**Agnim Gupta**

**2028083**

**A-23,CSSE**

**Question 1**

#include <stdio.h>

#include<stdlib.h>

#include <conio.h>

void merge(int arr[], int l, int m, int r)

{

    int i, j, k;

    int n1 = m - l + 1;

    int n2 = r - m;

    int L[n1], R[n2];

    for (i = 0; i < n1; i++)

    L[i] = arr[l + i];

    for (j = 0; j < n2; j++)

    R[j] = arr[m + 1 + j];

    i = 0;

    j = 0;

    k = l;

    while (i < n1 && j < n2)

    {

        if (L[i] <= R[j])

        {

            arr[k] = L[i];

            i++;

        }

        else

        {

            arr[k] = R[j];

            j++;

        }

            k++;

        }

        while(i < n1)

        {

            arr[k] = L[i];

            i++;

            k++;

        }

        while(j < n2)

        {

            arr[k] = R[j];

            j++;

            k++;

        }

}

void mergeSort(int arr[], int l, int r)

{

    if(l < r)

    {

        int m = l + (r - l) / 2;

        mergeSort(arr, l, m);

        mergeSort(arr, m + 1, r);

        merge(arr, l, m, r);

    }

}

//QuickSort:

int partition(int a[], int start, int end)

{

    int pivot = a[end];

    int i = (start - 1);

    for (int j = start; j <= end - 1; j++)

    {

        if (a[j] < pivot)

        {

            i++;

            int t = a[i];

            a[i] = a[j];

            a[j] = t;

        }

    }

    int t = a[i+1];

    a[i+1] = a[end];

    a[end] = t;

    return (i + 1);

}

void quickSort(int a[], int start, int end){

    if (start < end){

        int p = partition(a, start, end);

        quickSort(a, start, p - 1);

        quickSort(a, p + 1, end);

    }

}

//Printing Array:

void printArray(int a[], int n){

    for (int i = 0; i < n; i++)

        printf("%d ", a[i]);

}

int main(){

    printf("1.Quick Sort\n2.Merge Sort\n0.Exit\nEnter your choice:");

    int ch,arr[10];

    scanf("%d",&ch);

    for(int i=0;i<10;i++){

        arr[i] = rand();

    }

    switch (ch)

    {

    case 1:

        mergeSort(arr,0, 10-1);

        printf("\nArray after merge sort: ");

        break;

    case 2:

        quickSort(arr,0, 10-1);

        printf("\nArray after quick sort: ");

        break;

    default:

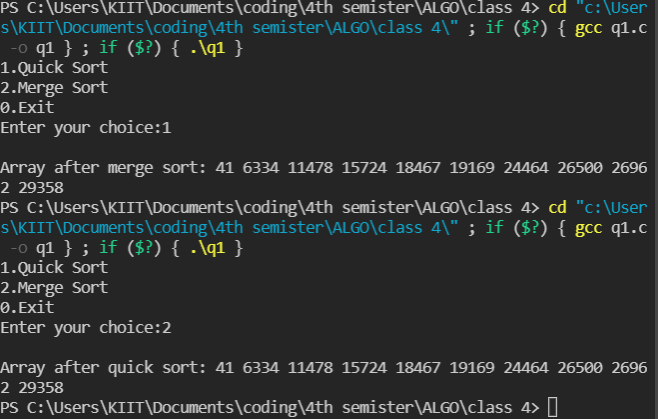
        printf("Wrong Input!");

    }

    printArray(arr,10);

}

**Output**

****