**CHAROTAR UNIVERSITY OF SCIENCE &**

**TECHNOLOGY**

**DEVANG PATEL INSTITUTE OF ADVANCE TECHNOLOGY & RESEARCH**

**Computer Science & Engineering**

**NAME: PARTH NITESHKUMAR PATEL**

**ID: 19DCS098**

**SUBJECT: DESIGN AND ANALYSIS OF**

**ALGORITHM**

**CODE: CS 351**

**PRACTICAL-3.1**

**AIM:**

ANALYSE THE DIVIDE AND CONQUER TECHNIQUE

Implement and perform analysis of worst case of Merge Sort and Quicksort.

Compare both algorithms.

**PROGRAM CODE:**

**MERGE SORT:**

#include<iostream>

using namespace std;

int counter=0;

void merge(int arr[], int left, int middle, int right){

    int sizeLeft = middle - left +  1;

    int sizeRight = right - middle;

    int leftArray[sizeLeft], rightArray[sizeRight];

    for(int i=0;i<sizeLeft;i++){

        leftArray[i] = arr[left + i];

    }

    for(int i=0;i<sizeRight;i++){

        rightArray[i] = arr[middle+1+i];

    }

    int i=0;

    int j=0;

    int k=left;

    while(i<sizeLeft && j< sizeRight){

        if(leftArray[i] <= rightArray[j]){

            arr[k] = leftArray[i];

            i++;

        } else {

            arr[k] = rightArray[j];

            j++;

        }

        k++;

    }

    while(i<sizeLeft){

        arr[k] = leftArray[i];

        i++;

        k++;

    }

    while(i<sizeRight){

        arr[k] = rightArray[j];

        j++;

        k++;

    }

}

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

    counter++;

    if (right > left){

        int middle = (left + right) / 2;

        mergeSort(arr, left, middle);

        mergeSort(arr, middle+1, right);

        merge(arr, left, middle, right);

    }

}

int main(){

    int size;

    cout << "Enter the input size: ";

    cin >> size;

    int arr[size];

    cout << "Enter the values of array: "<< endl;

    for(int i=0;i<size;i++){

        cin >> arr[i];

    }

    mergeSort(arr,0,size);

    cout << "The sorted array : "<< endl;

    for(int i=0;i<size;i++){

        cout << arr[i] << " ";

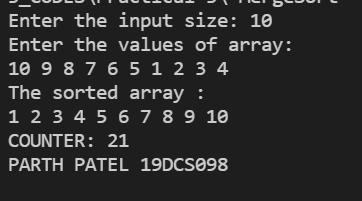
    }

    cout<<"\nCOUNTER: "<<counter<<endl;

    cout<<"PARTH PATEL 19DCS098"<<endl;

}

**OUTPUT:**



**QUICK SORT:**

#include <iostream>

using namespace std;

int counter=0;

int partition(int arr[], int low, int high){

    int pivot = arr[high];

    int i = low-1;

    for(int j=low;j<=high-1;j++){

        if(arr[j] <= pivot){

            i++;

            swap(arr[i],arr[j]);

        }

    }

    swap(arr[i+1],arr[high]);

    return i+1;

}

void quickSort(int arr[], int low, int high){

   counter++;

    if(low < high){

        int pivotIndex = partition(arr,low,high);

        quickSort(arr,low,pivotIndex-1);

        quickSort(arr,pivotIndex+1,high);

    }

}

int main(){

    int size;

    cout << "Enter the input size : ";

    cin >> size;

    int arr[size];

    cout << "Enter the values : "<< endl;

    for(int i=0;i<size;i++){

        cin >> arr[i];

    }

    quickSort(arr,0,size-1);

    cout << "The sorted array : "<< endl;

    for(int i=0;i<size;i++){

        cout << arr[i] << " ";

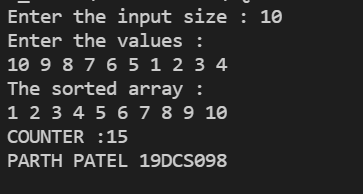
    }

    cout<<"COUNTER :"<<counter<<endl;

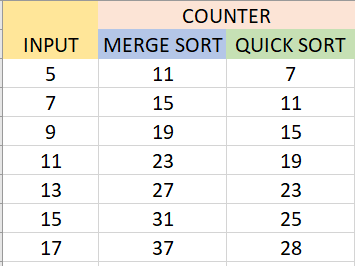
    cout<<"PARTH PATEL 19DCS098"<<endl;

}

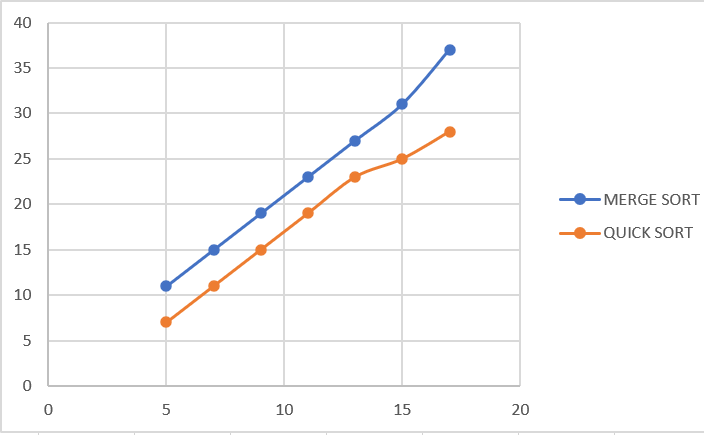
**OUTPUT:**



**ANALYSIS TABLE:**



**GRAPH:**



**CONCLUSION:**

In this practical, we have learnt about how quick sort and merge sort are behaving in the worst case.

**PRACTICAL-3.2**

**AIM:**

Implement the program to find X^Y using divide and conquer strategy and print

the number of multiplications required to find X^Y.

**PROGRAM CODE:**

#include<iostream>

using namespace std;

static int counter = 0;

int  power(int x, unsigned int y) {

    if (y == 0)

        return 1;

    else if (y % 2 == 0){

        counter++;

        return power(x, y / 2) \* power(x, y / 2);

    }

    else{

        counter++;

        return x \* power(x, y / 2) \* power(x, y / 2);

    }

}

int main() {

    int x;

    unsigned int y;

    cout<<"Enter the value of X : ";

    cin>>x;

    cout<<"Enter the value of Y : ";

    cin>>y;

    cout<<"X ^ Y : "<<power(x,y)<<endl;

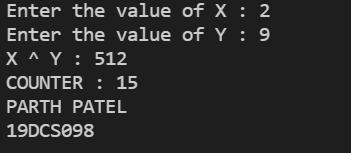
    cout<<"COUNTER : "<<counter<<endl;

    cout<<"PARTH PATEL\n19DCS098"<<endl;

    return 0;

}

**OUTPUT:**



**CONCLUSION:**

In this practical, we learned to implement the power function using divide and conquer strategy.