Consider a student database of SEIT class (at least 15 records). Database contains different fields of
every student like Roll No, Name and SGPA.(array of structure)
A. Design a roll call list, arrange list of students according to roll numbers in ascending order (Use
Bubble Sort)
B. Arrange a list of students alphabetically. (Use Insertion sort)
C. Arrange a list of students to find the first ten toppers from a class. (Use Quick sort)
D. Search students according to SGPA. If more than one student has the same SGPA, then print a
list of all students having the same SGPA.
E. Search a particular student according to name using binary search without recursion.

Code

```
#include <iostream>
using namespace std;

struct student
{
 int roll;
 string name;
 float sgpa;
};

void show1(student *a[], int n)
```

```
{
 for (int i=0;i<n;i++)
 {
   cout<<a[i]->roll<<" ";
   cout<<a[i]->name<<" ";
   cout<<a[i]->sgpa<<"\n";

 }
}

void show2(student *a[], int n)
{
 for (int i=n-1;i>n-11;i--)
 {
   cout<<a[i]->roll<<" ";
   cout<<a[i]->name<<" ";
   cout<<a[i]->sgpa<<"\n";
 }
}

void bubSort(student *a[],int n)
{ student *t=new student;
 int check=1;
 for (int i=0;i<n-1;i++)
```

```
 {

   for (int j=0;j<n-i-1;j++)
   {
     if (a[j]->roll>a[j+1]->roll)
     {
       t= a[j];
       a[j] = a[j+1];
       a[j+1] = t;
     }
     else
     {
       check=0;
     }
   }
   if (check=0)
   {
     break;
   }
 }
}

void inserSort(student *a[],int n)
{
```

```
 int j;
 student *t;
 string k;
 for (int i=0;i<n;i++)
 {
   k=a[i]->name;
   t=a[i];
   j=i-1;
   while(j>=0 && a[j]->name>k)
   {
     a[j+1]=a[j];
     j--;
   }
   a[j+1]=t;

 }
}

int partition (student *a[], int l, int u)
{
 student *t;

 float p=a[l]->sgpa;
 int start=l,end=u;
```

```
 {
   while(a[start]->sgpa <= p && start<u)
   {
     start++;
   }
   while (a[end]->sgpa>p)
   {
     end--;
   }

   if(start < end)
   {
     swap(a[start],a[end]);
   }
   else
   {
     t=a[end];
     a[end]=a[l];
     a[l]=t;
     return end--;
   }
 }
 return 0;
}
```

```
void quickSort(student *a[], int l, int u)
{
 if (l< u)
 {
   int loc = partition(a, l, u);
   quickSort(a, l, loc - 1);
   quickSort(a, loc + 1, u);
 }
}

void biSearch(student *a[],int l,int u, string k)
{ student *t=new student;
 int i,mid,check=0;
 while(l<=u)
 {
   mid=(u+l)/2;
   if (a[mid]->name==k)
   {
     t=a[mid];
     check=1;
     i=mid;
     break;
   }
```

```
  else if(a[mid]->name<k)
  {
    l=mid+1;
  }
  else
  {
    u=mid-1;
  }
}
if (check==0)
{
  cout<<"NO MATCH FOUND"<<"\n";
}
else
{
  while(i >=0 && a[i]->name == a[mid]->name)
  {
    i--;
  }

  i++;
  while(a[i]->name == a[mid]->name && i <= u)
  {
    cout<<a[i]->roll<<" ";
```

```
    cout<<a[i]->name<<" ";
    cout<<a[i]->sgpa<<"\n";
    i++;
   }
 }

}

void linSearch(student *a[],int u,float x)
