BRACT’s

Vishwakarma Institute of Information Technology, Pune

**Practical Implementation Sheet**

| **Department:** IT | **Semester:** IV | **Academic Year:** 2024-25 | **Practical No: 4** |
| --- | --- | --- | --- |
| **Class/ Division/ Batch: SY (B)- B3** | | **Roll no: 70** | |
| **Course:** Data Structures and Analysis of Algorithms | | **Name of Student**: Anushka Kadam | |

**Aim:** Divide and Conquer: Implement a program to find minimum and maximum elements from a given list using Divide and Conquer strategy.

**Code:**

#include <iostream>

using namespace std;

void merge(int arr[], int left, int mid, int right)

{

int n1 = mid - left + 1;

int n2 = right - mid;

int leftArr[n1], rightArr[n2];

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

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

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

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

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

while (i < n1 && j < n2)

{

if (leftArr[i] <= rightArr[j])

{

arr[k] = leftArr[i];

i++;

}

else

{

arr[k] = rightArr[j];

j++;

}

k++;

}

while (i < n1)

{

arr[k] = leftArr[i];

i++;

k++;

}

while (j < n2)

{

arr[k] = rightArr[j];

j++;

k++;

}

}

void mergeSort(int arr[], int left, int right)

{

if (left < right)

{

int mid = left + (right - left) / 2;

mergeSort(arr, left, mid);

mergeSort(arr, mid + 1, right);

merge(arr, left, mid, right);

}

}

void printArray(int arr[], int size)

{

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

cout << arr[i] << " ";

cout << endl;

}

int main()

{

int size;

cout << "Enter the number of elements: ";

cin >> size;

int arr[size];

cout << "Enter " << size << " elements: ";

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

{

cin >> arr[i];

}

cout << "Original array: ";

printArray(arr, size);

mergeSort(arr, 0, size - 1);

cout << "Sorted array: ";

printArray(arr, size);

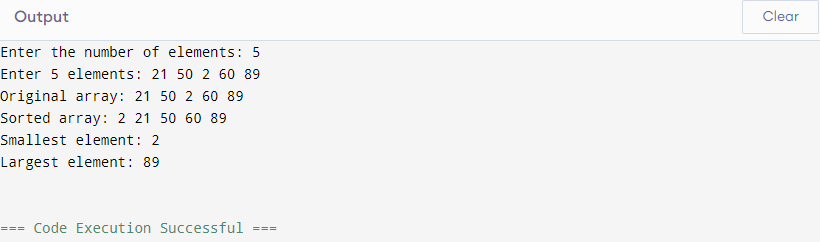
cout << "Smallest element: " << arr[0] << endl;

cout << "Largest element: " << arr[size - 1] << endl;

return 0;

}

**Output:**

****