Practical-3

(1) Generate large number of elements randomly and search a given number from it Using sequential search and binary search. Analyse the time complexity for best, Average and worst case.

Code:

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

int sequential_searching(int a[],int n,int x)
{
   int i,flag=0;
   for(i=0;i<n;i++)
   {
      if(x==a[i])
      {
         flag=1;
         break;
      }
   }
   if(flag==1)
      return i;
   else
      return -1;
}

int binary_searching(int a[],int l,int h,int x)
{
   int mid;
   while(l<=h)
   {
      mid=(l+h)/2;
      if(x==a[mid])
         return mid;
```

```
            else if(x>a[mid])
                l=mid+1;
            else
                h=mid-1;
        }
        return -1;
    }

    void bubblesort(int a[],int n)
    {
        int i,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 main()
    {
      int a[20000],i,x,n=10000,index,l=0,h=n-1;
      int t2,t1;
      struct timeval tv;
      struct timezone tz;
      for(i=0;i<n;i++)
      {
          a[i]=rand()%n;
```

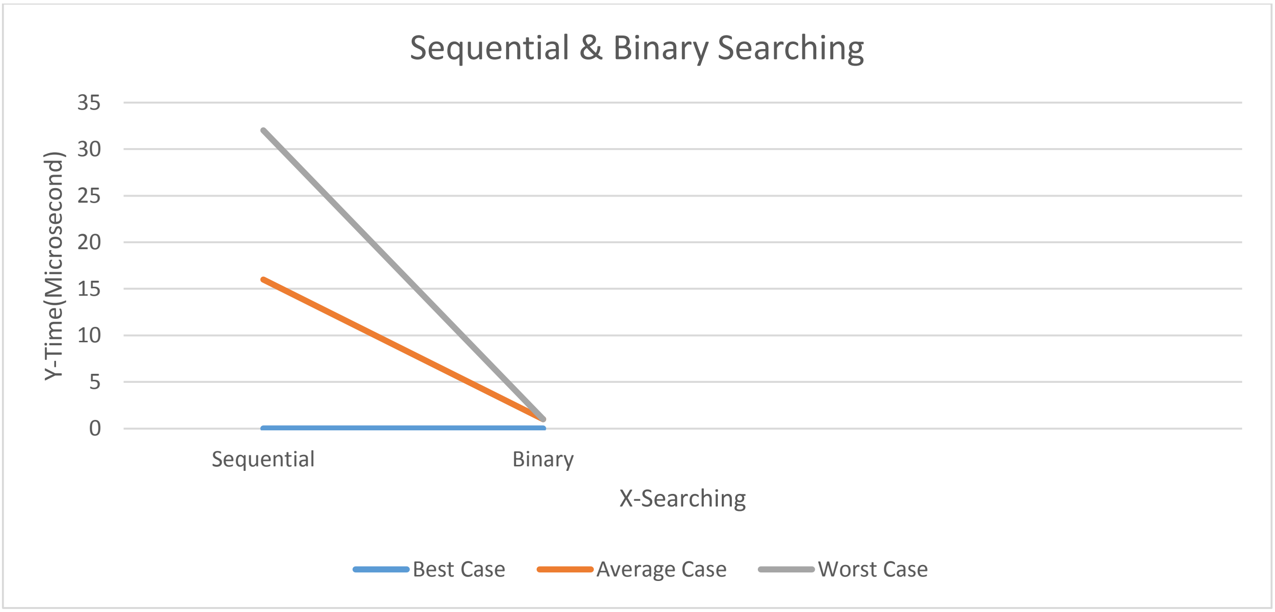
```
}
bubblesort(a,n);
printf("\n Searching  Best case  Average Case  Worst Case");

//Sequential Searching
//Best Case

x=a[0];
gettimeofday(&tv,&tz);
t1=((tv.tv_sec*1000000)+(tv.tv_usec));
index=sequential_searching(a,n,x);
gettimeofday(&tv,&tz);
t2=((tv.tv_sec*1000000)+(tv.tv_usec));
printf("\n Sequential \t %d ",(t2-t1));

//average case

x=h/2;
x=a[x];
gettimeofday(&tv,&tz);
t1=((tv.tv_sec*1000000)+(tv.tv_usec));
index=sequential_searching(a,n,x);
gettimeofday(&tv,&tz);
t2=((tv.tv_sec*1000000)+(tv.tv_usec));
printf("      %d ",(t2-t1));

//Worst case
x=a[n-1];
gettimeofday(&tv,&tz);
t1=((tv.tv_sec*1000000)+(tv.tv_usec));
index=sequential_searching(a,n,x);
gettimeofday(&tv,&tz);
t2=((tv.tv_sec*1000000)+(tv.tv_usec));
printf("      %d ",(t2-t1));

//Binary Searching
//Best case
```

```
 x=h/2;
 x=a[x];
 gettimeofday(&tv,&tz);
 t1=((tv.tv_sec*1000000)+(tv.tv_usec));
 index=binary_searching(a,l,h,x);
 gettimeofday(&tv,&tz);
 t2=((tv.tv_sec*1000000)+(tv.tv_usec));
 printf("\n Binary \t %d ",(t2-t1));

 //Average Case

 x=((h/2)+200);
 x=a[x];
 gettimeofday(&tv,&tz);
 t1=((tv.tv_sec*1000000)+(tv.tv_usec));
 index=binary_searching(a,l,h,x);
 gettimeofday(&tv,&tz);
 t2=((tv.tv_sec*1000000)+(tv.tv_usec));
 printf("     %d ",(t2-t1));

 //Worst Case

 x=a[h];
 gettimeofday(&tv,&tz);
 t1=((tv.tv_sec*1000000)+(tv.tv_usec));
 index=binary_searching(a,l,h,x);
 gettimeofday(&tv,&tz);
 t2=((tv.tv_sec*1000000)+(tv.tv_usec));
 printf("     %d ",(t2-t1));
}
```

Output-Table:

Searching	**Best Case**	**Average Case**	**Worst Case**
Sequential	0	16	32
Binary	0	1	1

Graph:


Sequential & Binary Searching
Y-Time(Microsecond)
35
30
25
20
15
10
5
0
Sequential
Binary
X-Searching
Best Case
Average Case
Worst Case