Shri Vile Parle Kelavani Mandal's
DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING
(Autonomous College Affiliated to the University of Mumbai)
NAAC Accredited with "A" Grade (CGPA : 3.18)

## Department of Information Technology

**COURSE CODE: DJS22ITC403** **DATE:5/10/2023**

**COURSE NAME: Design and Analysis of Algorithms** **CLASS: I1-Batch1**

**NAME : ANISH SHARMA** **SAP ID: 60003220045**

### Experiment No.1

**Aim:** Complexity analysis for Linear search and Binary search

1) **Analysis of Linear Search**

**Code:**

```
#include <stdio.h>

void main()
{

   int arr1[5];
   for (int h = 0; h < 5; h++)
   {
      scanf("%d", &arr1[h]);
   }
   int t = 0;
   printf("input size | no.of searches\n");
   while (++t < 5)
   {
      int n = arr1[t];
      int i = 0, j = 0;
      int arr[n];
      int s = -1;
      int count = 0, flag = 0;
      for (i = 0; i < n; i++)
      {
         arr[i] = i * n;
      }
      for (j = 0; j < n; j++)
      {
         if (s == arr[j])
```

```
            {
               count++;
               flag = 1;
               break;
            }
            count++;
         }
         printf("%d\t\t%d\n",arr1[t],count);
         count=0;
      }
}
```

**Output:**

```
1 10 100 1000 10000
input size | no.of searches
10              10
100             100
1000            1000
10000           10000
```

**2) Analysis of binary search**

**Code:**

```
#include <stdio.h>

#include <stdio.h>
#include <math.h>

void main()
{
   int nums[] = {1, 10, 100, 1000, 10000, 100000};
   int table[6];
   int logm[6];
   for (int k = 0; k < 6; k++)
   {
      int n = nums[k];
      int i = 0, j = 0;
```

```
        int arr[n];
        int s = 123;
        int start = 0, end = n - 1;
        int mid = (start + end) / 2;
        int count = 0, flag = 0;
        for (i = 0; i < n; i++)
        {
            arr[i] = i * n;
        }
        while (start <= end)
        {
            if (s == arr[mid])
            {
                count++;
                break;
                flag = 1;
            }
            else if (s > arr[mid])
            {
                start = mid + 1;
            }
            else if (s < arr[mid])
            {
                end = mid - 1;
            }
            mid = (start + end) / 2;
            count++;
        }
        table[k] = count;
        logm[k]=ceil(log(n)/log(2));
    }
    printf("number splits logn");
    for (int i = 0; i < 6; i++)
    {
        printf("\n");
        printf("%d\t%d\t%d", nums[i], table[i], logm[i]);
        printf("\n");
    }
}
```

**Output:**

```
number splits logn
1       1       0

10      4       4

100     7       7

1000    10      10

10000   13      14

100000  17      17
```

**Conclusion:** We implemented analysis of linear search and analysis of binary search in this experiment by comparing their time complexities for each input size and comparison with logn for binary search