/\*Write C++ program to store first year percentage of students in array. Write function for

sorting array of floating point numbers in ascending order using

a) Selection Sort

b) Bubble sort and display top five scores.\*/

#include<iostream>

#define size 10

using namespace std;

void selection(float \*s,int n)

{

int i,j;

float temp;

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

{

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

{

if(s[i]>s[j])

{

temp=s[i];

s[i]=s[j];

s[j]=temp;

}

}

}

}

void bubble(float \*s, int n)

{

int i,j;

float temp;

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

{

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

{

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

{

temp=s[i];

s[i]=s[j];

s[j]=temp;

}

}

}

}

int main()

{

int n;

int c=0;

float stud[size];

cout<<"\n enter total number of students";

cin>>n;

cout<<"\n enter parsentages of students";

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

{

cin>>stud[i];

}

selection(stud,n);

cout<<"\n top 5 students using selection sort:";

int i=n-1;

for(int c=0;c!=5;c++)

{

cout<<"\t"<<stud[i];

i--;

}

bubble(stud,n);

cout<<"\n top 5 students using bubble sort:";

i=n-1;

for(int c=0;c!=5;c++)

{

cout<<"\t"<<stud[i];

i--;

}

return 0;

}

/\*Write C++ program to store second year percentage of students in array. Write function for

sorting array of floating point numbers in ascending order using

a) Insertion sort

b) Shell Sort and display top five scores.\*/

**#include**<iostream>

**#define** size 10

**using** **namespace** std;

**void** **insertion**(**float** \*s,**int** n)

{

**int** i,p;

**float** temp;

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

{

temp=s[i];

p=i-1;

**while**((p>=0) && (s[p]>temp))

{

s[p+1]=s[p];

p--;

}

s[p+1]=temp;

}

cout<<"\n output of insertion sort";

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

{

cout<<"\t"<<s[i];

}

cout<<"\n top 5 students using insertion sort:";

i=n;

**for**(**int** c=0;c!=5;c++)

{

cout<<"\t"<<s[i];

i--;

}

}

**void** **shellsort**(**float** \*s, **int** n)

{

**int** i,j,gap,flag,swap;

**float** temp;

gap=n/2;

**while**(gap>0)

{

flag=0;

**while**(flag==0)

{

swap=0;

**for**(i=0;i<n-gap;i++)

{

**if**(s[i]>s[i+gap])

{

temp=s[i];

s[i]=s[i+gap];

s[i+gap]=temp;

swap=i;

}

}

**if**(swap==0)

flag=1;

}

gap=gap/2;

}

cout<<"\n output of shell sort";

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

{

cout<<"\n\t"<<s[i];

}

cout<<"\n top 5 students using shell sort:";

i=n;

**for**(**int** c=0;c!=5;c++)

{

cout<<"\t"<<s[i];

i--;

}

}

**int** **main**()

{

**int** n;

**int** c=0;

**float** stud[size];

cout<<"\n enter total number of students";

cin>>n;

cout<<"\n enter parsentages of students";

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

{

cin>>stud[i];

}

insertion(stud,n);

shellsort(stud,n);

**return** 0;

}

enter total number of students5

enter parsentages of students55.55

33.33

100.100

77.77

22.22

output of insertion sort 22.22 33.33 55.55 77.77 100.1

top 5 students using insertion sort: 100.1 77.77 55.55 33.33 22.22

output of shell sort

22.22

33.33

55.55

77.77

100.1

top 5 students using shell sort: 100.1 77.77 55.55 33.33 22.22

/\*Write C++ program to store names and mobile numbers of your friends in sorted order on  
names. a) Search your friend from list using binary search (recursive and non recursive).  
Insert friend if not present in phonebook. b) Search your friend from list using Fibonacci  
search. Insert friend if not present in phonebook.\*/  
  
