Subject :	Software :
	Hardware :
Branch : / Semester :	Page No. / Prog No.

I STATEMENT: Sort a given set of n integer elements using Mergesort method and compute its time complexity. Run the program for varied values of n > 5000, and record the time taken to sort. Plot a graph of the time taken versus n. The elements can be read from a file or can be generated using the random number generator. Demonstrate how the divide-and-conquer method works along with its time complexity analysis: worst-case, best-case and average-case.

HM & CODE:

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
void merge(int arr[], int left, int mid, int right)
{  int n1 = mid - left + 1;
   int n2 = right - mid;
   int L[n1], R[n2];
   for (int i=0; i<n1; i++)
       L[i] = arr[left + i];
   for (int j=0; j<n2; j++)
       R[j] = arr[mid + 1 + j];
   int i=0; j=0; k=left;
   while (i<n1 && j<n2)
  { if (L[i] <= R[j])
```

IVEN	
BTAINED	
KS	

Signature of Faculty Date :	Signature of Student Date :

Subject :	Software :
	Hardware :
Branch : / Semester :	Page No. / Prog No.

PROBLEM STATEMENT

ALGORITHM & CODE :

```
    { arr[K] = L[i];
      i++;
    }
    else
    { arr[K] = R[j];
      j++;
    }
    K++;
  }
  while (i<n1)
  { arr[K] = L[i];
    i++;
    K++;
  }
  while (j<n2)
  { arr[K] = R[j];
    j++;
    K++;
  }
}
void mergeSort(int arr[], int left, int right)
{ if (left<right)
```

INPUT GIVEN

OUTPUT OBTAINED

REMARKS

Signature of Faculty	Signature of Student
GRADE :	Date :

Subject :	Software :
	Hardware :
Branch : / Semester :	Page No. / Prog No.

PROBLEM STATEMENT

ALGORITHM & CODE :

```
{   int mid = left + (right - left)/2;
    mergesort (arr, left, mid);
    mergesort (arr, mid+1, right);
    merge (ar, left, mid, right);
  }
}
int main()
{ int n;
  printf ("Enter the number of elements (n>5000):");
  Scanf ("%d", &n);
 if (n < 5000)
 { printf ("please enter a value greater than
            5000.\n");
   return 1;
 }
 int arr[n];
 Srand (time(0));
for (int F=0; i<n; i++)
 { arr[i] = rand() % 10000;
 }
 Clock_t start = clock();
```

INPUT GIVEN

OUTPUT OBTAINED

REMARKS

GRADE :

Signature of Faculty Date :	Signature of Student Date :

Subject :

Software :

Hardware :

Branch :

Semester :

Page No.

Prog No.

.EM STATEMENT

RITHM & CODE :

```
mergesort(arr, 0, n-1);
clock_t end = clock();
double time_taken = ((double)(end - start))/
                    CLOCKS_PER_SEC;
printf("Time taken to sort %d elements : %f
        seconds\n", n, time_taken);
printf("Sorted elements (first 10) :\n");
for (int i=0; i<10; i++)
{ printf("%d", arr[i]);
}
printf("\n");
return 0;
}
```

output obtained

```
enter the no of elements in the array (>=5000):
5000
Oreginal array (first 10 elements):
36533 73769 65114 33391 10093 64849
10388 70953 88403 78109
```

IVEN

OBTAINED

KS

Signature of Faculty

Signature of Student

Software :

Hardware :

Branch : | Semester : | Page No. | Prog No.

ATEMENT

CODE :

Sorted array :
4 70 101 102 140 154 184 185 198 230
time taken by merge sort : 0.000993 seconds.

enter the no. of elements in the array (>=5000): 10000
orfginal array (first 10 elements):
92885 1647 8690 6204 92957 94969
60224 67452 25917 57709
Sorted array :
12 13 22 29 36 54 68 77 98 99
time taken by merge sort : 0.001939 seconds.