n";

                        cin>>year>>division;

                        cout<<"Enter address : \n";

                        cin>>address;
```
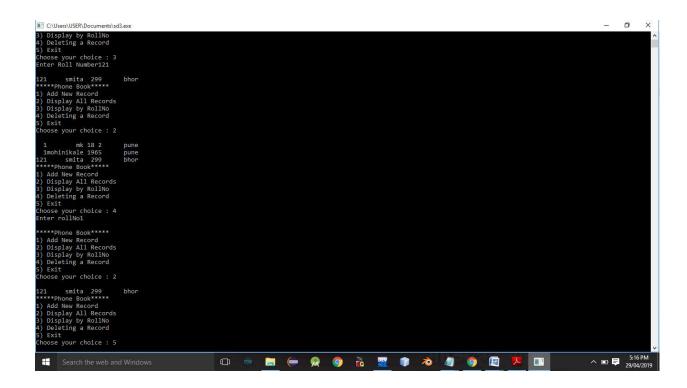
```cpp
                file.insertRecord(rollNo,name,year,division,address);

                break;

        case 2 :

                file.displayAllRecords();

                break;

        case 3 :

                cout<<"Enter Roll Number";

                cin>>rollNo;

                try

                {

                        file.displayRecord(rollNo);

                }

                catch(const char *str)

                {

                        cout<<str;

                }

                break;

        case 4:

                cout<<"Enter rollNo";

                cin>>rollNo;

                file.deleteRecord(rollNo);

                break;

        case 5 :break;

    }
```

```
    }

}
```

Output Screenshots:-

Conclusion:- Thus,this assignment implemented successfully.