```
 #include<iostream>
#include<cstring>
using namespace std;
struct student{
int rno;
char name[50];
float sgpa;
};
void accept(struct student list[4]);
void display(struct student list[4]);
void bubblesort(struct student list[4]);
void insertionsort(struct student list[4]);
void search(struct student list[4]);
void binarysearch(struct student list[4]);
void quicksort(struct student list[4],int start,int end);
int partition(struct student list[4],int start,int end);
int main(){
int ch,choice;
do{
char con[5];
int start=0;int end=3;
struct student data[4];
accept(data);
cout<<"\n 1:Bubble sort";
cout<<"\n 2:Insertion sort";
cout<<"\n 3:Search";
cout<<"\n 4:Binary search";
cout<<"\n 5:Quick sort";
cout<<"\n Enter your choice:";
cin>>ch;
switch(ch){
case 1:bubblesort(data);
display(data);
break;
case 2:insertionsort(data);
display(data);
break;
case 3:search(data);
break;
case 4:binarysearch(data);
break;
case 5:quicksort(data,0,3);
display(data);
break;
default:
cout<<"Invalid Choice...";
}
cout<<"\n\nDo you want to continue (Yes=1/No=0) : ";
cin>>choice;
}while(choice==1);
```

```
}
void accept(struct student list[4]){
int i;
for(i=0;i<4;i++){
cout<<"\nEnter Roll.No, Name and SGPA of student : ";
cin>>list[i].rno>>list[i].name>>list[i].sgpa;
}
}
void display(struct student list[4]){
int i;
cout<<"\nRoll No\tName\tSGPA";
for(i=0;i<4;i++){
cout<<"\n"<<list[i].rno<<"\t"<<list[i].name<<"\t"<<list[i].sgpa;
}
}
void bubblesort(struct student list[4]){
int k,j;
struct student temp;
for(k=0;k<4-1;k++){
for(j=0;j<(4-k-1);j++){
if(list[j].rno>list[j+1].rno){
temp=list[j];
list[j]=list[j+1];
list[j+1]=temp;
}
}
}
}
void insertionsort(struct student list[4]){
int k,j;
struct student temp;
for(k=1;k<4;k++){
temp=list[k];
j=k-1;
while(strcmp(list[j].name,temp.name)>0 && j>=0){
list[j+1]=list[j];
--j;
}
list[j+1]=temp;
}
}
void search(struct student list[4]){
int i;
float sgpa;
cout<<"Enter SGPA to search : ";
cin>>sgpa;
cout<<"\nRoll No\tName\tSGPA";
for(i=0;i<4;i++){
```

```
        if(sgpa==list[i].sgpa){                                                 }
        cout<<"\n"<<list[i].rno<<"\t"<<list[i].name<<"\t"<<list[i].sgpa;        }
                                                                                }
void binarysearch(struct student list[4]){
int k,lower,upper,mid;
char binarysearch1[80];
cout<<"Enter the name of the student : ";
cin>>binarysearch1;
lower=0;
upper=4-1;
mid=(lower+upper)/2;
while(lower<=upper){
if(strcmp(list[mid].name,binarysearch1)<0){
lower = mid + 1;
}
else if(strcmp(list[mid].name,binarysearch1)==0){
cout<<"\nRoll No\tName\tSGPA";
cout<<"\n"<<list[mid].rno<<"\t"<<list[mid].name<<"\t"<<list[mid].sgpa;
break;
}
else{
upper=mid-1;
mid=(lower+upper)/2;
}
if(lower>upper){
cout<<binarysearch1<<" not found in the list";
}
}
}
int partition(struct student list[4],int start,int end){
struct student pivot;
pivot=list[start];
int count=0;
for(int i=start+1;i<=end;i++){
if(list[i].sgpa>=pivot.sgpa){
count++;
}
}
int pivotindex=start+count;
swap(list[pivotindex],list[start]);
//Sorting left and right parts of the pivot element
int i=start;
int j=end;
while(i<pivotindex && j>pivotindex){
while(list[i].sgpa>=pivot.sgpa){
i++;
}
while(list[j].sgpa<pivot.sgpa){
j--;
```

```
}                                                   }
if(i>pivotindex &&
j<pivotindex){
swap(list[i++],list[j--]);
}
}
return pivotindex;
void quicksort(struct student
list[4],int start,int end){
if(start<end){
int partitionindex =
partition(list,start,end);
quicksort(list,start,partitionindex-
1);
quicksort(list,partitionindex+1,end
);
}
}
```