{ int check=0;
 for (int i=0;i<u;i++)
 {
   if(a[i]->sgpa==x)
   {
    cout<<a[i]->roll<<" ";
    cout<<a[i]->name<<" ";
    cout<<a[i]->sgpa<<"\n";
    check=1;
   }
 }
 if(check==0)
   cout<<"NO MATCH FOUND"<<"\n";

}
```

```
int main()
{
 int i=0,n,ch;
 string cf;
 float m;
 student *s[60];

 loop1:
 cout<<"\n ENTER YOUR CHOICE: \n";
 cout<<"1. INSERT RECORDS: \n";
 cout<<"2. CLASS DETAILS(According to ROLL
NO.) USING BUBBLE SORT \n";
 cout<<"3. FIRST TEN TOPPERS OF THE CLASS
USING QUICK SORT \n";
 cout<<"4. CLASS DETAILS (IN ALPHABETICAL
ORDER) \n";
 cout<<"5. FIND STUDENT (According to SGPA)
USING BINARY SEARCH \n";
 cout<<"6. FIND STUDENT (According to Name)
USING BINARY SEARCH\n";
 cin>>n;
 if(n==1)
 {
```

```
  cout<<"ENTER DETAILS: \n";
  do
  {
    cout<<"STUDENT "<<i+1<<" RECORD:\n";
    s[i]=new student;
    cout<<"Roll no :";
    cin>>s[i]->roll;
    cout<<"Name: ";
    cin>>s[i]->name;
    cout<<"SGPA: ";
    cin>>s[i]->sgpa;

    cout<<"DO YOU WANT TO INPUT MORE
RECORDS (enter 1 if yes) \n";
    cin>>ch;
    i++;

  }
  while(ch==1);
  goto loop1;
 }
 if(n==2)
 {
```

```
   cout<<"**CLASS DETAILS(according to ROLL
NO.)**"<<"\n"<<"\n";
   bubSort(s,i);
   show1(s,i);
 }
 else if(n==3)
 {
   cout<<"**FIRST TEN TOPPERS**"<<"\n"<<"\n";
   quickSort(s,0,i-1);
   show2(s,i);
 }
 else if(n==4)
 {
   cout<<"**CLASS DETAILS (IN ALPHABETICAL
ORDER)**"<<"\n"<<"\n";
   inserSort(s,i);
   show1(s,i);
 }
 else if(n==5)
 {
   cout<<"**FIND STUDENT (According to
SGPA)**"<<"\n"<<"ENTER SGPA"<<"\n"<<"\n";
   cin>>m;
   linSearch(s,i,m);
```

```
 }
 else if(n==6)
 {
   cout<<"**FIND STUDENT (According to
Name)**"<<"\n"<<"ENTER NAME"<<"\n"<<"\n";
   string nm;
   cin>>nm;
   inserSort(s,i);
   biSearch(s,0,i,nm);
 }
 else
 {
   cout<<"INVALID INPUT!!\n";
 }
 cout<<"WANT TO CONTINUE? (enter y or Y):\n";
 cin>>cf;
 if(cf=="Y"||"y")
 {
   goto loop1;
 }
 return 0;
}
```

Output

```
 ENTER YOUR CHOICE:
1. INSERT RECORDS:
2. CLASS DETAILS(According to ROLL NO.) USING BUBBLE SORT
3. FIRST TEN TOPPERS OF THE CLASS USING QUICK SORT
4. CLASS DETAILS (IN ALPHABETICAL ORDER)
5. FIND STUDENT (According to SGPA) USING BINARY SEARCH
6. FIND STUDENT (According to Name) USING BINARY SEARCH
2
**CLASS DETAILS(according to ROLL NO.)**

14 Shravani 6.8
48 Harsh 8.5
49 Soma 6.9
50 Kinjal 7.3
51 Gauri 8.9
WANT TO CONTINUE? (enter y or Y):
y
```

```
3
**FIRST TEN TOPPERS**

51 Gauri 8.9
48 Harsh 8.5
50 Kinjal 7.3
49 Soma 6.9
14 Shravani 6.8
```

```
**CLASS DETAILS (IN ALPHABETICAL ORDER)**

51 Gauri 8.9
48 Harsh 8.5
50 Kinjal 7.3
14 Shravani 6.8
49 Soma 6.9
```

```
**FIND STUDENT (According to SGPA)**
ENTER SGPA

8.9
51 Gauri 8.9
WANT TO CONTINUE? (enter y or Y):
y
```

```
6
**FIND STUDENT (According to Name)**
ENTER NAME

Harsh
48 Harsh 8.5
WANT TO CONTINUE? (enter y or Y):
```