Practical-4

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

Code:

```
#include<stdio.h>
#include<time.h>
#include<sys/time.h>
void heapify(int a[],int n,int i)
{
   int l,r,largest,temp;
   largest=i;
   l=2*i+1;
   r=2*i+2;

   if(l<n && a[l]>a[largest])
      largest = l;
   if(r<n && a[r]>a[largest])
      largest = r;
   if (largest != i)
   {
     temp=a[i];
     a[i]=a[largest];
     a[largest]=temp;
     heapify(a,n,largest);
   }
}
void buildheap(int a[],int n)
{
   int i,half,temp;
   half=(n/2)-1;
   for(i=n;i>=0;i--)
   {
      heapify(a,half,i);
```

```
    }
    for(i=n-1;i>0;i--)
    {
       temp=a[0];
       a[0]=a[i];
       a[i]=temp;
       heapify(a,i,0);
    }
}
void checkfor(int a[],int n,int i)
{
    int t2,t1;
    struct timeval tv;
    struct timezone tz;

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

Output-Table:

Value	Best Case	Average Case	Worst Case
5000	657	1287	5312
10000	1405	10559	1899
15000	1617	64069	4289
20000	7120	60727	9940

Graph: