Practical-1

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

Code:

```
#include<stdio.h>
#include<time.h>
#include<sys/time.h>

int a[20000],i,n;
void bubblesort()
{
   int j,temp,flag=0;
      for(i=0;i<n-1;i++)
      {
         flag=0;
         for(j=0;j<n-i-1;j++)
         {
            if(a[j]>a[j+1])
            {
               temp=a[j];
               a[j]=a[j+1];
               a[j+1]=temp;
               flag=1;
            }
         }
         if(flag==0)
            break;
      }
}
void timec()
{
   int t2,t1;
   struct timeval tv;
```

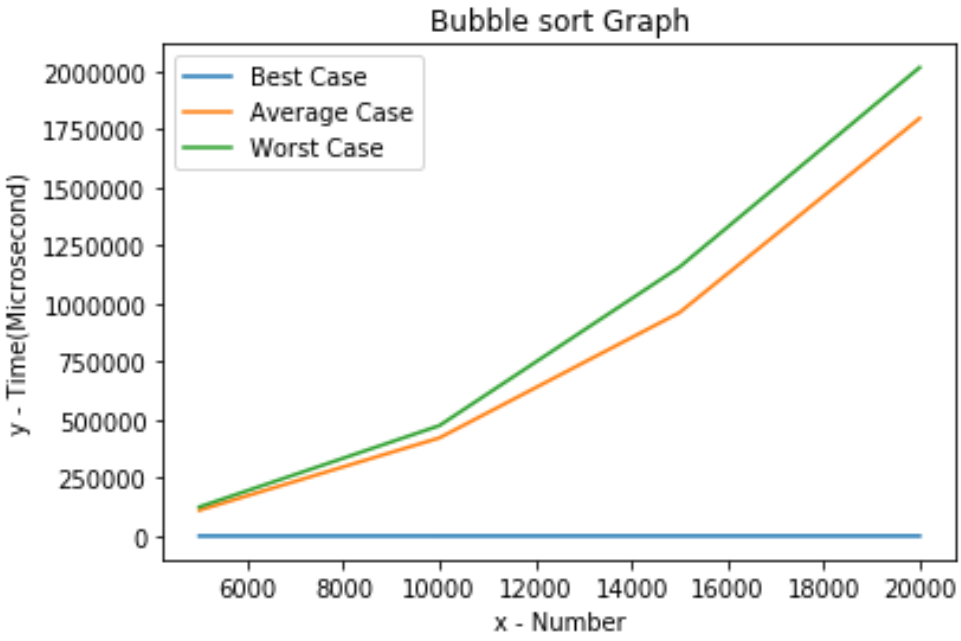
```
struct timezone tz;

*Best Case*

for(i=0;i<n;i++)
{
   a[i]=i;
}
gettimeofday(&tv,&tz);
t1=((tv.tv_sec*1000000)+(tv.tv_usec));
bubblesort();
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));
bubblesort();
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));
bubblesort();
gettimeofday(&tv,&tz);
```

```
    t2=((tv.tv_sec*1000000)+(tv.tv_usec));
    printf("\t\t%d",(t2-t1));
}
void main()
{
  printf("\n Value  \tBest case\tAverage case\tWorst case\n");
  n=5000;
  timec();
  n=10000;
  timec();
  n=15000;
  timec();
  n=20000;
  timec();
}
```

Output-Table:

Value	Best Case	Average Case	Worst Case
5000	22	110687	134924
10000	33	422403	485029
15000	121	963643	1157372
20000	69	1791852	2116448

Graph:


Bubble sort Graph
Best Case
Average Case
Worst Case
y - Time(Microsecond)
2000000
1750000
1500000
1250000
1000000
750000
500000
250000
0
6000
8000
10000
12000
14000
16000
18000
20000
x - Number