Jawahar Education Societys Annasaheb Chudaman Patil College of Engineering, Kharghar, Navi Mumbai

NAME: PRIYUSH BHIMRAO KHOBRAGADE

PRN NO: 211112018

Roll No: 52

SUBJECT: Analysis of Algorithms Lab

Merge Sort using Divide and Conquer

EXPERMINT: 02

PAGE NO.:
DATE.: / / 20

Experiment No-02

- Aim :- 'C' Program for merge sort using Divide and conque.

- Hardware / Software Required :- Turbo 'C'

- Theory :-

'Merge Sort'

Merge sort is a recursive algorithm that if it has one element, it is sorted by definition (the base case). if the list has more than one element, we split the list and recursively invoke a merge sort on both halves. once the two halves are sorted, the operation, called a merge, is performed. Mergining is the process of taking two smaller sorted lists and combining them together into a a single, sorted, new list.

- Algorithm:-

Algorithm mergesort(a, low, high)

1. if (low < high)
2. mid ← (low + high) / 2
3. mergesort (a, low, mid)
4. mergesort (a, mid+1, high)
5. merge (a, low, mid, high).

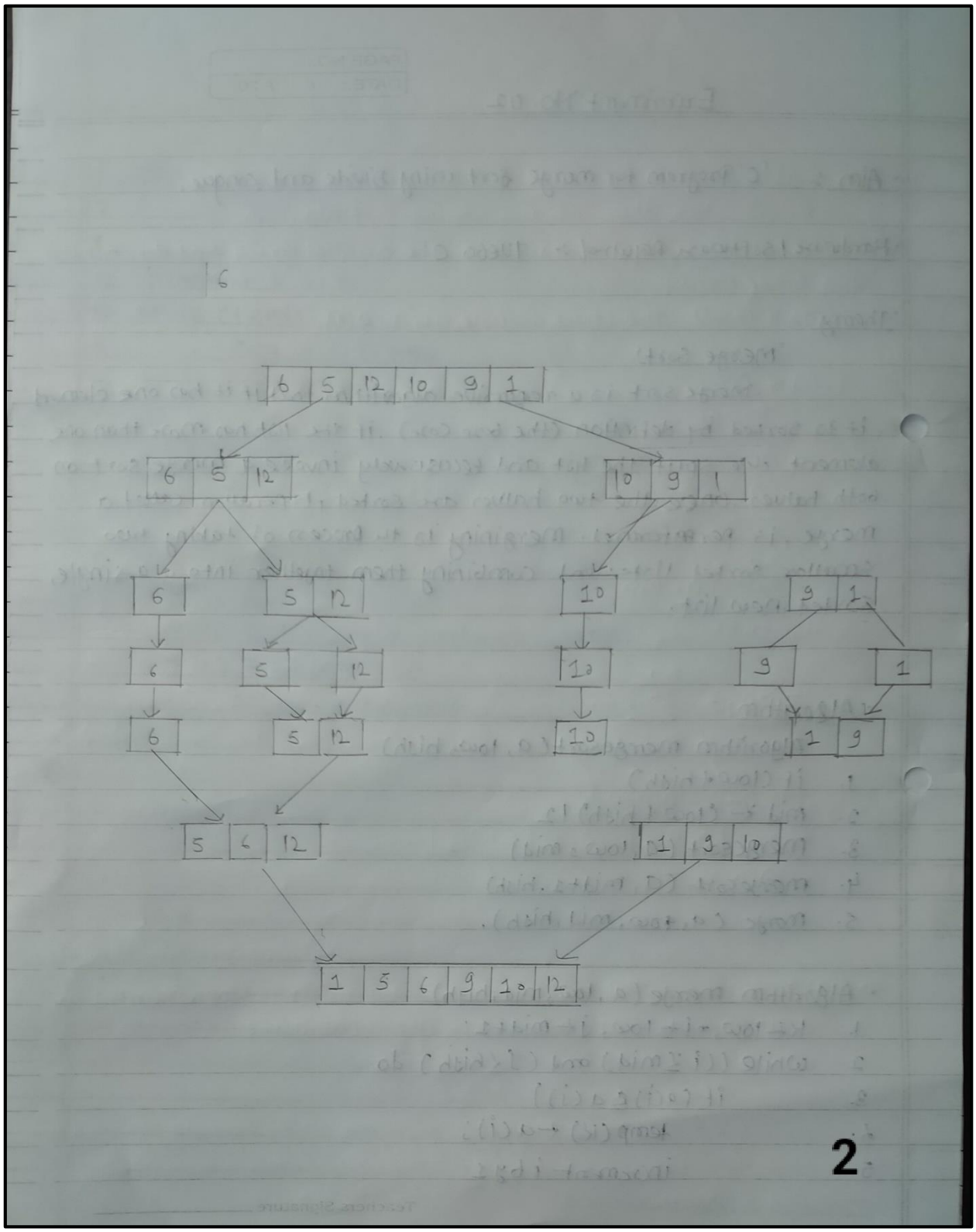
- Algorithm merge (a, low, mid, high)