#include<iostream>  
#include<strings.h>  
#include<string.h>  
  
#define size 10  
using namespace std;  
  
struct student  
{  
    char name[size];  
    long int mob;  
}s[size];  
void non\_rec\_binary(struct student[], char [],int);  
void rec\_binary(struct student[], char [],int);  
void fibo(struct student[],int,char[],int,int,int);  
int fib(int);  
void non\_rec\_binary(struct student s[],char key[],int n)  
{  
    int low,high,m,flag=0;  
    low=0;  
    high=n-1;  
  
    while(low<=high)  
    {  
        m=(low+high)/2;  
        if(strcmp(key,s[m].name)==0)  
        {  
            cout<<"\n search key is present at location "<<m;  
            flag=1;  
            break;  
        }  
        else if(strcmp(key,s[m].name)<0)  
            high=m-1;  
        else  
            low=m+1;  
  
    }  
    if(flag==0)  
        cout<<"\n search key is not awailable in list";  
  
}  
  
void rec\_binary(struct student s[],char key[],int n,int low,int high)  
{  
    int mid;  
  
    if(low>high)  
        cout<<"\n friend is not available in phonebook";

Else

{  
    mid=(low+high)/2;  
    if(strcmp(key,s[mid].name)==0)  
        cout<<"\n search key is present at location "<<mid;  
    else if(strcmp(key,s[mid].name)<0)  
        rec\_binary(s,key,n,low,mid-1);  
    else  
        rec\_binary(s,key,n,mid+1,high);  
}  
}  
  
int fib(int n)//obtaining fibonacci number  
{  
    int a,b,f;  
    if(n<1)  
        return n;  
    a=0;  
    b=1;  
    while(b<n)  
    {  
        f=a+b;  
        a=b;  
        b=f;  
    }  
    return a;  
}  
void fibo(struct student s[],int n,char key[],int f,int b,int a)  
{  
    if(f<1||f>n)  
        cout<<"\n the number is not present";  
    else if(strcmp(key,s[f].name)<0)  
    {  
        if(a<=0)  
            cout<<"\nelement is not present";  
        else  
            fibo(s,n,key,f-a,a,b-a);  
    }  
    else if(strcmp(key,s[f].name)>0)  
    {  
        if(b<=1)  
            cout<<"\nelement not present";  
        else  
            fibo(s,n,key,f+a,b-a,a-b);  
    }  
    else  
        cout<<"\nelement present at:"<<f;  
  
}  
  
int main()  
{  
    int n,low,high,a,b;  
    char key[size];  
    cout<<"\neneter total number of student";  
    cin>>n;  
    for(int i=0;i<n;i++)  
    {  
        cout<<"\nenter name of student "<<i;  
        cin.ignore();  
        cin.getline(s[i].name, sizeof(s[i].name));  
        cout<<"\nenter phone no of student"<<i;  
        cin>>s[i].mob;  
    }  
    cout<<"\nname\tmob no\n";  
    for(int i=0;i<n;i++)  
    {  
        cout<<s[i].name<<"\t"<<s[i].mob<<"\n";  
    }  
    cout<<"\nsearching of friend using non recursive binary search\n";  
    cout<<"\nenter key element(friend name) to be searched\n";  
    cin.ignore();  
    cin.getline(key,sizeof(key));  
    cout<<"\n key element:"<<key;  
    non\_rec\_binary(s,key,n);  
  
    cout<<"\nsearching of friend using recursive binary search\n";  
    cout<<"\nenter key element(friend name) to be searched\n";  
    cin.getline(key,sizeof(key));  
    cout<<"\n key element:"<<key;  
    low=0;  
    high=n-1;  
    rec\_binary(s,key,n,low,high);  
  
    cout<<"\nsearching of friend using fibonacci search\n";  
    cout<<"\nenter key element(friend name) to be searched\n";  
    //cin.ignore();  
    cin.getline(key,sizeof(key));  
    cout<<"\n key element:"<<key;  
    b=fib(n);  
    a=fib(b);  
    fibo(s,n,key,n-1,b,a);  
    return 0;  
}