Name : Esha Patel
Roll number : 19BCE169
Course : Data Structure and Algorithm
Course name : 2CS301
Date : November 20, 2020

1) Comprehensive assignment Definition:

- In a school, students of 5th Grade are going for a picnic. For a particular game between 10 players, they need to be organized in the ascending order of their height. Teacher selects a one random student out of 10. That student acts as a mediator, all students having height less than mediator goes on left and rest on his right. The same process repeats again between the left and right group. The process continues and will stop when all the players are in ascending order of their height. Signify which sorting algorithm can be helpful to design this model and how? What additional functionality can you add to this. Implement the given model.

2) Program file.

- Name of file: 19BCSE169_DSA_Comrephensive Assignment.c
- File purpose: Sort Student in the ascending order by their height using quick sort algorithm.

3) Program Code:

```
//Esha Patel
//19BCE169
//DSA Comprehensive Assignment
//2CS301

/*PROGRAM DEFINITION ----------> In a school, students of 5th Grade are going
for a picnic.
For a particular game between 10 players, they needs to be organized in the
ascending order of their height.
Teacher selects a one random student out of 10.
That student acts as a mediator, all students having height less than mediator
goes on left and rest on his right.
The same process repeats again between the left and right group.
The process continues and will stop when all the players are in ascending
order of their height.
Implement the given model.  */
```

```
/*QUESTION--------->Signify which sorting algorithm can be helpful to design
this model and how.
  ANSWERE--------->QUICK sorting algorithm can be helpful here.....
  QUICKSORT is a divide and conquer algorithm. It works by selecting a "pivot"
element from the array
  and partitioning the other element into two sub-arrays , according to
whether they are less than or greater than the pivot.
  The sub-array are then sorted recursively */

/*QUESTION---------->What additional functionality can you add to this.
  ANSWERE----------->first off all I sort student into Ascending order by them
height.
  I give option to user that add more student or remove any student and then
sort into Ascending Order and display them
  Then I give option for searching that you can search student position by
name or you can search student name and height by position*/

#include<stdio.h>
#include<stdlib.h>
#include<string.h>

typedef struct student              //Declaring structure STUDENT
    {
        char name[100];             //structure members that store student
name, height and position rank
        float height;
        int rank;
    }stu;

 stu s[30];

void quick_sort(int p, int q);                  //function declaration
int partion( int p, int r);
void swap(stu *s1, stu *s2);
void print_data(int n);
void search(int n);
void search_data(int n);
void delete_detail(struct student s[],int n);
void delete_name(struct student s[],int n);

void quick_sort(int p, int q)                   //Recursive Function for quick
sort
{
    int j;
    if (p < q)
    {
        j = partion(p, q);
        quick_sort( p, j-1);
```

```
        quick_sort(j+1, q);
    }
}
int partion( int p, int r)                              //function for
partitioning array element into two sub-arrays
{
    int pivotIndex = p + rand()%(r - p + 1);            //generates a random
number as a pivot
    int pivot;
    int i = p - 1;
    int j;
    pivot = s[pivotIndex].height;
    swap(&s[pivotIndex], &s[r]);
    for (j = p; j < r; j++)
    {
        if (s[j].height < pivot)
        {
            i++;
            swap(&s[i], &s[j]);
        }

    }
    swap(&s[i+1], &s[r]);
    return i + 1;
}

void swap(stu *s1, stu *s2)                             //function for
swapping two structure
{
    stu temp;
    temp = *s1;
    *s1 = *s2;
    *s2 = temp;
}
void print_data(int n)                                  //function for print
student height in Ascending order
{
    int i=0,j;
        printf("\nAscending order of student heights ");
        printf("\n---------------------------");
        printf("\nRANK\tNAME\tHEIGHT");
        printf("\n---------------------------\n");
        do{
            for(j=0;j<n;j++)
            {
                s[i].rank=i+1;
                printf("%d \t",i+1);
                printf("%s \t",s[j].name);
```

```
                printf("%.2f \n",s[j].height);
                i=i+1;
            }
        }while(i<n);
}
void add(struct student s[],int n)                      //function for add
more student
{
    int i,sal;
    char ename;
        printf("\n Enter Student Name:");
        scanf("%s",s[n].name);
        printf("\n Enter Height:");
        scanf("%f",&s[n].height);
        quick_sort(0, n);
        print_data(n+1);
}

void search(int n)                                          //function for
search student name by position
{
    int eid,i,j,p=1,g;
    printf("Enter Student Position to be Searched:");
    scanf("%d",&eid);
    for(i=0;i<n;i++)
    {
        if(s[i].rank==eid)
        {
            p++;
            g=i;
        }
    }
    if(p==2)
    {
        printf("\nStudent Name:%s",s[g].name);
        printf("\nStudent Height:%.2f",s[g].height);
        printf("\n");
    }
    else
    {
        printf("No Such Name found!");
    }
}
void search_data(int n)                             //function for search
student position by name
{
    int i;
    char name[100];
```

```
    printf("Name of Student to be searched :");
    scanf("%s",name);
    for(i=0;i<n;i++)
        {
            if(!strcmp(name,s[i].name))
            {
                printf("\nStudent Position");
                printf("%d ",s[i].rank);
                printf("\nStudent Name");
                printf("%s ",s[i].name);
                printf("\nStudent Height");
                printf(".2%f ",s[i].height);
            }
        }
}
void delete_detail(struct student s[],int n)                    //function for
remove student by position
{
    int f,t=1,eid,i;
    printf("Enter student position:");
    scanf("%d",&eid);
        for(i=0;i<n;i++)
        {
                                        //to search id to be deleted
            if(s[i].rank==eid)
            {
                f=i;
                t++;
            }
                        else
                                        continue;
        }
        if(t=2)
        {
            for(i=f;i<n;i++)
            {
                s[i]=s[i+1];
            }
    }
    else
    printf("Student not found!");
     print_data(n-1);
}
void delete_name(struct student s[],int n)                      //function for
remove student by name
{
    int f,t=1,eid,i;
    char dlname[10];
```

```
    printf("Enter student name:");
    scanf("%s",dlname);
    for(i=0;i<n;i++)
    {
                                        //to search id to be deleted
        if(!strcmp(dlname,s[i].name))
        {
            f=i;
            t++;
        }
        else
            continue;
    }
        if(t=2)
        {
            for(i=f;i<n;i++)
            {
                s[i]=s[i+1];
            }
        }
        else
        {
            printf("Student not found!");
        }
         print_data(n-1);
}
/*****************************************************************main
function*/
int main()
{
    int i,j,n,opt,DL;
    printf("Enter Total Number of the Students: ");
    scanf("%d",&n);
    printf("Enter the NAME and HEIGHT of the student \n");
    for(i=0;i<n;i++)
    {
        printf("\nName of Student %d = ",i+1);
        scanf("%s",s[i].name);
        printf("Height of student %d =",i+1);
        scanf("%f",&s[i].height);
        printf("\n");
            //arr[i]=s[i].height;
    }
        quick_sort(0, n - 1);
        print_data(n);
    while(1)
         {
            int ch,ex=0;
```

```
        printf("\n1 Add more Student ");
        printf("\n2 Search Student");
        printf("\n3 Remove Student");
        printf("\n4 Exit");
        printf("\n---->");
        scanf("%d",&ch);
        switch(ch)
        {
            case 1:
                add(s,n);
                break;
            case 2:
                printf("\n1) Search Student by them POSITION");
                printf("\n2) search Student by NAME");
                printf("\n----->");
                scanf("%d",&opt);
                switch(opt)
                {
                    case 1:
                        search(n);
                        break;
                    case 2:

                        search_data(n);
                        break;
                    default:
                        printf("\nNO SUCH INPUT!");
                        break;
                }
              break;
              case 3:
                      printf("\n1) Remove Student by Position");
                      printf("\n2) Remove Student by Name");
                      printf("\n---->");
                      scanf("%d",&DL);
                      switch(DL)
                      {
                           case 1:
                               delete_detail(s,n);
                               break;
                           case 2:
                               delete_name(s,n);
                               break;
                           default:
                               printf("NO SUCH INPUT!");
                               break;
                      }
                break;
```

```
                case 4:
                        ex++;
                        break;
                default:
                        printf("NO SUCH INPUT!! ");
        }
    if(ex==1)
    return 0;
    }
}
```

4) Output Screen Shots

```
"D:\Turbo C++ 3.2\Turbo C++ 3.2.msi\D_S_A\bin\Debug\D_S_A.exe"

Enter Total Number of the Students: 6
--------------------------------------------
Enter the NAME and HEIGHT of the student

Name of Student 1 = esha
Height of student 1 =145.57


Name of Student 2 = rushi
Height of student 2 =180.0


Name of Student 3 = mitul
Height of student 3 =167.43


Name of Student 4 = riya
Height of student 4 =130.0


Name of Student 5 = sikha
Height of student 5 =189.37


Name of Student 6 = diya
Height of student 6 =160
```

```
"D:\Turbo C++ 3.2\Turbo C++ 3.2.msi\D_S_A\bin\Debug\D_S_A.exe"


Ascending order of student heights
---------------------------
RANK    NAME    HEIGHT
---------------------------
1       riya    130.00
2       esha    145.57
3       diya    160.00
4       mitul   167.43
5       rushi   180.00
6       sikha   189.37

1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->1

 Enter Student Name:harvi

 Enter Height:162.32

Ascending order of student heights
---------------------------
RANK    NAME    HEIGHT
---------------------------
1       riya    130.00
2       esha    145.57
3       diya    160.00
4       harvi   162.32
5       mitul   167.43
6       rushi   180.00
7       sikha   189.37

1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->
```

```
"D:\Turbo C++ 3.2\Turbo C++ 3.2.msi\D_S_A\bin\Debug\D_S_A.exe"

Ascending order of student heights
---------------------------
RANK    NAME    HEIGHT
---------------------------
1       riya    130.00
2       esha    145.57
3       diya    160.00
4       harvi   162.32
5       mitul   167.43
6       rushi   180.00
7       sikha   189.37

1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->2

1) Search Student by them POSITION
2) search Student by NAME
----->1
Enter Student Position to be Searched:9
No Such Name found!
1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->2

1) Search Student by them POSITION
2) search Student by NAME
----->2
Name of Student to be searched :harvi

Student Position: 4
Student Name: harvi
Student Height: 162.32
1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->
```

```
"D:\Turbo C++ 3.2\Turbo C++ 3.2.msi\D_S_A\bin\Debug\D_S_A.exe"

1) Remove Student by Position
2) Remove Student by Name
---->1
Enter student position:2

Ascending order of student heights
--------------------------
RANK    NAME    HEIGHT
--------------------------
1       riya    130.00
2       diya    160.00
3       harvi   162.32
4       mitul   167.43
5       rushi   180.00
6       sikha   189.37

1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->3

1) Remove Student by Position
2) Remove Student by Name
---->2
Enter student name:mitul

Ascending order of student heights
--------------------------
RANK    NAME    HEIGHT
--------------------------
1       riya    130.00
2       diya    160.00
3       harvi   162.32
4       rushi   180.00
5       sikha   189.37
6               0.00

1 Add more Student
2 Search Student
3 Remove Student
4 Exit
---->4

Process returned 0 (0x0)   execution time : 701.688 s
Press any key to continue.
```