1. K ← low, i ← low, j ← mid+1
2. while ((i ≤ mid) and (j ≤ high) do
3. if (a(i) ≤ a(j))
4. temp(K) ← a(i);
5. increment i by 1

1

Teachers Signature _____

Merge Sort using Divide and Conquer



PAGE NO.:
DATE.: / / 20

```
else
        temp (k) ← a(j)
        increment j by 1
increment k by 1
while (i ≤ mid) do
        copy a(i) to temp
while (i ≤ mid) do
        copy a(i) to temp
for i ← low to high        // copying element from temp array to original
        a(i) ← temp(i)
```

- Time Complexity:

$T(n) = 0$ if $n=1$

$T(n) = T(n/2) + T(n/2) + cn$ if $n>1$

$T(n) = 2T(n/2) + cn$

where c constant.

Teachers Signature 3

PAGE NO.:
DATE.: / / 20

$T(n) = 2T(n/2) + cn$

$= 2(2.T(n/4) + cn/2) + cn$

$= 2^2.T(n/4) + cn + cn$

$= 2^2(2.T(n/4) + cn + cn$

$= 2^2(2.T(n/8) + n/4) + 2cn$

$= 2^3.T(n/8) + 3cn$

$= 2^3.T(n/2i) + icn$

We know $T(1) = 0$

$\therefore$ substitute $n = 2^i$ $\therefore i = \log_2 n$.

$= n.T(1) + \log_2 n.cn$

$= cn\log_2 n = O(n\log_2 n)$

• Conclusion :- Thus it is observed that in all case the complexity of merge sort is $O(n\log n)$.

Teachers Signature _____

Merge Sort using Divide and Conquer

Input

```
#include<stdio.h>
void merge(int a[],int l,int r);
void mergesort(int a[],int m,int l,int r);
void main()
{
int a[10],n,i,l,r;
printf("\nEnter the range of array:");
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("\nELE %d:",i+1);
scanf("%d",&a[i]);
}
l=0;
r=n-1;
merge(a,l,r);
printf("OUTPUT IS\n");
for(i=0;i<n;i++)
{
printf("%d ",a[i]);
}
}
void merge(int a[],int l,int r)
{
if(l<r)
{
int m;
m=(l+r)/2;
merge(a,l,m);
merge(a,m+1,r);
mergesort(a,m,l,r);
}
}
void mergesort(int a[],int m,int l,int r)
{
int temp[10],i,j,k,b;
i=l;
k=l;
j=m+1;
while(i<=m && j<=r)
{
if(a[i]<=a[j])
{
temp[k]=a[i];
k++;i++;
}
else
{
temp[k]=a[j];
k++;j++;
}
}
if(i>m)
{
for(b=j;b<=r;b++)
{
temp[k]=a[b];
k++;
}
}
else
{
for(b=i;b<=m;b++)
{
temp[k]=a[b];
k++;
}
}
for(b=l;b<=r;b++)
{
a[b]=temp[b];
}
}
```

Output:

```
C:\Users\Rupesh\Documents\merge\bin\Debug\merge.exe

Enter the range of array:5

ELE 1:22

ELE 2:64

ELE 3:47

ELE 4:85

ELE 5:90
OUTPUT IS
22 47 64 85 90
Process returned 5 (0x5)   execution time : 48.739 s
Press any key to continue.
```

Conclusion: Thus, it is observes that in all case the complexity of merge sort is 0 (nlogn).