KIET Group of Institutions, Ghaziabad

Department of Computer Applications

(An ISO – 9001: 2015 Certified & 'A' Grade accredited Institution by NAAC)

Design and Analysis of Algorithm

RCA 352: Session 2020-21

DAA Lab

Experiment-No.1

Objective: Implement the merge sort algorithm to sort the given list of N numbers and plot graph.

Scheduled Date:	Compiled Date:	Submitted Date:
14/08/2020	25/08/2020	26/08/2020

Algorithm:

Merge sort (CLRS)

MERGE(A, p, q, r)

```
1  n1 = q − p + 1
2  n2 = r − q
3  let L[1 .. n1 + 1] and R[1 .. n2 + 1] be new arrays
4  for i = 1 to n1
5       L[i] = A[p + i − 1]
6  for j = 1 to n2
7       R[j] = A[q + j]
8  L[n1 + 1] = ∞
9  R[n2 + 1] = ∞
10 i = 1
11 j = 1
12 for k = p to r
13      if L[i] ≤ R[j]
14           A[k] = L[i]
15           i = i + 1
16      else A[k] = R[j]
17           j = j + 1
```

MERGE-SORT(A, p, r)

```
1  if p < r
2       q = ⌊(p + r)/2⌋
3       MERGE-SORT(A, p, q)
4       MERGE-SORT(A, q + 1, r)
5       MERGE(A, p, q, r)
```

7	1	3	9	4	1	6	8

41

```
PROGRAM FILE: mergeSort.c
#include<stdio.h>
int count=0;
void Merge(int A[],int l,int mid,int h)
{
 int i=l,j=mid+1,k=l;
```

```
 int B[100];
 count++;
 count++;
 while(i<=mid && j<=h)
 {
  if(A[i]<A[j])
  {
  count++;
  B[k++]=A[i++];
  count++;
  }
 else{
 B[k++]=A[j++];
 count++;
  }
 }
 for(;i<=mid;i++)
 {
   B[k++]=A[i];
   count++;
 }
 count++;
 for(;j<=h;j++)
 {
   B[k++]=A[j];
   count++;
 }
 count++;
 for(i=l;i<=h;i++)
 {
   A[i]=B[i];
   count++;
 }
}

void MergeSort(int A[],int l,int h)
{
 int mid;
 count++;
 if(l<h)
 {
 count++;
 mid=(l+h)/2;
 count++;
 MergeSort(A,l,mid);
 count++;
```

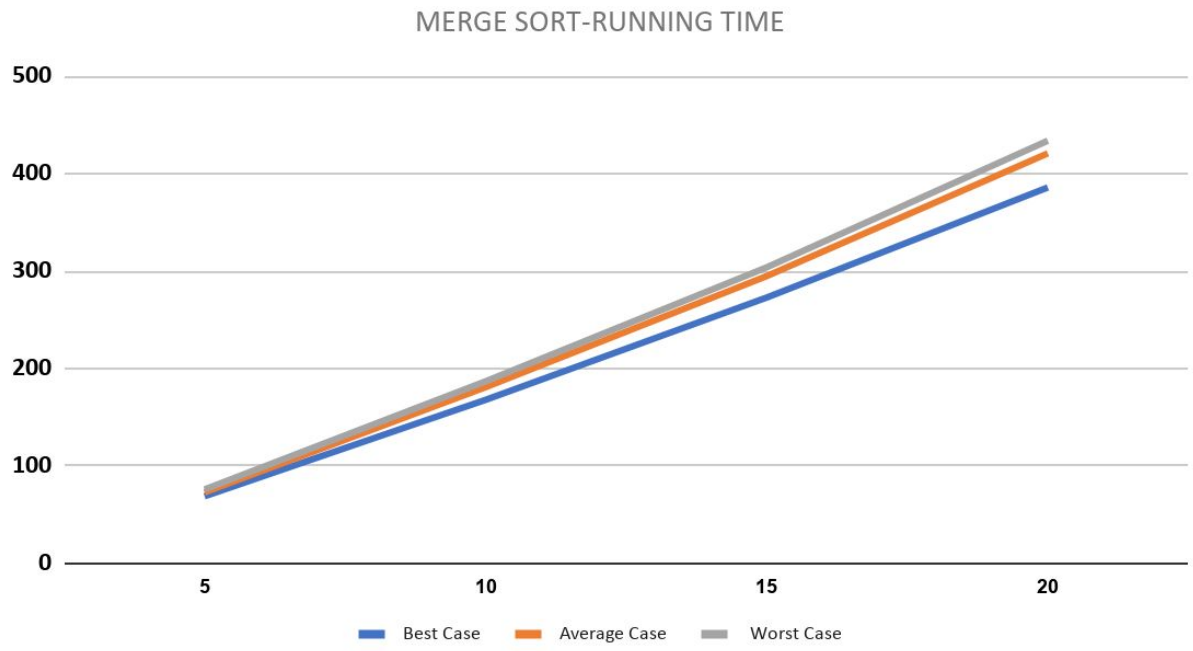
```
 MergeSort(A,mid+1,h);
 count++;
 Merge(A,l,mid,h);
 count++;
 }
}
int main()
{
 int A[100],n,i;
 printf("enter the size of array :");
 scanf("%d",&n);

 printf("enter the elements");
 for(i=0;i<n;i++)
 {
   scanf("%d",&A[i]);
 }
 printf("elements are :\n");
 for(i=0;i<n;i++)
 {
   printf("%d ",A[i]);
 }
 printf("\n");
 MergeSort(A,0,n-1);
 printf("sorted elements are :\n");
 for(i=0;i<n;i++)
 printf("%d ",A[i]);
 printf("\n");
 printf("count =%d",count);
 return 0;
}
```

Output

Input Size	Best Case	Average Case	Worst Case
5	69	73	76
10	168	181	187
15	273	295	304
20	386	421	434

Design and Analysis of Algorithm
RCA 352: Session 2020-21
DAA Lab

Graph:



Conclusion

Case	Running Time : Growth of Function mathematically	Running Time : Growth of Function after observing graph
Best Case	O(nlogn)	O(nlogn)
Average Case	O(nlogn)	O(nlogn)
Worst Case	O(nlogn)	O(nlogn)