Program :

*#include* <iostream>

*#include* <string.h>

*using* *namespace* std;

*typedef* *struct* student

{

*int* roll\_no;

*char* name[20];

*float* sgpa;

} stud;

*void* create(stud *s*[20], *int* *n*);

*void* display(stud *s*[20], *int* *n*);

*void* bubble\_sort(stud *s*[20], *int* *n*);

*void* insertionSort(stud *s*[20], *int* *n*);

*void* quick\_sort(stud *s*[20], *int*, *int*);

*int* partition(stud *s*[20], *int*, *int*);

*void* search(stud *s*[20], *int* *n*, *int* *key*);

*int* bsearch(stud *s*[20], *char* *x*[20], *int* *low*, *int* *high*);

*int* main()

{

    stud s[20];

*int* ch, n, key, result;

*char* x[20];

*do*

    {

        cout *<<* "\n 1) Create Student Database ";

        cout *<<* "\n 2) Display Student Records ";

        cout *<<* "\n 3) Bubble Sort ";

        cout *<<* "\n 4) Insertion Sort ";

        cout *<<* "\n 5) Quick Sort ";

        cout *<<* "\n 6) Linear search ";

        cout *<<* "\n 7) Binary search ";

        cout *<<* "\n 8) Exit ";

        cout *<<* "\n Enetr Your Choice:=";

        cin *>>* ch;

*switch* (ch)

        {

*case* 1:

            cout *<<* "\n Enter The Number Of Records:=";

            cin *>>* n;

            create(s, n);

*break*;

*case* 2:

            display(s, n);

*break*;

*case* 3:

            bubble\_sort(s, n);

*break*;

*case* 4:

            insertionSort(s, n);

*break*;

*case* 5:

            quick\_sort(s, 0, n *-* 1);

            cout *<<* "\n"

*<<* "\t"

*<<* "Roll No"

*<<* "\t"

*<<* " Name"

*<<* "\t"

*<<* "sgpa";

*for* (*int* i *=* n *-* 1; i *>=* n *-* 10; i*--*)

            {

                cout *<<* "\n";

                cout *<<* "\t " *<<* s[i].roll\_no *<<* "\t " *<<* s[i].name *<<* "\t " *<<* s[i].sgpa;

            }

*break*;

*case* 6:

            cout *<<* "\n Enter the sgpa which u want to search:=";

            cin *>>* key;

            search(s, n, key);

*break*;

*case* 7:

            cout *<<* "\n Enter the name of student which u want to search:=";

            cin *>>* x;

            insertionSort(s, n);

            result *=* bsearch(s, x, 0, (n *-* 1));

*if* (result *==* *-*1)

            {

                cout *<<* " \n Student name you want to search for is not present ! \n";

            }

*else*

            {

                cout *<<* " \n The student is present :\t" *<<* s[result].name;

            }

*break*;

*case* 8:

*return* 0;

*default*:

            cout *<<* "\n Invalid choice !! Please enter your choice again." *<<* endl;

        }

    } *while* (ch *!=* 8);

}

*void* create(stud *s*[20], *int* *n*)

{

*int* i;

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

    {

        cout *<<* "\n Enter the roll number:=";

        cin *>>* *s*[i].roll\_no;

        cout *<<* "\n Enter the Name:=";

        cin *>>* *s*[i].name;

        cout *<<* "\n Enter the sgpa:=";

        cin *>>* *s*[i].sgpa;

    }

}

*void* display(stud *s*[20], *int* *n*)

{

*int* i;

    cout *<<* "\n"

*<<* "\t"

*<<* "Roll No"

*<<* "\t"

*<<* " Name"

*<<* "\t"

*<<* "sgpa";

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

    {

        cout *<<* "\n";

        cout *<<* "\t " *<<* *s*[i].roll\_no *<<* "\t " *<<* *s*[i].name *<<* "\t " *<<* *s*[i].sgpa;

    }

}

*//bubble sort to sort in ascending order on roll number void bubble\_sort(stud s[20],int n)*

*void* bubble\_sort(stud *s*[20], *int* *n*)

{

*int* i, j;

    stud temp;

    {

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

        {

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

            {

*if* (*s*[j].roll\_no *>* *s*[j *+* 1].roll\_no)

                {

                    temp *=* *s*[j];

*s*[j] *=* *s*[j *+* 1];

*s*[j *+* 1] *=* temp;

                }

            }

        }

    }

}

*// insertion sort to sort on names in ascending order void insertionSort(stud s[20], int n)*

*void* insertionSort(stud *s*[20], *int* *n*)

{

*int* i, j;

    stud key;

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

    {

        key *=* *s*[i];

        j *=* i *-* 1;

*/\* Move elements of arr[0..i-1], that are greater than key, to one position ahead of their current position \*/*

*while* (j *>=* 0 *&&* strcmp(*s*[j].name, key.name) *>* 0)

        {

*s*[j *+* 1] *=* *s*[j];

            j *=* j *-* 1;

        }

*s*[j *+* 1] *=* key;

    }

}

*//Quick sort to sort on sgpa*

*void* quick\_sort(stud *s*[20], *int* *l*, *int* *u*)

{

*int* j;

*if* (*l* *<* *u*)

    {

        j *=* partition(*s*, *l*, *u*);

        quick\_sort(*s*, *l*, j *-* 1);

        quick\_sort(*s*, j *+* 1, *u*);

    }

}

*int* partition(stud *s*[20], *int* *l*, *int* *u*)

{

*int* i, j;

    stud temp, v;

    v *=* *s*[*l*];

    i *=* *l*;

    j *=* *u* *+* 1;

*do*

    {

*do*

            i*++*;

*while* (*s*[i].sgpa *<* v.sgpa *&&* i *<=* *u*);

*do*

            j*--*;

*while* (v.sgpa *<* *s*[j].sgpa);

*if* (i *<* j)

        {

            temp *=* *s*[i];

*s*[i] *=* *s*[j];

*s*[j] *=* temp;

        }

    } *while* (i *<* j);

*s*[*l*] *=* *s*[j];

*s*[j] *=* v;

*return* (j);

*// s[l] = s[j];*

*// s[j] = v;*

*// return (j);*

}

*// linear search for sgpa if more than one student having same sgpa print all of them*

*void* search(stud *s*[20], *int* *n*, *int* *key*)

{

*int* i;

    cout *<<* "\n"

*<<* "\t"

*<<* "Roll No"

*<<* "\t"

*<<* " Name"

*<<* "\t"

*<<* "sgpa";

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

    {

*// if (key == s[i].sgpa)*

*if* (*s*[i].sgpa *==* *key*)

        {

            cout *<<* "\n\t " *<<* *s*[i].roll\_no *<<* "\t " *<<* *s*[i].name *<<* "\t " *<<* *s*[i].sgpa;

        }

    }

}

*int* bsearch(stud *s*[20], *char* *x*[20], *int* *low*, *int* *high*)

{

*int* mid;

*while* (*low* *<=* *high*)

    {

        mid *=* (*low* *+* *high*) */* 2;

*if* (strcmp(*x*, *s*[mid].name) *==* 0)

        {

*return* mid;

        }

*else* *if* (strcmp(*x*, *s*[mid].name) *<* 0)

        {

*high* *=* mid *-* 1;

        }

*else*

        {

*low* *=* mid *+* 1;

        }

    }

*return* 0;

}

Output:







