Ganpat University

Faculty of Engineering & Technology

Computer Science & Engineering

Name:- ***Dwij Vatsal Desai***

Sem:- ***2***

Sub: - ***ESFP-II***

Enrollment No.:- ***23162121027***

Prac:- ***11***

Practical 11

Definition:
Complete the code for the object assigned to you to satisfy following specifications.
1. For the solving purpose of given topic practical, you need to create minimum two classes, rest as per your requirement.
2. Declare minimum six function with the same name and same signature prototype. Function should be perform the following task like (read data record, display data record, delete data record, update data record, search data record and display all data record in ascending or in descending order. Perform this point by using pure virtual function and abstract class.
3. Implement the concept of virtual function also in practical.
4. Minimum 1 constructor method should be available in the program, rest as per your requirement.
5. You must use access specifier for data member and member function declaration in program.
6. Wherever is required to use character data member in class, instead of that use compulsorily string data member.
7. Take minimum 5 data record from the user and display according to the choice of user category wise. (Minimum six different options should be there for displaying information, and if you want to add more choice option as per your program requirement, you can add it.).
8. Use all possibility filter method from stored record information:
9. After all functionality execution, you need to call the destructor function.

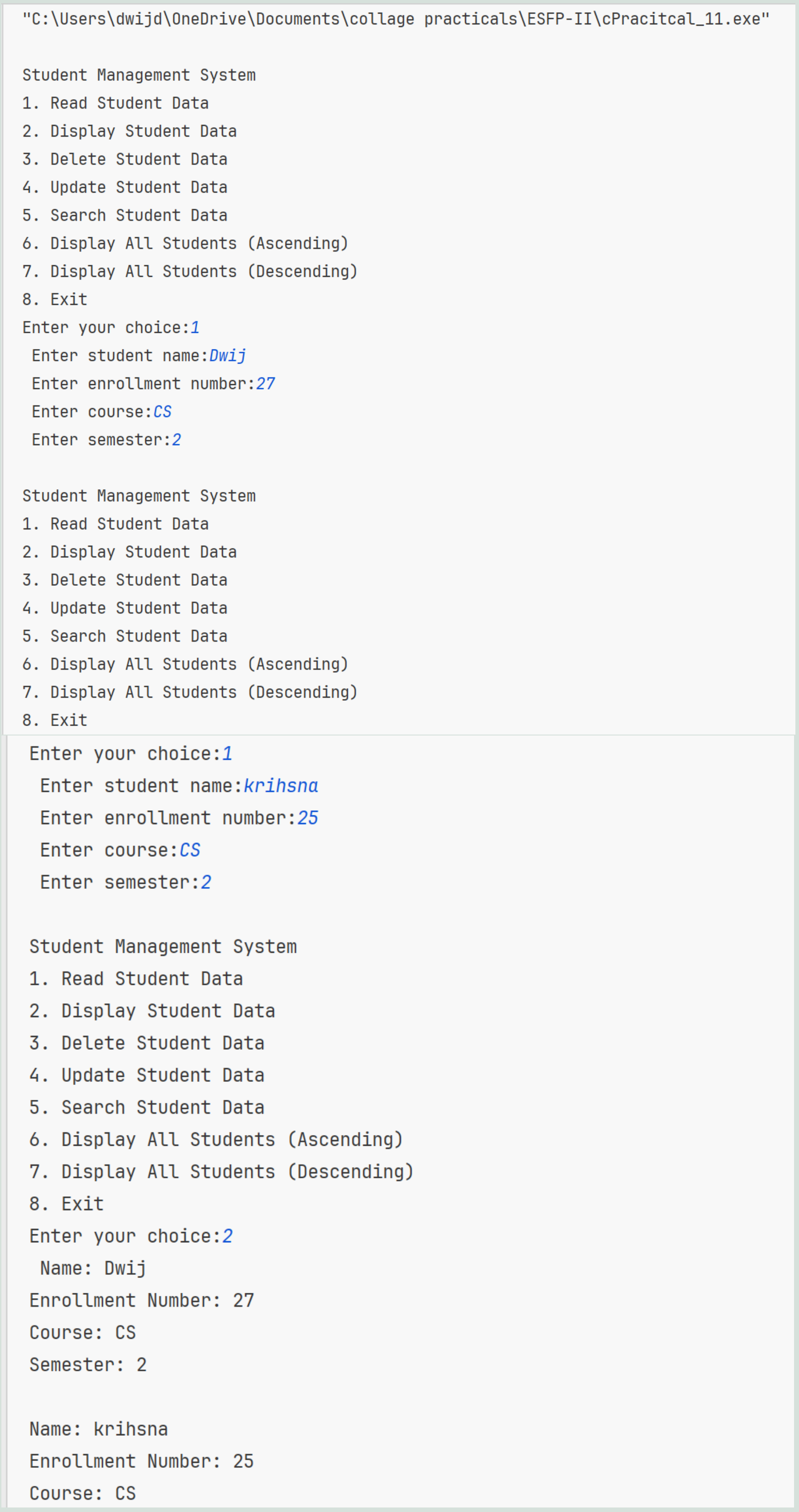
Code:-

```
#include <iostream>
#include <string>
using namespace std;

const int MAX_STUDENTS = 100;

class Student {
```

