**COURSE CODE: DJS22ITC403 DATE:12/10/2023**

**COURSE NAME:** **Design and Analysis of Algorithms CLASS: I1-Batch1**

**NAME : ANISH SHARMA SAP ID: 60003220045**

**Experiment No.2**

# Aim: Merge Sort and Min\_Max using Divide and Conquer

# Analysis of Merge Sort

# Code:

#include <stdio.h>

#include <stdlib.h>

#include <math.h>

int ctr = 0;

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++;

        ctr++;

    }

    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);

        ctr++; // Counting the divide-and-conquer steps

    }

}

void printArray(int A[], int size)

{

    int i;

    for (i = 0; i < size; i++)

        printf("%d ", A[i]);

    printf("\n");

}

int main()

{

    int arr[] = {6, 5, 12, 10, 9, 1};

    int arr\_size = sizeof(arr) / sizeof(arr[0]);

    // printf("Given array is \n");

    // printArray(arr, arr\_size);

    mergeSort(arr, 0, arr\_size - 1);

    // printf("\nSorted array is \n");

    // printArray(arr, arr\_size);

    printf("Size of input is %d | ", arr\_size);

    printf("Number of divide-and-conquer steps: %d | ", ctr);

    printf("Logarithmic value of array size: %d", (int)log2(arr\_size));

    return 0;

}

# Output:

# 

# Analysis of Min Max

# Code:

#include <stdio.h>

int main()

{

    int arr[5] = {3, 1, 5, 7, 2};

    int n=5;

    int max = arr[0];

    int min = arr[0];

    int ctr = 0;

    for (int i = 1; i < 5; i++)

    {

        ctr++;

        if (min > arr[i])

        {

            min = arr[i];

        }

    }

    for (int i = 1; i < 5; i++)

    {

        ctr++;

        if (max < arr[i])

        {

            max = arr[i];

        }

    }

    printf("Input size is %d | ", n);

    printf("No. of comparison is %d | ", ctr);

    printf("Time Complexity: O(%d)\n",n);

    return 0;

}

# Output:

# 

**Conclusion:** We implemented analysis of merge sort and analysis of min max in this experiment using divide and conquer method.