#include <bits/stdc++.h>

using namespace std;

struct student {

string Name;

int roll;

float SGPA;

};

bool compareNames(const student\* s1, const student\* s2) {

return s1->Name < s2->Name;

}

void bubbleSortByRoll(student\* arr[], int size) {

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

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

if (arr[j]->roll > arr[j + 1]->roll) {

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

}

}

}

}

int partition(student\* arr[], int low, int high) {

float pivot = arr[high]->SGPA;

int i = (low - 1);

for (int j = low; j <= high - 1; j++) {

if (arr[j]->SGPA < pivot) {

i++;

swap(arr[i], arr[j]);

}

}

swap(arr[i + 1], arr[high]);

return (i + 1);

}

void quickSortBySGPA(student\* arr[], int low, int high) {

if (low < high) {

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

quickSortBySGPA(arr, low, pivotIndex - 1);

quickSortBySGPA(arr, pivotIndex + 1, high);

}

}

int main() {

const int var = 60;

student\* student1[var];

int ss = 0;

int a;

do {

cout << "1. Enter Record" << endl;

cout << "2. Display" << endl;

cout << "3. Linear Search" << endl;

cout << "4. Binary Search" << endl;

cout << "5. Bubble Sort" << endl;

cout << "6. Quick Sort" << endl;

cout << "7. Insertion Sort" << endl;

int p;

cin >> p;

switch (p) {

case 1: {

int num;

cout << "Enter number of students:";

cin >> num;

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

student1[ss] = new student;

cout << "Enter the name of student:";

cin >> student1[ss]->Name;

cout << "Enter the roll:";

cin >> student1[ss]->roll;

cout << "Enter the SGPA:";

cin >> student1[ss]->SGPA;

ss++;

}

break;

}

case 2: {

for (int j = 0; j < ss; j++) {

cout << "Name: " << student1[j]->Name << endl;

cout << "Roll: " << student1[j]->roll << endl;

cout << "SGPA: " << student1[j]->SGPA << endl;

}

break;

}

case 3: {

float SGPA;

cout << "Enter the SGPA of student to search:";

cin >> SGPA;

bool found = false;

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

if (student1[i]->SGPA == SGPA) {

cout << "Name: " << student1[i]->Name << endl;

cout << "Roll: " << student1[i]->roll << endl;

cout << "SGPA: " << student1[i]->SGPA << endl;

found = true;

break;

}

}

if (!found) {

cout << "Not Available" << endl;

}

break;

}

case 4: {

string name;

cout << "Enter the name of student to search:";

cin >> name;

sort(student1, student1 + ss, compareNames);

int left = 0, right = ss - 1;

bool found = false;

while (left <= right) {

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

if (student1[mid]->Name == name) {

cout << "Name: " << student1[mid]->Name << endl;

cout << "Roll: " << student1[mid]->roll << endl;

cout << "SGPA: " << student1[mid]->SGPA << endl;

found = true;

break;

} else if (student1[mid]->Name < name) {

left = mid + 1;

} else {

right = mid - 1;

}

}

if (!found) {

cout << "Not Available" << endl;

}

break;

}

case 5: {

bubbleSortByRoll(student1, ss);

for (int j = 0; j < ss; j++) {

cout << "Name: " << student1[j]->Name << endl;

cout << "Roll: " << student1[j]->roll << endl;

cout << "SGPA: " << student1[j]->SGPA << endl;

}

break;

}

case 6: {

quickSortBySGPA(student1, 0, ss - 1);

for (int j = 0; j < ss; j++) {

cout << "Name: " << student1[j]->Name << endl;

cout << "Roll: " << student1[j]->roll << endl;

cout << "SGPA: " << student1[j]->SGPA << endl;

}

break;

}

default: {

cout << "Enter Valid Input" << endl;

}

}

cout << "If you want to continue, press 1. To exit, press 0: ";

cin >> a;

} while (a == 1);

return 0;

}