```
#include<iostream>
#include<cstring>
#include <chrono>

using namespace std;

//structure for a student
struct Student{
    char name[50];
    int rno;
    float sgpa;
};

int swapcount;
int iteration_cnt;
int comparison_cnt;

class StudentDB{
    struct Student s[100];
    int num_of_stud=0;
    public:

    // Function to add a student record
    void addstudent(){
        cout<<"Enter rollnumber: ";
        cin>>s[num_of_stud].rno;
        cout<<"Enter Name: ";
        cin>>s[num_of_stud].name;
        cout<<"Enter SGPA: ";
        cin>>s[num_of_stud].sgpa;
        num_of_stud+=1;
    }

    // Function to display all student records
    void display(){
        cout<<"Rollnumber"<<"    "<<"Name"<<"        "<<"SGPA"<<endl;
        cout<<"_____________________________________"<<endl;
        cout<<endl;
        for(int i=0;i<num_of_stud;i++){

           cout<<s[i].rno<<"         "<<s[i].name<<"
"<<s[i].sgpa<<endl;
        }
    }

    // Function to display a specific student record
    void displayrec(){
        int index;
        cout<<"Enter the record number of student: "<<endl;
        cin>>index;
        cout<<"record "<<index<<endl;
        cout<<s[index-1].rno<<"       "<<s[index-1].name<<"
"<<s[index-1].sgpa<<endl;

    }

    // Display the menu of operations
    void menu(){
      cout<<endl;
```

```
        cout<<"Enter the operation: "<<endl;
        cout<<"1. Add student record "<<endl;
        cout<<"2. Display all student records "<<endl;
        cout<<"3. Display student record "<<endl;
        cout<<"4. Sort by roll numbers" <<endl;
        cout<<"5. Sort by Names"<<endl;
        cout<<"6. Search by Names"<<endl;
        cout<<"7. Sort by SGPA"<<endl;
        cout<<"8. Search SGPA"<<endl;
        cout<<"9. Exit"<<endl;
        cout<<endl;


    }

    // Sort student records by roll numbers using the Bubble Sort
algorithm
    void rno_sort(){
        swapcount=0;
        iteration_cnt=0;
        comparison_cnt=0;
        bool flag=false;
        cout<<"Sorted all records by roll numbers.";
        auto start = std::chrono::high_resolution_clock::now();
        for(int i=0;i<num_of_stud-1;i++){
            iteration_cnt+=1;
            for(int j=0;j<num_of_stud-i-1;j++){
                comparison_cnt+=1;
                if(s[j].rno>s[j+1].rno)
                {
                    flag=true;
                    Student temp=s[j];
                    s[j]=s[j+1];
                    s[j+1]=temp;
                    swapcount+=1;
                }
                if(flag==false){
                    break;
                }
            }

        }
        auto end = std::chrono::high_resolution_clock::now();
        chrono::duration<double,
        milli> duration = end - start;
        cout<<"Total time required = "<<duration.count()<<"
milliseconds"<<endl;
    }

    // Binary search for a student by name
    void name_search() {
        int l=0;
        int h=num_of_stud-1;
        char elem[50];

        cout<<"Enter the name to be searched: "<<endl;
        cin>>elem;

        while(h>=l){
            int mid=(l + h)/2;
```

```
            int result=strcmp(s[mid].name, elem);

            if (result==0){
                cout<<"Record found at location: "<<mid+1<<endl;
                return;
            }
            else if(result<0){
                l=mid+1;
            }
            else{
                h=mid-1;
            }
        }
        cout<<"Name not found in the array."<<endl;

    }

    // Sort student records by names using the Insertion Sort algorithm
    void name_sort(){
        swapcount=0;
        iteration_cnt=0;
        comparison_cnt=0;
        auto start = std::chrono::high_resolution_clock::now();
        for(int i=1;i<num_of_stud;i++){
            iteration_cnt+=1;
            Student key=s[i];
            int j=i-1;
            while(j>=0 && strcmp(s[j].name,key.name)>0){
                comparison_cnt+=1;
                s[j+1]=s[j];
                j--;
                swapcount+=1;
            }
            s[j+1]=key;
        }
        auto end = std::chrono::high_resolution_clock::now();
        chrono::duration<double,
        milli> duration = end - start;
        cout << "Total time required = " << duration.count() << "
milliseconds" <<endl;

    }

    // Partition function for the Quick Sort algorithm
    int partition(int low,int high){
      int i=low;
      int j=high;
      float pivot=s[i].sgpa;
      while(i<j){
          iteration_cnt+=1;
            while(s[i].sgpa >= pivot && i<=high){
                i++;
                comparison_cnt+=1;
            }
            while(s[j].sgpa < pivot && j>=low){
                j--;
                comparison_cnt+=1;
            }
```

```
            if(i<j){
                 Student temp=s[i];
                 s[i]=s[j];
                 s[j]=temp;
                 swapcount+=1;
                }
          }
            Student temp=s[low];
            s[low]=s[j];
            s[j]=temp;
            swapcount+=1;
            return j;
        }

     // Quick Sort algorithm to sort student records by SGPA
    void qs(int low,int high){
      if(low<high){
         int pIndex=partition(low,high);
            qs(low,pIndex-1);
            qs(pIndex+1,high);
      }
    }