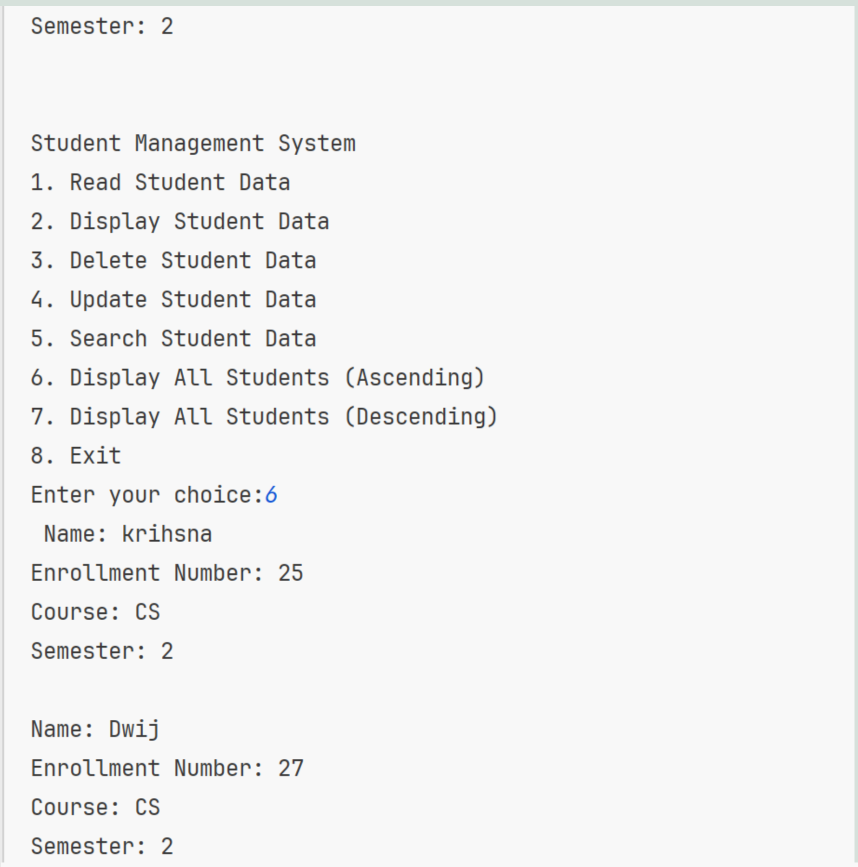
```cpp
public:
    string Name;
    int enroll_no;
    string course;
    int sem;

    virtual void readData() {
        cout << "Enter student name: ";
        getline(cin, Name);
        cout << "Enter enrollment number: ";
        cin >> enroll_no;
        cin.ignore();
        cout << "Enter course: ";
        getline(cin, course);
        cout << "Enter semester: ";
        cin >> sem;
    }

    virtual void displayData() const {
        cout << "Name: " << Name << endl;
        cout << "Enrollment Number: " << enroll_no << endl;
        cout << "Course: " << course << endl;
        cout << "Semester: " << sem << endl;
    }

    virtual void deleteData() {}
    virtual void updateData() {}
    virtual void searchData(int enroll_no) const {}
    virtual void displayAllData(bool ascending) const {}
};

class StudentManager : public Student {
private:
    Student students[MAX_STUDENTS];
    int numStudents;

public:
    StudentManager() : numStudents(0) {}

    void readData() override {
        if (numStudents < MAX_STUDENTS) {
            Student newStudent;
            newStudent.Student::readData();
            students[numStudents] = newStudent;
            numStudents++;
        } else {
            cout << "Maximum number of students reached." << endl;
        }
    }

    void displayData() const override {
        for (int i = 0; i < numStudents; i++) {
            students[i].displayData();
            cout << endl;
        }
    }

    void deleteData() override {
        int studentEnroll;
        cout << "Enter student enrollment number to delete: ";
        cin >> studentEnroll;

        for (int i = 0; i < numStudents; i++) {
            if (students[i].enroll_no == studentEnroll) {
                for (int j = i; j < numStudents - 1; j++) {
                    students[j] = students[j + 1];
                }
                numStudents--;
                cout << "Student deleted successfully." << endl;
                return;
            }
        }

        cout << "Student not found." << endl;
    }

    void updateData() override {
        int studentEnroll;
        cout << "Enter student enrollment number to update: ";
        cin >> studentEnroll;

        for (int i = 0; i < numStudents; i++) {
            if (students[i].enroll_no == studentEnroll) {
                Student updatedStudent;
                updatedStudent.readData();
                students[i] = updatedStudent;
                cout << "Student updated successfully." << endl;
                return;
            }
        }

        cout << "Student not found." << endl;
    }

    void searchData(int searchEnroll) const override {
        for (int i = 0; i < numStudents; i++) {
            if (students[i].enroll_no == searchEnroll) {
                students[i].displayData();
