~\OneDrive\Desktop\focp-II\Assignment-II.cpp

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

class Student {
   private:
        int rno;
        float cgpa;
        string name;
        string courses[10];
        int courseCount;

    public:

    Student() {
        name = "Unknown";
        rno = 0;
        cgpa = 0.0;
        courseCount = 0;
        cout << "Default constructor called.\n";
    }


    Student(string n, int r, float c) {
        name = n;
        rno = r;
        setCGPA(c);
        courseCount = 0;
    }


    Student(const Student& other) {
        name = other.name;
        rno = other.rno;
        cgpa = other.cgpa;
        courseCount = other.courseCount;
        for (int i = 0; i < courseCount; i++) {
            courses[i] = other.courses[i];
        }
    }

    // Destructor
    ~Student() {
        cout << "Destructor called for: " << name << endl;
    }


    void getInfo() {
        cout << "Enter name: ";
```

```
        cin >> name;
        cout << "Enter roll number: ";
        cin >> rno;
        cout << "Enter CGPA: ";
        float tempCgpa;
        cin >> tempCgpa;
        setCGPA(tempCgpa);
    }


    void setCGPA(float c) {
        if (c >= 0.0 && c <= 10.0)
            cgpa = c;
        else {
            cout << "Invalid CGPA. Must be between 0.0 and 10.0. Setting CGPA to 0.\n";
            cgpa = 0.0;
        }
    }


    void inputCourses() {
        int n;
        cout << "How many courses (max 10)? ";
        cin >> n;
        if (n > 10) n = 10;
        for (int i = 0; i < n; i++) {
            cout << "Enter course " << i + 1 << ": ";
            cin >> courses[i];
        }
        courseCount = n;
    }


    void addCourses() {
        int choice;
        cout << "Do you want to add more courses? (1 = Yes, 0 = No): ";
        cin >> choice;

        if (choice == 1) {
            int extra;
            cout << "How many courses to add? ";
            cin >> extra;
            if (courseCount + extra > 10) {
                cout << "You can only have up to 10 courses total.\n";
                extra = 10 - courseCount;
            }

            for (int i = 0; i < extra; i++) {
                cout << "Enter additional course " << i + 1 << ": ";
                cin >> courses[courseCount];
```

```
                courseCount++;
            }
        }
    }


    void display() const {
        cout << "\nStudent Name: " << name;
        cout << "\nRoll Number: " << rno;
        cout << "\nCGPA: " << cgpa;
        cout << "\nCourses: ";
        for (int i = 0; i < courseCount; i++) {
            cout << courses[i] << " ";
        }
        cout << "\n";
    }

    int getRollNumber() const {
        return rno;
    }
};
class StudentManagementSystem {
    private:
        Student students[100];
        int studentCount;

    public:
        StudentManagementSystem() {
            studentCount = 0;
        }

        void addStudent() {
            if (studentCount >= 100) {
                cout << "Maximum number of students reached.\n";
                return;
            }

            Student s;
            s.getInfo();
            s.inputCourses();
            s.addCourses();
            students[studentCount] = s;
            studentCount++;

            cout << "Student added successfully!\n";
        }

    void searchStudent(int rollNo) const {
            for (int i = 0; i < studentCount; i++) {
                if (students[i].getRollNumber() == rollNo) {
```

```
                    cout << "Student found:\n";
                    students[i].display();
                    return;
                }
            }
        cout << "Student with roll number " << rollNo << " not found.\n";
    }

    void displayAllStudents() const {
            if (studentCount == 0) {
                cout << "No students in the system.\n";
                return;
            }

            for (int i = 0; i < studentCount; i++) {
                students[i].display();
            }
    }
};
int main() {
    StudentManagementSystem sms;

    int choice;
    do {
        cout << "\n----- Student Management Menu -----\n";
        cout << "1. Add Student\n";
        cout << "2. Search Student by Roll Number\n";
        cout << "3. Display All Students\n";
        cout << "0. Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;

        switch (choice) {
            case 1:
                sms.addStudent();
                break;
            case 2: {
                int rollNo;
                cout << "Enter roll number to search: ";
                cin >> rollNo;
                sms.searchStudent(rollNo);
                break;
            }
            case 3:
                sms.displayAllStudents();
                break;
            case 0:
                cout << "Exiting...\n";
                break;
            default:
```

```
199                 cout << "Invalid choice.\n";
200             }
201         } while (choice != 0);
202 
203         return 0;
204     }
205 
206 
207 
```