```
//Name: 1. Searching and Sorting
#include <iostream>
 #include <string>
 #include <algorithm>
 using namespace std;
 const int MAX_STUDENTS = 15;
 struct Student {
   int rollNumber;
   string name;
   double sgpa;
 };
 void bubbleSortRollNumbers(Student students[], int n) {
   for (int i = 0; i < n - 1; i++) {
      for (int j = 0; j < n - i - 1; j++) {
         if (students[j].rollNumber > students[j + 1].rollNumber) {
            swap(students[j], students[j + 1]);
         }
      }
   }
 }
 int partition(Student students[], int low, int high) {
   double 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 quickSortSGPA(Student students[], int low, int high) {
    if (low < high) {
        int pi = partition(students, low, high);
        quickSortSGPA(students, low, pi - 1);
        quickSortSGPA(students, pi + 1, high);
    }
}
int linearSearchSGPA(const Student students[], int n, double targetSGPA) {
    for (int i = 0; i < n; i++) {
        if (students[i].sgpa == targetSGPA) {
            return i;
        }
    }
    return -1;
}
int binarySearchName(const Student students[], int n, const string& targetName)
{
```

```
    int low = 0, high = n - 1;
    while (low <= high) {
        int mid = low + (high - low) / 2;
        if (students[mid].name == targetName) {
            return mid;
        } else if (students[mid].name < targetName) {
            low = mid + 1;
        } else {
            high = mid - 1;
        }
    }
    return -1;
}
void displayTop10SGPA(Student students[], int n) {
    quickSortSGPA(students, 0, n - 1);
    cout << "\nTop 10 Students by SGPA:\n";
    for (int i = 0; i < 10 && i < n; i++) {
        cout << students[i].rollNumber << " " << students[i].name << " " <<
students[i].sgpa << endl;
    }
}
int main() {
    Student students[MAX_STUDENTS] = {
        {1, "Aarav", 8.4},
```

```
    {2, "Vivaan", 7.8},
    {3, "Aditya", 9.2},
    {4, "Reyansh", 6.5},
    {5, "Vihaan", 9.5},
    {6, "Kabir", 8.1},
    {7, "Sai", 7.0},
    {8, "Anaya", 8.9},
    {9, "Saanvi", 9.0},
    {10, "Riya", 6.8},
    {11, "Kavya", 9.3},
    {12, "Nisha", 5.7},
    {13, "Mira", 8.7},
    {14, "Nisha", 8.0},
    {15, "Krishna", 9.1}
};
int n = MAX_STUDENTS;

bubbleSortRollNumbers(students, n);

sort(students, students + n, [](const Student& a, const Student& b) {
    return a.name < b.name;
});
int choice;
do {
```

```
cout << "\nChoose an option:\n";
cout << "1. Search by SGPA\n";
cout << "2. Search by Name\n";
cout << "3. Display Top 10 SGPA\n";
cout << "4. Exit\n";
cout << "Enter your choice: ";
cin >> choice;
switch (choice) {
   case 1: {
      double targetSGPA;
      cout << "Enter SGPA to search: ";
      cin >> targetSGPA;
      int indexSGPA = linearSearchSGPA(students, n, targetSGPA);
      if (indexSGPA != -1) {
         cout << "Student with SGPA " << targetSGPA << ": "
             << students[indexSGPA].name << endl;
      } else {
         cout << "No student found with SGPA " << targetSGPA << endl;
      }
      break;
   }
   case 2: {
      string targetName;
      cout << "Enter name to search: "
```

```
            cin.ignore();
            getline(cin, targetName);
            int indexName = binarySearchName(students, n, targetName);
            if (indexName != -1) {
                cout << "Student found by name " << targetName << ": Roll
Number: "
                     << students[indexName].rollNumber << ", SGPA: "
                     << students[indexName].sgpa << endl;
            } else {
                cout << "No student found with name " << targetName << endl;
            }
            break;
        }
        case 3:
            displayTop10SGPA(students, n);
            break;
        case 4:
            cout << "Exiting the program." << endl;
            break;
        default:
            cout << "Invalid choice. Please try again." << endl;
    }
  } while (choice != 4);
  return 0;
```

```
}
```