    // Sort student records by SGPA using Quick Sort
    void sgpa_sort(){
        auto start = std::chrono::high_resolution_clock::now();
      qs(0,num_of_stud-1);
      auto end = std::chrono::high_resolution_clock::now();
         chrono::duration<double,
         milli> duration = end - start;
         cout << "Total time required = " << duration.count() << "
milliseconds" <<endl;

    }

    // Search Sgpa by linear search
    void sgpa_search(){
        int i;
        float element;
        cout<<"Enter the sgpa to be searched: ";
        cin>>element;
        for(i=0;i<num_of_stud;i++){
            if(s[i].sgpa==element)
            {
                cout<<"Record found at record number "<<i+1<<endl;
                break;
            }
        }
        if(i==num_of_stud){
            cout<<"Record not found"<<endl;
        }

    }
};

//Main function
int main(){
    StudentDB s1;
    int op;
    while(1){
```

```
        s1.menu();
        cin>>op;
        switch(op){
            case 1:s1.addstudent();
            break;

            case 2:s1.display();
            break;

            case 3:s1.displayrec();
            break;

            case 4:s1.rno_sort();
            cout<<endl;
            cout<<"Swap count: "<<swapcount<<endl;
            cout<<"iteration count: "<<iteration_cnt<<endl;
            cout<<"comparison count: "<<comparison_cnt<<endl;
            break;

            case 5:s1.name_sort();
            cout<<"Sorted all record by names"<<endl;
            cout<<"Shift count: "<<swapcount<<endl;
            cout<<"iteration count: "<<iteration_cnt<<endl;
            cout<<"comparison count: "<<comparison_cnt<<endl;
            break;

            case 6:s1.name_search();
            break;

            case 7:s1.sgpa_sort();
            cout<<"Sorted all records by sgpa"<<endl;
            cout<<"Swap count: "<<swapcount<<endl;
            cout<<"iteration count: "<<iteration_cnt<<endl;
            cout<<"comparison count: "<<comparison_cnt<<endl;
            break;

            case 8:s1.sgpa_search();
            break;

            case 9:exit(0);
        }
    }

return 0;
}
```

```
//OUTPUT
Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

1
Enter rollnumber: 23323
Enter Name: Pankaj
Enter SGPA: 6.9
Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

1
Enter rollnumber: 23324
Enter Name: Lalit
Enter SGPA: 9.41
Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

1
Enter rollnumber: 23325
Enter Name: Shounak
Enter SGPA: 9.84
Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

1
Enter rollnumber: 23326
Enter Name: Swaraj
```

```
Enter SGPA: 9.27
Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

1
Enter rollnumber: 23327
Enter Name: Shravani
Enter SGPA: 9
Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

2
Rollnumber    Name        SGPA
_____________________________

23323         Pankaj       6.9
23324         Lalit       9.41
23325         Shounak       9.84
23326         Swaraj       9.27
23327         Shravani       9

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

3
Enter the index of student:
3
record 3
23325       Shounak       9.84

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
```

```
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

5
Total time required = 0.00307 milliseconds
Sorted all record by names
Shift count: 2
iteration count: 4
comparison count: 2

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

2
Rollnumber    Name        SGPA
_____________________________

23324         Lalit       9.41
23323         Pankaj       6.9
23325         Shounak       9.84
23327         Shravani       9
23326         Swaraj       9.27

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

7
Total time required = 0.00146 milliseconds
Sorted all records by sgpa
Swap count: 6
iteration count: 8
comparison count: 12

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
```

```
9. Exit

2
Rollnumber    Name        SGPA
_____________________________

23325         Shounak       9.84
23324         Lalit       9.41
23326         Swaraj       9.27
23327         Shravani       9
23323         Pankaj       6.9

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

4
Sorted all records by roll numbers.Total time required = 0.00082
milliseconds

Swap count: 5
iteration count: 4
comparison count: 10

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

2
Rollnumber    Name        SGPA
_____________________________

23323         Pankaj       6.9
23324         Lalit       9.41
23325         Shounak       9.84
23326         Swaraj       9.27
23327         Shravani       9

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
```

```
8. Search SGPA
9. Exit

6
Enter the name to be searched:
Shounak
Record found at location: 3

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

8
Enter the sgpa to be searched: 9.27
Record found at record number 4

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

6
Enter the name to be searched:
abc
Name not found in the records.

Enter the operation:
1. Add student record
2. Display all student records
3. Display student record
4. Sort by roll numbers
5. Sort by Names
6. Search by Names
7. Sort by SGPA
8. Search SGPA
9. Exit

8
Enter the sgpa to be searched: 7
Record not found.

9
```