                return;
            }
        }

        cout << "Student not found." << endl;
    }

    void displayAllData(bool ascending) const override {
        Student sortedStudents[MAX_STUDENTS];
        for (int i = 0; i < numStudents; i++) {
            sortedStudents[i] = students[i];
        }

        for (int i = 0; i < numStudents - 1; i++) {
            for (int j = 0; j < numStudents - i - 1; j++) {
                if (ascending) {
                    if (sortedStudents[j].enroll_no > sortedStudents[j +
1].enroll_no) {
                        Student temp = sortedStudents[j];
                        sortedStudents[j] = sortedStudents[j + 1];
                        sortedStudents[j + 1] = temp;
                    }
                } else {
                    if (sortedStudents[j].enroll_no < sortedStudents[j +
1].enroll_no) {
                        Student temp = sortedStudents[j];
                        sortedStudents[j] = sortedStudents[j + 1];
                        sortedStudents[j + 1] = temp;
                    }
                }
            }
        }

        for (int i = 0; i < numStudents; i++) {
            sortedStudents[i].displayData();
            cout << endl;
        }
    }
```

```cpp
};

int main() {
    StudentManager manager;

    int choice;
    do {
        cout << "\nStudent Management System\n";
        cout << "1. Read Student Data\n";
        cout << "2. Display Student Data\n";
        cout << "3. Delete Student Data\n";
        cout << "4. Update Student Data\n";
        cout << "5. Search Student Data\n";
        cout << "6. Display All Students (Ascending)\n";
        cout << "7. Display All Students (Descending)\n";
        cout << "8. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;
        cin.ignore();

        switch (choice) {
            case 1:
                manager.readData();
                break;
            case 2:
                manager.displayData();
                break;
            case 3:
                manager.deleteData();
                break;
            case 4:
                manager.updateData();
                break;
            case 5:
                {
                    int studentEnroll;
                    cout << "Enter student enrollment number to search: ";
                    cin >> studentEnroll;
                    manager.searchData(studentEnroll);
                    break;
                }
            case 6:
                manager.displayAllData(true);
                break;
            case 7:
                manager.displayAllData(false);
                break;
            case 8:
                cout << "Exiting program..." << endl;
                break;
            default:
                cout << "Invalid choice. Please try again." << endl;
        }
    } while (choice != 8);

    return 0;
}
```

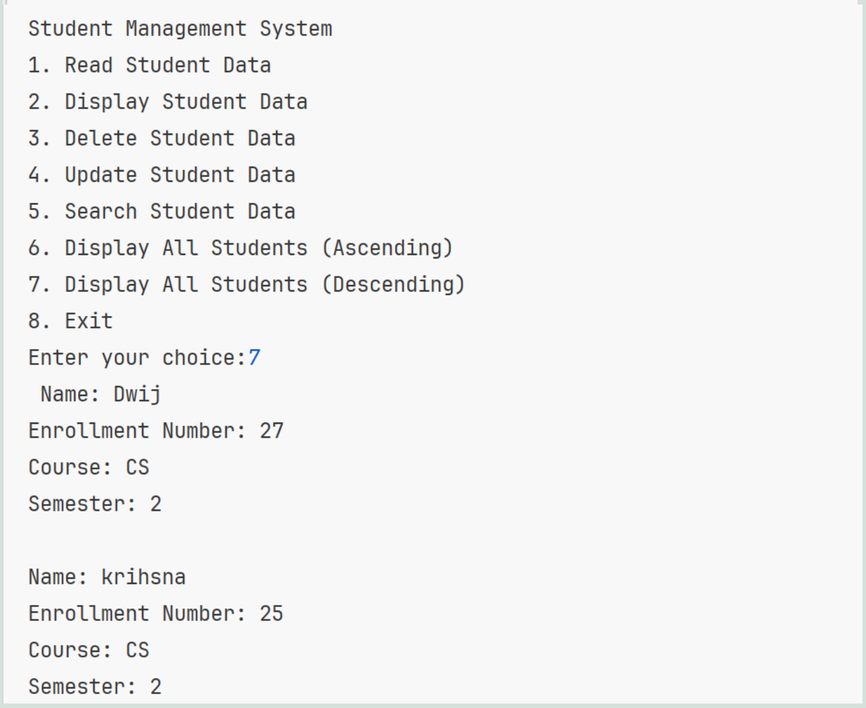
Output:-



```
"C:\Users\dwijd\OneDrive\Documents\collage practicals\ESFP-II\cPracitcal_11.exe"

Student Management System
1. Read Student Data
2. Display Student Data
3. Delete Student Data
4. Update Student Data
5. Search Student Data
6. Display All Students (Ascending)
7. Display All Students (Descending)
8. Exit
Enter your choice:1
 Enter student name:Dwij
 Enter enrollment number:27
 Enter course:CS
 Enter semester:2

Student Management System
1. Read Student Data
2. Display Student Data
3. Delete Student Data
4. Update Student Data
5. Search Student Data
6. Display All Students (Ascending)
7. Display All Students (Descending)
8. Exit
Enter your choice:1
 Enter student name:krihsna
 Enter enrollment number:25
 Enter course:CS
 Enter semester:2

Student Management System
1. Read Student Data
2. Display Student Data
3. Delete Student Data
4. Update Student Data
5. Search Student Data
6. Display All Students (Ascending)
7. Display All Students (Descending)
8. Exit
Enter your choice:2
 Name: Dwij
Enrollment Number: 27
Course: CS
Semester: 2

Name: krihsna
Enrollment Number: 25
Course: CS
```



```
Semester: 2


Student Management System
1. Read Student Data
2. Display Student Data
3. Delete Student Data
4. Update Student Data
5. Search Student Data
6. Display All Students (Ascending)
7. Display All Students (Descending)
8. Exit
Enter your choice:6
 Name: krihsna
Enrollment Number: 25
Course: CS
Semester: 2

Name: Dwij
Enrollment Number: 27
Course: CS
Semester: 2
```



```
Student Management System
1. Read Student Data
2. Display Student Data
3. Delete Student Data
4. Update Student Data
5. Search Student Data
6. Display All Students (Ascending)
7. Display All Students (Descending)
8. Exit
Enter your choice:7
 Name: Dwij
Enrollment Number: 27
Course: CS
Semester: 2

Name: krihsna
Enrollment Number: 25
Course: CS
Semester: 2
```