Practical-5

(1) Generate large number of elements randomly and sort all the elements in ascending order using Merge sort. Analyze the time complexity for best, average and worst case.

Code:

```
#include<stdio.h>
#include <time.h>
#include<sys/time.h>
void merge(int arr[],int first,int middle,int last)
{
   int i=first,j=middle+1,k=0,t;
   int brr[last-first+1];
   while(i<=middle && j<=last)
   {
      if(arr[i]<=arr[j])
      {
         brr[k]=arr[i];
         i++;
         k++;
      }
      else
      {
         brr[k]=arr[j];
         j++;
         k++;
      }
   }
   while(i<=middle)
   {
      brr[k]=arr[i];
      i++;
      k++;
   }
   while(j<=last)
   {
```

```
        brr[k]=arr[j];
        j++;
        k++;
    }
    for(t=0;t<last-first+1;t++)
    {
        arr[first+t]=brr[t];
    }
}

void merge_sort(int arr[],int first,int last)
{
    if(first<last)
    {
        int middle=(first+last)/2;
        merge_sort(arr,first,middle);
        merge_sort(arr,middle+1,last);
        merge(arr,first,middle,last);
    }

}
void checkfor(int arr[],int n,int i)
{
    int t2,t1;
    struct timeval tv;
    struct timezone tz;

    //Best Case

    for(i=0;i<n;i++)
    {
        arr[i]=i;
    }
    gettimeofday(&tv,&tz);
    t1=((tv.tv_sec*1000000)+(tv.tv_usec));
    merge_sort(arr,0,n-1);
    gettimeofday(&tv,&tz);
    t2=((tv.tv_sec*1000000)+(tv.tv_usec));
```

```
    printf("\n %d \t\t %d ",n,(t2-t1));

    //Average case

    for(i=0;i<n;i++)
    {
       arr[i]=rand()%n;
    }
    gettimeofday(&tv,&tz);
    t1=((tv.tv_sec*1000000)+(tv.tv_usec));
    merge_sort(arr,0,n-1);
    gettimeofday(&tv,&tz);
    t2=((tv.tv_sec*1000000)+(tv.tv_usec));
    printf("\t\t%d",(t2-t1));

    //worst case

    for(i=0;i<n;i++)
    {
       arr[i]=n-i;
    }
    gettimeofday(&tv,&tz);
    t1=((tv.tv_sec*1000000)+(tv.tv_usec));
    merge_sort(arr,0,n-1);
    gettimeofday(&tv,&tz);
    t2=((tv.tv_sec*1000000)+(tv.tv_usec));
    printf("\t\t%d",(t2-t1));
}
void main()
{
  int arr[20000],n,i;
  printf("\n Value     \tBest case\tAverage case\tWorst case\n");
  n=5000;
  checkfor(arr,n,i);
  n=10000;
  checkfor(arr,n,i);
  n=15000;
  checkfor(arr,n,i);
```

```
 n=20000;
 checkfor(arr,n,i);
}
```

Output-Table:

Value	Best Case	Average Case	Worst Case
5000	1030	1588	2158
10000	2165	3749	78814
15000	3312	5288	11313
20000	3628	5709	68769

Graph: