Assignment 11

```
#include <iostream>

#include <fstream>
#include <cstring>
using namespace std;

class Student {
private:
    int roll;
    char name[50];
    char division[5];
    char address[100];

public:
    void input();
    void display();
    int getRoll() { return roll; }

    void addRecord(int n);
    void showAll();
    void searchRecord(int);
    void deleteRecord(int);
    void editRecord(int);
};

void Student::input() {
    cout << "\nEnter Roll No: ";
    cin >> roll;
    cin.ignore();

    cout << "Enter Name: ";
    cin.getline(name, 50);

    cout << "Enter Division: ";
    cin.getline(division, 5);

    cout << "Enter Address: ";
    cin.getline(address, 100);
}

void Student::display() {
    cout << "\nRoll: " << roll
         << "\nName: " << name
         << "\nDivision: " << division
         << "\nAddress: " << address << endl;
}

void Student::addRecord(int n) {
    ofstream out("STUDENT.DAT", ios::binary | ios::app);
    Student s;
```

```
    for (int i = 0; i < n; i++) {
        s.input();
        out.write(reinterpret_cast<char*>(&s), sizeof(s));
    }
    out.close();
}

void Student::showAll() {
    ifstream in("STUDENT.DAT", ios::binary);
    Student s;
    bool found = false;
    while (in.read(reinterpret_cast<char*>(&s), sizeof(s))) {
        found = true;
        s.display();
    }
    if (!found)
        cout << "\nNo records found.";
    in.close();
}

void Student::searchRecord(int r) {
    ifstream in("STUDENT.DAT", ios::binary);
    Student s;
    bool found = false;
    while (in.read(reinterpret_cast<char*>(&s), sizeof(s))) {
        if (s.getRoll() == r) {
            cout << "\nRecord found:";
            s.display();
            found = true;
            break;
        }
    }
    if (!found)
        cout << "\nRecord not found.";
    in.close();
}

void Student::deleteRecord(int r) {
    ifstream in("STUDENT.DAT", ios::binary);
    ofstream out("TEMP.DAT", ios::binary);
    Student s;
    bool found = false;
    while (in.read(reinterpret_cast<char*>(&s), sizeof(s))) {
        if (s.getRoll() != r)
            out.write(reinterpret_cast<char*>(&s), sizeof(s));
        else
            found = true;
    }
    in.close();
    out.close();
    remove("STUDENT.DAT");
    rename("TEMP.DAT", "STUDENT.DAT");
    if (found)
        cout << "\nRecord deleted successfully.";
    else
        cout << "\nRecord not found.";
}

void Student::editRecord(int r) {
```

```
    ifstream in("STUDENT.DAT", ios::binary);
    ofstream out("TEMP.DAT", ios::binary);
    Student s;
    bool found = false;
    while (in.read(reinterpret_cast<char*>(&s), sizeof(s))) {
        if (s.getRoll() == r) {
            cout << "\nEnter new details:\n";
            s.input();
            found = true;
        }
        out.write(reinterpret_cast<char*>(&s), sizeof(s));
    }
    in.close();
    out.close();
    remove("STUDENT.DAT");
    rename("TEMP.DAT", "STUDENT.DAT");
    if (found)
        cout << "\nRecord updated successfully.";
    else
        cout << "\nRecord not found.";
}

int main() {
    Student s;
    int choice, n, roll;

    do {
        cout << "\nMenu\n1. Add Student\n2. Display All\n3. Search\n4. Delete\n5. Edit\n6.
Exit\nChoice: ";
        cin >> choice;
        switch (choice) {
            case 1:
                cout << "Enter number of students to add: ";
                cin >> n;
                s.addRecord(n);
                break;
            case 2:
                s.showAll();
                break;
            case 3:
                cout << "Enter roll number to search: ";
                cin >> roll;
                s.searchRecord(roll);
                break;
            case 4:
                cout << "Enter roll number to delete: ";
                cin >> roll;
                s.deleteRecord(roll);
                break;
            case 5:
                cout << "Enter roll number to edit: ";
                cin >> roll;
                s.editRecord(roll);
                break;
            case 6:
                cout << "Exiting...\n";
                break;
            default:
                cout << "Invalid choice!";
```

```
        }
    } while (choice != 6);

    return 0;
}
```

Output:

```
Menu
1. Add Student
2. Display All
3. Search
4. Delete
5. Edit
6. Exit
Choice: 1
Enter number of students to add: 2

Enter Roll No: 101
Enter Name: Alice
Enter Division: A
Enter Address: 123 Apple Street

Enter Roll No: 102
Enter Name: Bob
Enter Division: B
Enter Address: 456 Banana Lane

Menu
Choice: 2

Roll: 101
Name: Alice
Division: A
Address: 123 Apple Street

Roll: 102
Name: Bob
Division: B
Address: 456 Banana Lane

Menu
Choice: 3
Enter roll number to search: 101

Record found:
Roll: 101
Name: Alice
Division: A
Address: 123 Apple Street

Menu
Choice: 4
Enter roll number to delete: 102
Record deleted successfully.

Menu
Choice: 2

Roll: 101
Name: Alice
Division: A
Address: 123 Apple Street

Menu
Choice: 5
Enter roll number to edit: 101

Enter new details:
Enter Roll No: 101
Enter Name: Alicia
Enter Division: A
Enter Address: 789 Cherry Blvd
Record updated successfully.
```