**Assignment-1**

**Name:**

**Roll No:**

**Branch:** S.E. I.T.

**Date:**

**Title: Searching and Sorting**

/\* 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 to find the first ten toppers from a class. (Use Quick sort)

c) 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.

d) Search a particular student according to name using binary search without recursion n. (all the

student records having the presence of search key should be displayed).\*/

* **Program**

#include<iostream>

#include<string>

#define MAX 20

using namespace std;

class stud\_db

{

struct student

{

int roll\_no;

string name;

float sgpa;

}stud[15];

int n;

public:

void accept()

{

//string str;

cout<< "How many records? ";

cin >> n;

cout << "Enter the data -\n";

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

{

cout << "Enter Roll No: ";

cin >> stud[i].roll\_no;

cout << "Enter Name : ";

//flush the stream as there may be a '\n' left in the stream from another part of your program

cin. ignore();

getline(cin, stud[i].name);

cout << "Enter SGPA : ";

cin >> stud[i].sgpa;

}

}

void display()

{

cout<< "R.No.\tName\t\tSGPA\n";

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

{

cout << stud[i].roll\_no << "\t" << stud[i].name << "\t" << stud[i].sgpa << endl;

}

}

//Arrange list of students according to roll numbers in ascending order (Use Bubble Sort)

void bubble\_sort()

{

for(int pass=1; pass<n; pass++) // from index 1 to n-1

{

int swapped=0; //variable to count no. of swaps in a pass

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

{

if(stud[i].roll\_no > stud[i+1].roll\_no) // swap the elements

{

student tmp = stud[i];

stud[i]=stud[i+1];

stud[i+1] = tmp;

swapped++;

}

}

if(swapped==0) // list is sorted

break;

}

cout << "Records Sorted by Roll No using Bubble Sort\n";

display();

}

//Arrange list of students alphabetically. (Use Insertion sort)

void insertion\_sort()

{

for(int pass=1; pass<n; pass++) // from index 1 to n-1

{

student s = stud[pass];

string ele = stud[pass].name;

int i=pass-1; //last element in the sorted part

while((i>=0)&&(stud[i].name > ele)) // shift the elements > than ele one position to the right

{

stud[i+1] = stud[i];

i--;

}

stud[i+1] = s; // place the record at its correct position

}

cout << "Records Sorted by FName using Insertion Sort\n" ;

display();

}

//Arrange list of students to find out first ten toppers from a class.(Use Quick sort)

//Sort in descending order

int partition (int l, int h)

{

int i, j ;

student temp;

float pivot;

pivot = stud[l].sgpa; //set the lowest index element as pivot

i = l + 1;

j=h;

while(i<=j)

{

while(stud[i].sgpa >= pivot)

i++;

while(stud[j].sgpa < pivot)

j--;

if(i<=j)

{

temp=stud[i];

stud[i]=stud[j];

stud[j]=temp;

i++;

j--;

}

}

temp = stud[j];

stud[j]=stud[l];

stud[l]= temp;

return j;

}

void quickSort(int l, int h)

{

int p; //index where pivot element is placed

if (l < h)

{

p=partition(l, h);

quickSort(l, p - 1);

quickSort(p + 1, h);

}

}

void quick()

{

quickSort(0, n-1);

cout<< "R. No.\tFName\tSName\tSGPA\n";

for(int i=0;i<3;i++)

{

cout << stud[i].roll\_no << "\t" << stud[i].name << "\t" << stud[i].sgpa << endl;

}

}

void linear\_search(float key)

{

int i, sr=0;

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

{

if(stud[i].sgpa == key)

{

sr++;

if(sr == 1)

cout<< "Sr.No.\tR.No.\tName\tSGPA\n";

cout << sr << "\t" << stud[i].roll\_no << "\t" << stud[i].name << "\t" << stud[i].sgpa << endl;

}

}

if(sr == 0)

cout << "Record NOT Found!!\n";

}

int b\_search(string key, string side) // 10 20 30 30 30 40 50

{

int mid, result=-1, lb=0, ub=n-1;

while(lb <= ub)

{

mid = (lb+ub)/2;

if(stud[mid].name < key)

lb = mid + 1;

else if(stud[mid].name > key)

ub = mid - 1;

else

{

result = mid;

if(side == "left")

ub = mid - 1;

else

lb = mid + 1;

}

}

return(result);

}

void bsearch(string key)

{

int lind, hind;

insertion\_sort();

lind = b\_search(key, "left");

if(lind == -1)

{

cout << "Record NOT Found!!\n";

return;

}

else

hind = b\_search(key, "right");

cout<< "R.No.\tName\tSGPA\n";

for(int i=lind;i<=hind;i++)

cout << stud[i].roll\_no << "\t" << stud[i].name << "\t" << stud[i].sgpa << endl;

}

};

int main()

{

int ch;

stud\_db s;

string key;

float sg;

do

{

cout << "\nOperations on Database\n";

cout << "1.Accept\n2.Display\n3.Ascending order of Roll No (Use Bubble Sort)\n";

cout << "4.Arrange list of students alphabetically. (Use Insertion sort)\n";

cout << "5.First three toppers using Quick Sort\n";

cout << "6.Search by SGPA (Linear Search)\n";

cout << "7.Search by Name (Binary Search)\n8.Exit\n";

cout << "Enter your choice: ";

cin >> ch;

switch(ch)

{

case 1: s.accept();

break;

case 2: s.display();

break;

case 3: s.bubble\_sort();

break;

case 4: s.insertion\_sort();

break;

case 5: s.quick();

break;

case 6: cout << "Enter SGPA to be searched: ";

cin >> sg;

s.linear\_search(sg);

break;

case 7: cout << "Enter Name to be searched: ";

cin. ignore();

getline(cin, key);

s.bsearch(key);

break;

case 8:

return(0);

}

}while(ch != 8);

}

***OUTPUT***

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 1

How many records? 7

Enter the data -

Enter Roll No: 2330

Enter Name : Sakshi

Enter SGPA : 8.9

Enter Roll No: 2350

Enter Name : Ankita

Enter SGPA : 7.9

Enter Roll No: 2310

Enter Name : Sneha

Enter SGPA : 9.7

Enter Roll No: 2320

Enter Name : Trisha

Enter SGPA : 8.6

Enter Roll No: 2340

Enter Name : Saniya

Enter SGPA : 9.9

Enter Roll No: 2360

Enter Name : Ayesha

Enter SGPA : 9.6

Enter Roll No: 2370

Enter Name : Jyoti

Enter SGPA : 7.6

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 2

R.No. Name SGPA

2330 Sakshi 8.9

2350 Ankita 7.9

2310 Sneha 9.7

2320 Trisha 8.6

2340 Saniya 9.9

2360 Ayesha 9.6

2370 Jyoti 7.6

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 3

Records Sorted by Roll No using Bubble Sort

R.No. Name SGPA

2310 Sneha 9.7

2320 Trisha 8.6

2330 Sakshi 8.9

2340 Saniya 9.9

2350 Ankita 7.9

2360 Ayesha 9.6

2370 Jyoti 7.6

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 4

Records Sorted by FName using Insertion Sort

R.No. Name SGPA

2350 Ankita 7.9

2360 Ayesha 9.6

2370 Jyoti 7.6

2330 Sakshi 8.9

2340 Saniya 9.9

2310 Sneha 9.7

2320 Trisha 8.6

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 5

R. No. FName SName SGPA

2340 Saniya 9.9

2310 Sneha 9.7

2360 Ayesha 9.6

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 6

Enter SGPA to be searched: 8.9

Sr.No. R.No. Name SGPA

1 2330 Sakshi 8.9

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 6

Enter SGPA to be searched: 5.8

Record NOT Found!!

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 7

Enter Name to be searched: Ankita

Records Sorted by FName using Insertion Sort

R.No. Name SGPA

2350 Ankita 7.9

2360 Ayesha 9.6

2370 Jyoti 7.6

2330 Sakshi 8.9

2340 Saniya 9.9

2310 Sneha 9.7

2320 Trisha 8.6

R.No. Name SGPA

2350 Ankita 7.9

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 7

Enter Name to be searched: Samiksha

Records Sorted by FName using Insertion Sort

R.No. Name SGPA

2350 Ankita 7.9

2360 Ayesha 9.6

2370 Jyoti 7.6

2330 Sakshi 8.9

2340 Saniya 9.9

2310 Sneha 9.7

2320 Trisha 8.6

Record NOT Found!!

Operations on Database

1.Accept

2.Display

3.Ascending order of Roll No (Use Bubble Sort)

4.Arrange list of students alphabetically. (Use Insertion sort)

5.First three toppers using Quick Sort

6.Search by SGPA (Linear Search)

7.Search by Name (Binary Search)

8.Exit

Enter your choice: 8

--------------------------------------------------------------------------