#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

// Define the structure to hold student data

struct Student {

int rollNo;

string name;

float SGPA;

};

// Function to display student details

void displayStudent(const Student& s) {

cout << "Roll No: " << s.rollNo << ", Name: " << s.name << ", SGPA: " << s.SGPA << endl;

}

// Bubble Sort - Sort students by Roll Number in ascending order

void bubbleSortRollNo(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] and students[j+1]

swap(students[j], students[j + 1]);

}

}

}

}

// Insertion Sort - Sort students by Name in alphabetical order

void insertionSortName(Student students[], int n) {

for (int i = 1; i < n; i++) {

Student key = students[i];

int j = i - 1;

// Shift elements of students[0..i-1] that are greater than key.name

while (j >= 0 && students[j].name > key.name) {

students[j + 1] = students[j];

j = j - 1;

}

students[j + 1] = key;

}

}

// Binary Search to find a student by Name

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

int left = 0, right = n - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

// Compare the targetName with the middle student's name

if (students[mid].name == targetName) {

return mid; // Found the student

}

else if (students[mid].name < targetName) {

left = mid + 1;

}

else {

right = mid - 1;

}

}

return -1; // Not found

}

int main() {

int n;

// Ask user for the number of students

cout << "Enter the number of students: ";

cin >> n;

// Create an array of students

Student\* students = new Student[n];

// Input student data

for (int i = 0; i < n; i++) {

cout << "\nEnter details for student " << i + 1 << ":\n";

cout << "Enter Roll Number: ";

cin >> students[i].rollNo;

cin.ignore(); // To ignore the newline character left by cin

cout << "Enter Name: ";

getline(cin, students[i].name);

cout << "Enter SGPA: ";

cin >> students[i].SGPA;

}

// a) Sort students by Roll Number using Bubble Sort

bubbleSortRollNo(students, n);

cout << "\nSorted by Roll No (Ascending):" << endl;

for (int i = 0; i < n; i++) {

displayStudent(students[i]);

}

// b) Sort students by Name using Insertion Sort

insertionSortName(students, n);

cout << "\nSorted by Name (Alphabetically):" << endl;

for (int i = 0; i < n; i++) {

displayStudent(students[i]);

}

// c) Search for a student by name using Binary Search

string searchName;

cout << "\nEnter name to search: ";

cin.ignore(); // Ignore the newline from previous input

getline(cin, searchName);

// Perform Binary Search (the list must be sorted alphabetically)

int result = binarySearchByName(students, n, searchName);

if (result != -1) {

cout << "Student found: " << endl;

displayStudent(students[result]);

} else {

cout << "Student not found!" << endl;

}

// Free the dynamically allocated memory

delete[] students;

return 0;}