#include <iostream>

#include<string.h>

using namespace std;

struct Student

{

int rollno;

char name[30];

float sgpa;

};

void display(struct Student s[],int n);

void bubbleSort(struct Student s[],int n);

void insertionSort(struct Student s[],int n);

void linearSearch(struct Student s[],int n, float data);

void binarySearch(struct Student s[],int n,char data[]);

void quickSort(struct Student s[],int n,int lb,int ub);

int partition(struct Student s[],int lb,int ub);

int main()

{

struct Student s[30];

int i,n,ch;

int count=0;

float sg;

char na[30];

cout <<"Enter total students: ";

cin>>n;

cout<<"\n Enter student information: ";

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

{

cout<<"\n Enter roll number, name and SGPA ";

cin>>s[i].rollno>>s[i].name>>s[i].sgpa;

}

display(s,n);

do

{

cout<<"\n\n 1. Bubble Sort(Sorting according to roll number wise)";

cout<<"\n 2. Insertion Sort(Sorting according to names)";

cout<<"\n 3. Linear Search(Searching based on SGPA)";

cout<<"\n 4. Binary Search(Searching based on given Name)";

cout<<"\n 5. Quick Sort(Top ten toppers)";

cout<<"\n 6. Exit";

cout<<"\n Enter your choice: ";

cin>>ch;

switch(ch)

{

case 1: bubbleSort(s,n);

break;

case 2: insertionSort(s,n);

break;

case 3: cout<<"\n enter SGPA to search: ";

cin>>sg;

linearSearch(s,n,sg);

break;

case 4: cout<<"\n enter name to search: ";

cin>>na;

binarySearch(s,n,na);

break;

case 5: quickSort(s,n,0,n-1);

cout<<"\n Top ten toppers are:\n";

for(i=n-1;i>=0;i--)

{

count++;

if(count>10)

{

break;

}

cout<<"\n"<<s[i].rollno<<"\t"<<s[i].name<<"\t"<<s[i].sgpa;

}

break;

case 6: break;

}

}while(ch!=6);

return 0;

}

void display(struct Student s[30],int n)

{

cout<<"\n Roll No\tName\tSGPA";

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

{

cout<<"\n"<<s[i].rollno<<" \t"<<s[i].name<<" \t"<<s[i].sgpa;

}

}

void bubbleSort(struct Student s[30],int n)

{

int i,j;

int temp;

char na[30];

float sg;

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

{

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

{

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

{

temp=s[j].rollno;

s[j].rollno=s[j+1].rollno;

s[j+1].rollno=temp;

sg=s[j].sgpa;

s[j].sgpa=s[j+1].sgpa;

s[j+1].sgpa=sg;

strcpy(na,s[j].name);

strcpy(s[j].name,s[j+1].name);

strcpy(s[j+1].name,na);

}

}

}

cout<<"\n Sorted data according to Roll Numbers:\n";

display(s,n);

}

void insertionSort(struct Student s[30],int n)

{

int i,j,r;

float sg;

char na[30];

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

{

strcpy(na,s[i].name);

r=s[i].rollno;

sg=s[i].sgpa;

j=i-1;

while(j>=0 && strcmp(s[j].name,na)>0)

{

strcpy(s[j+1].name,s[j].name);

s[j+1].rollno=s[j].rollno;

s[j+1].sgpa=s[j].sgpa;

j--;

}

strcpy(s[j+1].name,na);

s[j+1].rollno=r;

s[j+1].sgpa=sg;

}

cout<<"\n Sorted data according to Names:\n";

display(s,n);

}

void linearSearch(struct Student s[30],int n, float data)

{

int i,flag=0;

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

{

if(s[i].sgpa==data)

{

cout<<"\n"<<s[i].rollno<<"\t"<<s[i].name<<"\t"<<s[i].sgpa;

flag=1;

}

}

if(flag==0)

{

cout<<"\n Data Not Found.";

}

}

void binarySearch(struct Student s[30],int n,char data[30])

{

int left=0,right=n-1,mid,flag=0;

while(left<=right)

{

mid=(left+right)/2;

if(strcmp(data,s[mid].name)==0)

{

flag=1;

break;

}

else if(strcmp(data,s[mid].name)<0)

{

right=mid-1;

}

else

{

left=mid+1;

}

}

if(flag==1)

{

cout<<"\n Roll No="<<s[mid].rollno;

cout<<"\n Name="<<s[mid].name;

cout<<"\n SGPA="<<s[mid].sgpa;

}

else

{

cout<<"\n Data not found.";

}

}

int partition(struct Student s[30],int lb,int ub)

{

int start,end;

float pivot=s[lb].sgpa;

float sg;

int r;

char na[30];

start=lb;

end=ub;

while(start<end)

{

while(s[start].sgpa<=pivot)

{

start++;

}

while(s[end].sgpa>pivot)

{

end--;

}

if(start<end)

{

sg=s[start].sgpa;

s[start].sgpa=s[end].sgpa;

s[end].sgpa=sg;

r=s[start].rollno;

s[start].rollno=s[end].rollno;

s[end].rollno=r;

strcpy(na,s[start].name);

strcpy(s[start].name,s[end].name);

strcpy(s[end].name,na);

}

}

sg=s[lb].sgpa;

s[lb].sgpa=s[end].sgpa;

s[end].sgpa=sg;

r=s[lb].rollno;

s[lb].rollno=s[end].rollno;

s[end].rollno=r;

strcpy(na,s[lb].name);

strcpy(s[lb].name,s[end].name);

strcpy(s[end].name,na);

return end;

}

void quickSort(struct Student s[],int n,int lb,int ub)

{

if(lb<ub)

{

int loc=partition(s,lb,ub);

quickSort(s,n,lb,loc-1);

quickSort(s,n,loc+1,ub);

}

}

OUTPUT:

Enter total students: 4

Enter student information:

Enter roll number, name and SGPA

16 Samruddhi 9.1

Enter roll number, name and SGPA

34 Khushi 9.45

Enter roll number, name and SGPA

35 Anjali 9.45

Enter roll number, name and SGPA

38 Vaishnavi 10

Roll No Name SGPA

16 Samruddhi 9.1

34 Khushi 9.45

35 Anjali 9.45

38 Vaishnavi 10

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 1

Sorted data according to Roll Numbers:

Roll No Name SGPA

16 Samruddhi 9.1

34 Khushi 9.45

35 Anjali 9.45

38 Vaishnavi 10

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 1

Sorted data according to Roll Numbers:

Roll No Name SGPA

16 Samruddhi 9.1

34 Khushi 9.45

35 Anjali 9.45

38 Vaishnavi 10

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 2

Sorted data according to Names:

Roll No Name SGPA

35 Anjali 9.45

34 Khushi 9.45

16 Samruddhi 9.1

38 Vaishnavi 10

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 3

enter SGPA to search: 10

38 Vaishnavi 10

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 4

enter name to search: Khushi

Roll No=34

Name=Khushi

SGPA=9.45

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 5

Top ten toppers are:

38 Vaishnavi 10

35 Anjali 9.45

34 Khushi 9.45

16 Samruddhi 9.1

1. Bubble Sort(Sorting according to roll number wise)

2. Insertion Sort(Sorting according to names)

3. Linear Search(Searching based on SGPA)

4. Binary Search(Searching based on given Name)

5. Quick Sort(Top ten toppers)

6. Exit

Enter your choice: 6