**Name : Tanishq Thuse**

**Branch : SY-CS(AI)**

**Div : B**

**Roll No. : 60**

**Subject : ADS Assignment-1**

**Title : Sorting Techniques**

**Q1) Bubble Sort**

Code:

#include <iostream>

#include <stdio.h>

using namespace std;

#include <cstring>

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

// int n = sizeof(a)/sizeof(a[0]);

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

cout<<a[i]<<" ";

}

cout<<endl;

}

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

int comparisons = 0;

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

int flag = 0;

//IMP : j < n-1-i (instead of j<n-1) is a very important condition otherwise the array is comparing the sorted part of array and it increases the number of comparisons from 10 -> 20 in worst case

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

if(a[j] > a[j+1]){

int temp = a[j];

a[j] = a[j+1];

a[j+1] = temp;

flag = 1; //make sure it works in O(n) for sorted array

}

comparisons++;

}

if(flag == 0){

//it means array is already sorted

break;

}

}

cout<<"Sorted array is : ";

print(a,n);

cout<<"Total comparisons : "<<comparisons<<endl;

}

int main(){

int n;

cout<<"Enter size of array"<<endl;

cin>>n;

int a[n];

cout<<"Enter no. of cases to execute"<<endl;

int cases;

cin>>cases;

int count = 1;

while(cases>0){

cout<<"Case "<<count++<<endl;

cout<<"Enter elements of array"<<endl;

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

cin>>a[i];

}

cout<<"Original array is : ";

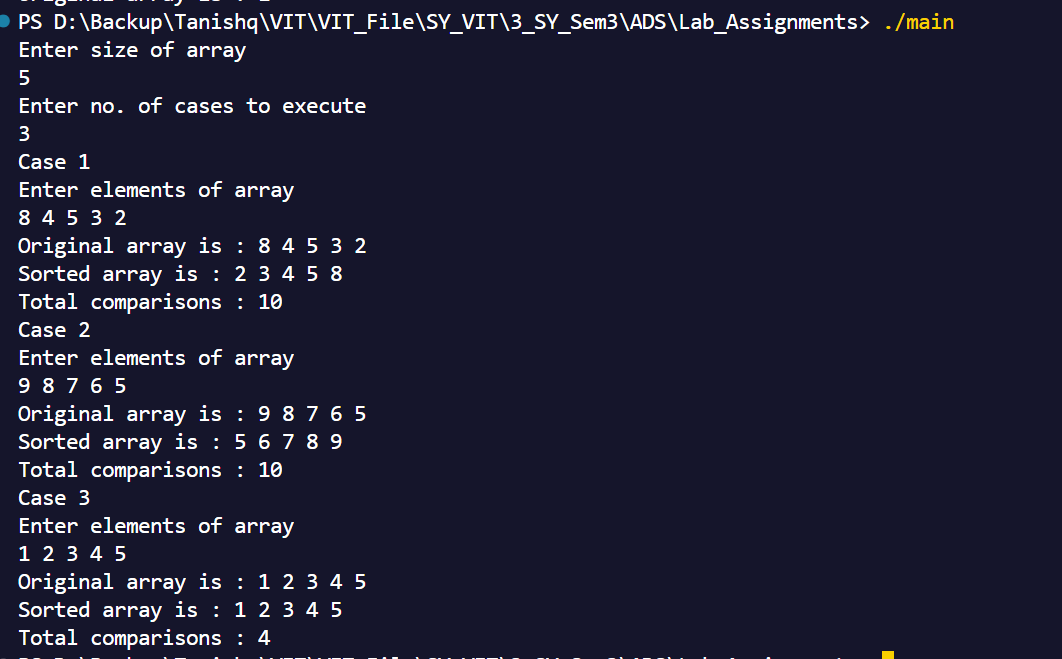
print(a,n);

BubbleSort(a,n);

cases--;

}}

Output :



**Q2) Insertion Sort**

**Code :**

#include<stdio.h>

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

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

printf("%d ",a[i]);

}

printf("\n");

}

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

for(int i=1; i<n; i++){

int key = a[i];

for(int j=i-1; j>=0; j--){

if(key < a[j]){

// int temp = key;

// key = a[j];

// a[j] = temp;

int temp = a[j+1];

a[j+1] = a[j];

a[j] = temp;

}

}

}

printf("Array after Insertion Sort : ");

print(a,n);

}

int main(){

int n;

printf("Enter the size of arrays\n");

scanf("%d",&n);

int a[n];

printf("Enter the elements in array\n");

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

scanf("%d",&a[i]);

}

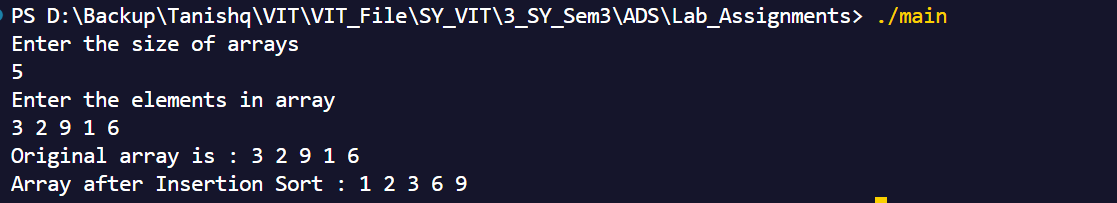
printf("Original array is : ");

print(a,n);

InsertionSort(a,n);

}

Output :



**Q3) Quick Sort**

Code :

#include<iostream>

using namespace std;

void takeInput(int arr[], int n){

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

cin>>arr[i];

}

}

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

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

cout<<a[i];

}

cout<<endl;

}

int partition(int lb, int ub, int arr[]){

int pivot = lb;

//lb is lower bound, ub is upper bound

j = ub;

}

void QuickSort(int LowerBound, int UpperBound, int arr[]){

if(LowerBound < UpperBound){

int k = partition(LowerBound, UpperBound, arr);

//k returns the index where the pivot is placed

QuickSort(LowerBound, k-1, arr);

QuickSort(k+1, UpperBound, arr);

}

}

int main(){

int n;

cout<<"Enter the number of integers"<<endl;

cin>>n;

int arr[n];

cout<<"Enter the elements in array"<<endl;

takeInput(arr,n);

cout<<"Before Quick Sort"<<endl;

print(arr,n);

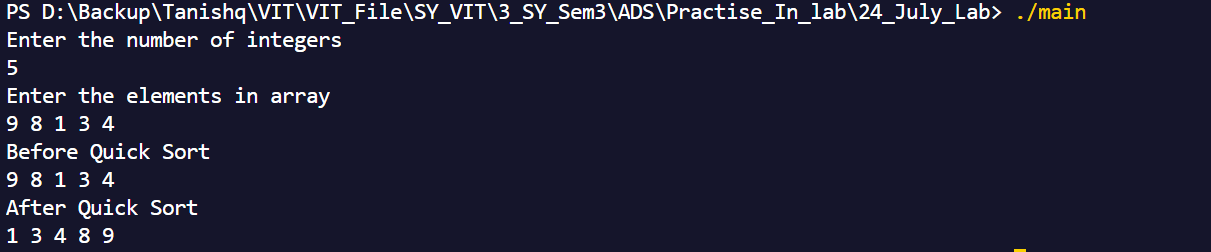
//Applying Quick Sort

QuickSort(0,n-1,arr);

cout<<"After Quick Sort"<<endl;

print(arr,n);

}

Output :

**Q4) Merge Sort**

Code :

#include<iostream>

using namespace std;

void takeInput(int arr[], int n){

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

cin>>arr[i];

}

}

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

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

cout<<a[i]<<" ";

}

cout<<endl;

}

void merge(int arr[], int lb, int mid, int ub){

int n1 = mid - lb + 1;

int n2 = ub - mid;

int left[n1], right[n2];

for(int i = 0; i < n1; i++)

left[i] = arr[lb + i];

for(int j = 0; j < n2; j++)

right[j] = arr[mid + 1 + j];

int i = 0, j = 0, k = lb;

while(i < n1 && j < n2){

if(left[i] <= right[j]){

arr[k] = left[i];

i++;

} else {

arr[k] = right[j];

j++;

}

k++;

}

while(i < n1){

arr[k] = left[i];

i++;

k++;

}

while(j < n2){

arr[k] = right[j];

j++;

k++;

}

}

void mergeSort(int arr[], int lb, int ub){

if(lb < ub){

int mid = lb + (ub - lb) / 2;

mergeSort(arr, lb, mid);

mergeSort(arr, mid + 1, ub);

merge(arr, lb, mid, ub);

}

}

int main(){

int n;

cout<<"Enter the number of integers"<<endl;

cin>>n;

int arr[n];

cout<<"Enter the elements in array"<<endl;

takeInput(arr,n);

cout<<"Before Merge Sort"<<endl;

print(arr,n);

//Applying Merge Sort

mergeSort(arr, 0, n-1);

cout<<"After Merge Sort"<<endl;

print(arr,n);

}

Output :

