#include <iostream>

using namespace std;

struct student

{

int rollno;

string name;

float sgpa;

};

void bubblesort(student arr[], int n)

{

for (int i = 0; i < n-1; i++)

{

for (int j = 0; j < n-i-1; j++)

{

if (arr[j].rollno > arr[j+1].rollno)

{

student a = arr[j];

arr[j] = arr[j+1];

arr[j+1] = a;

}

}

}

}

void insertionsort(student arr[], int n)

{

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

{

student key = arr[i];

int j = i - 1;

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

{

arr[j + 1] = arr[j];

j = j - 1;

}

arr[j + 1] = key;

}

}

int partition(student arr[], int low, int high)

{

float pivot = arr[high].sgpa;

int i = low - 1;

for (int j = low; j < high; ++j)

{

if (arr[j].sgpa > pivot)

{

++i;

student temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

student temp = arr[i + 1];

arr[i + 1] = arr[high];

arr[high] = temp;

return i + 1;

}

void quicksort(student arr[], int low, int high)

{

if (low < high)

{

int pi = partition(arr, low, high);

quicksort(arr, low, pi - 1);

quicksort(arr, pi + 1, high);

}

}

void searchSgpa(student arr[], int n, float search)

{

bool found = false;

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

{

if (arr[i].sgpa == search)

{

if (!found)

{

cout << "Students with SGPA " << search << ":" << endl;

found = true;

}

cout << "Name: " << arr[i].name << endl;

}

}

if (!found)

{

cout << "No students found with SGPA " << search << endl;

}

}

int binarysearch(student arr[], int size, string key)

{

int start = 0;

int end = size - 1;

int mid;

while (start <= end)

{

mid = start + (end - start) / 2;

if (arr[mid].name == key)

{

return mid;

}

if (key > arr[mid].name)

{

start = mid + 1;

}

else

{

end = mid - 1;

}

}

return -1;

}

int main()

{

student s[5];

cout << "Enter the details of 5 students:" << endl;

for (int i = 0; i < 5; i++)

{

cout << "Enter the name of student " << i + 1 << ": ";

cin >> s[i].name;

cout << "Enter the roll no of student " << i + 1 << ": ";

cin >> s[i].rollno;

cout << "Enter the SGPA of student " << i + 1 << ": ";

cin >> s[i].sgpa;

cout << endl;

}

cout << "Details of students are:" << endl;

for (int i = 0; i < 5; i++)

{

cout << "Student " << i + 1 << ":" << endl;

cout << "Name: " << s[i].name << endl;

cout << "Roll no: " << s[i].rollno << endl;

cout << "SGPA: " << s[i].sgpa << endl;

cout << endl;

}

// Bubble sort by roll number

bubblesort(s, 5);

cout << "Sorted by roll number:" << endl;

for (int j = 0; j < 5; j++)

{

cout << s[j].rollno << endl;

}

// Insertion sort by name

insertionsort(s, 5);

cout << "Sorted by names:" << endl;

for (int i = 0; i < 5; i++)

{

cout << s[i].name << endl;

}

cout << endl;

// Quick sort by SGPA

quicksort(s, 0, 4);

cout << "Top three students based on SGPA:" << endl;

for (int i = 0; i < 3; i++)

{

cout << "Name: " << s[i].name << ", Roll no: " << s[i].rollno << ", SGPA: " << s[i].sgpa << endl;

}

cout << endl;

// Search by SGPA

float search;

cout << "Enter SGPA to search: ";

cin >> search;

cout << endl;

searchSgpa(s, 5, search);

// Binary search by name

string key;

cout << "Enter name to search: ";

cin >> key;

cout << endl;

// Ensure the array is sorted by name before binary search

insertionsort(s, 5);

int index = binarysearch(s, 5, key);

if (index != -1)

{

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

cout << "Name: " << s[index].name << endl;

cout << "Roll no: " << s[index].rollno << endl;

cout << "SGPA: " << s[index].sgpa << endl;

}

else

{

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

}

return 0;

}





