#include <iostream>

using namespace std;

//Structure for student database

struct Student

{

int rno;

string name;

float sgpa;

};

//Class with data members and methods

class SE

{

private : struct Student S[25];

int n;

public : void Accept();

void Display();

void Linear();

void Display\_one(int c);

void Bubble();

void insertion();

void binary();

void quick\_sort(int l, int h);

int partition1(int l, int h);

void callQS();

};

//Function to accept data in the student database for each object

void SE :: Accept()

{

int rno;

string name;

float sgpa;

cout<<"Enter number of students : ";

cin>>n;

cout<<"\n";

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

{

cout<<"Enter roll no : ";

cin>>S[i].rno;

cout<<"Enter name : ";

cin>>S[i].name;

cout<<"Enter sgpa : ";

cin>>S[i].sgpa;

cout<<endl;

}

}

//Function to display accepted data in a tabular format

void SE :: Display()

{

int rno;

string name;

float sgpa;

cout<<"Roll no\t\tName\t\tSGPA"<<endl;

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

{

cout<<S[i].rno<<"\t\t";

cout<<S[i].name<<"\t\t";

cout<<S[i].sgpa<<endl;

}

cout<<"\n";

}

//Function to display only one entry from the database

void SE :: Display\_one(int c)

{

cout<<"Roll no: "<<S[c].rno<<"\t"<<"Name: "<<S[c].name<<"\t"<<"SGPA :"<<S[c].sgpa<<endl;

}

//Function to search student by sgpa by using linear search

void SE :: Linear()

{

float tsgpa;

cout<<"Enter sgpa to search : ";

cin>>tsgpa;

int counter = 0;

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

{

if(tsgpa == S[i].sgpa)

{

Display\_one(i);

counter++;

}

}

if(counter == 0)

{

cout<<"No student found"<<endl;

}

}

//Function to sort student database by roll numbers

void SE :: Bubble()

{

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

{

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

{

if(S[j].rno > S[j+1].rno)

{

struct Student t;

t = S[j];

S[j] = S[j+1];

S[j+1] = t;

cout<<"\nAfter swap : ";

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

cout<<S[i].rno<<"\t";

}

}

}

cout<<"\nAfter iteration : ";

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

cout<<S[i].rno<<"\t";

}

}

}

//Function to sort student database by name

void SE :: insertion()

{

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

{

struct Student t = S[i];

for(int j=i; j>0; j--)

{

if(S[j-1].name>[t.name](http://t.name/))

{

S[j] = S[j-1];

S[j-1] = t;

}

cout<<"\nAfter shift : ";

for(int i=0; i<n; i++) {cout<<S[i].name<<"\t";}

}

cout<<"\nAfter iteration : ";

for(int i=0; i<n; i++) {cout<<S[i].name<<"\t";}

}

}

//Function to sort student data by sgpa using quicksort (referred)

int SE :: partition1(int l, int h)

{

int i,j;

i = l;

j = h;

float pivot = S[i].sgpa;

while(i<j)

{

//cout<<"Pivot : "<<pivot<<endl;

while(S[i].sgpa<=pivot && i<h)

{

i++;

}

while(S[j].sgpa>pivot)

{

j--;

}

if(i<j)

{

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

cout<<"\nAfter swap of "<<S[i].sgpa<<" And "<<S[j].sgpa<<"\t";

for(int i=0; i<n; i++) {cout<<S[i].sgpa<<"\t";}

}

}

swap(S[l], S[j]);

cout<<"\nAfter swap of pivot and "<<S[j].sgpa<<"\t";

for(int i=0; i<n; i++) {cout<<S[i].sgpa<<"\t";}

cout<<"\n";

return j;

}

void SE :: quick\_sort(int l, int h)

{

if(l<h)

{

int I = partition1(l, h);

quick\_sort(l, I-1);

quick\_sort(I+1, h);

}

}

//Function to call quicksort

void SE :: callQS()

{

quick\_sort(0, n-1);

}

//Function to search by name

void SE :: binary()

{

int f=0,l=n-1,mid;

string key;

cout<<"Enter name to search : ";

cin>>key;

do

{

mid = (f+l)/2;

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

{

Display\_one(mid);

break;

}

else if(key>S[mid].name)

{

f = mid+1;

}

else if(key<S[mid].name)

{

l = mid-1;

}

else

{

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

}

}while(f<=l);

}

MAIN:

#include <iostream>

#include </home/administrator/23329\_harsha/sorting.cpp>

int main()

{

SE SE11;

SE11.Accept();

cout<<"SE11"<<endl;

SE11.Display();

int choice;

cout<<"\n";

int yes = 1;

while(yes == 1)

{

cout<<"\nOperation\n1. Sort by Roll numbers (Bubble sort)\n2. Sort by name (Insertion sort)\n3. Search student by sgpa (Linear search)\n4. Sort by student sgpa (Quicksort)\n5. Search by name (Binary search)\n6. Exit\n";

cout<<"Enter your choice : ";

cin>>choice;

switch(choice)

{

case 1: SE11.Bubble();

cout<<"\nSorted database : "<<endl;

SE11.Display();

break;

case 2:SE11.insertion();

cout<<"Sorted database : "<<endl;

SE11.Display();

break;

case 3: SE11.Linear();

break;

case 4:SE11.callQS();

cout<<"Sorted database : "<<endl;

SE11.Display();

break;

case 5:SE11.insertion();

SE11.binary();

break;

default : cout<<"Inalid input !";

case 6:

exit(0);

cout<<endl;

}

}

return 0;

}