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

struct Student {
    int rollNo;
    string name;
    float sgpa;
};

void bubbleSortByRollNo(Student students[], int n) {
    for (int i = 0; i < n - 1; i++) {
        for (int j = 0; j < n - i - 1; j++) {
            if (students[j].rollNo > students[j + 1].rollNo) {
                swap(students[j], students[j + 1]);
            }
        }
    }
}

void insertionSortByName(Student students[], int n) {
    for (int i = 1; i < n; i++) {
        Student key = students[i];
        int j = i - 1;
        while (j >= 0 && students[j].name > key.name) {
            students[j + 1] = students[j];
            j--;
        }
        students[j + 1] = key;
    }
}

int partition(Student students[], int low, int high) {
    float pivot = students[high].sgpa;
    int i = low - 1;
    for (int j = low; j < high; j++) {
        if (students[j].sgpa > pivot) {
            i++;
            swap(students[i], students[j]);
        }
    }
    swap(students[i + 1], students[high]);
    return i + 1;
}

void quickSortBySGPA(Student students[], int low, int high) {
    if (low < high) {
        int pi = partition(students, low, high);
        quickSortBySGPA(students, low, pi - 1);
        quickSortBySGPA(students, pi + 1, high);
    }
}

int binarySearchByName(Student students[], int n, const string& name) {
```

```
    int left = 0, right = n - 1;
    while (left <= right) {
        int mid = left + (right - left) / 2;
        if (students[mid].name == name) {
            return mid;
        } else if (students[mid].name < name) {
            left = mid + 1;
        } else {
            right = mid - 1;
        }
    }
    return -1;
}

void searchBySGPA(Student students[], int n, float sgpa) {
    bool found = false;
    for (int i = 0; i < n; i++) {
        if (students[i].sgpa == sgpa) {
            cout << "Roll No: " << students[i].rollNo << ", Name: " <<
students[i].name << ", SGPA: " << students[i].sgpa << endl;
            found = true;
        }
    }
    if (!found) {
        cout << "No students found with SGPA " << sgpa << endl;
    }
}

void printStudents(Student students[], int n) {
    for (int i = 0; i < n; i++) {
        cout << "Roll No: " << students[i].rollNo << ", Name: " <<
students[i].name << ", SGPA: " << students[i].sgpa << endl;
    }
}

int main() {
    Student students[] = {
        {1, "Alice", 9.2}, {2, "Bob", 8.5}, {3, "Charlie", 7.8}, {4,
"David", 8.5}, {5, "Eve", 9.7},
        {6, "Frank", 8.0}, {7, "Grace", 7.2}, {8, "Hannah", 9.1}, {9,
"Ivy", 8.9}, {10, "Jack", 7.5},
        {11, "Kathy", 8.4}, {12, "Leo", 9.0}, {13, "Mona", 7.3}, {14,
"Nina", 8.6}, {15, "Oscar", 9.4}
    };
    int n = sizeof(students) / sizeof(students[0]);

    cout << "Original List:\n";
    printStudents(students, n);

    bubbleSortByRollNo(students, n);
    cout << "\nSorted by Roll Numbers:\n";
    printStudents(students, n);

    insertionSortByName(students, n);
```

```
    cout << "\nSorted by Names:\n";
    printStudents(students, n);

    quickSortBySGPA(students, 0, n - 1);
    cout << "\nTop 10 Students by SGPA:\n";
    for (int i = 0; i < 10 && i < n; i++) {
        cout << "Roll No: " << students[i].rollNo << ", Name: " <<
students[i].name << ", SGPA: " << students[i].sgpa << endl;
    }

    float sgpaToSearch;
    cout << "\nEnter SGPA to search: ";
    cin >> sgpaToSearch;
    searchBySGPA(students, n, sgpaToSearch);

    string nameToSearch;
    cout << "\nEnter Name to search: ";
    cin >> nameToSearch;
    int index = binarySearchByName(students, n, nameToSearch);
    if (index != -1) {
        cout << "Student found: Roll No: " << students[index].rollNo <<
", Name: " << students[index].name << ", SGPA: " << students[index].sgpa
<< endl;
    } else {
        cout << "Student with name " << nameToSearch << " not found." <<
endl;
    }

    return 0;
}
```