Practical No: 4(B)

Practical Title: Implementation of Searching Technique Using Python
.

Aim: a) Write a Python program to store roll numbers of students in an array who attended a training program in random order. Write a function for searching whether a particular student attended a training program or not, using Linear search and Sentinel search.
b) Write a Python program to store roll numbers of student array who attended training program in sorted order. Write a function for searching whether a particular student attended a training program or not, using Binary search and Fibonacci search.

Prerequisite:

- Basics of searching.

Objectives:

- To understand implementation of Array data structure.
- Understand the implementation of One Dimensional Array with various operations

.

Input: Roll no of student who does not attained or attained training program.

Output: Searching the given roll number that attained the training program using all the searching techniques.

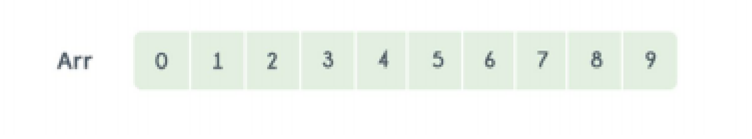
Theory :

Linear Search:
Linear search is used on a collection of items. It relies on the technique of traversing a list from start to end by exploring properties of all the elements that are found on the way. For example, consider an array of integers of size N. You should find and print the position of all the elements with value x. Here, the linear search is based on the idea of matching each element from the beginning of the list to the end of the list with the integer x, and then printing the position of the element if the condition is `True'.

Pseudo code for Linear Search:

```
def linear_search (list, value) :
    for each item in the list :
        if  item == value :
                return the item's location
        end if
    end for
```

For example, consider the following image:



If you want to determine the positions of the occurrence of the number 7 in this array, determine the positions, every element in the array from start to end, i.e., from index 0 to index 9 will be compared with number 7, to check which element matches the number 7.

Time Complexity:
The time complexity of the linear search is O(n) because each element in an array is compared only once.

B.Sentinel Search:
The algorithm ends either the target is found or the last element is compared. The algorithm can be modified to eliminate the end of list test by placing the target at the end of list as just one additional entry. This additional entry at the end of the list is called as Sentinel Ex. Searching key is 13, then it is to be inserted at the end of list.

0	1	2	3	4	5	6	7
10	19	7	11	12	5	3	13

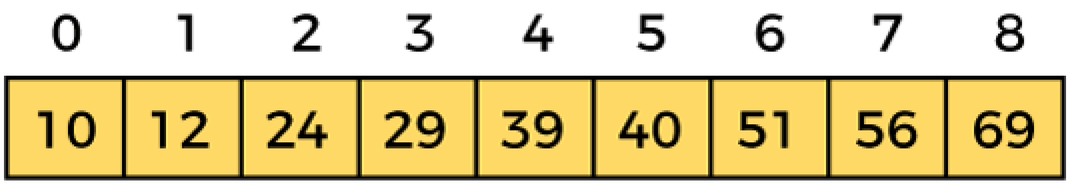
Pseudo code for Sentinel search:

```
int Sentinel_search( int a[], int n, int key)
 {
            int i = 0;
            a[n] = key;
            while(a[i]  != key)
                    i++;
            if(i= =n)
                    return(-1);
            return(i);
```

Time complexity for sentinel search O(n).

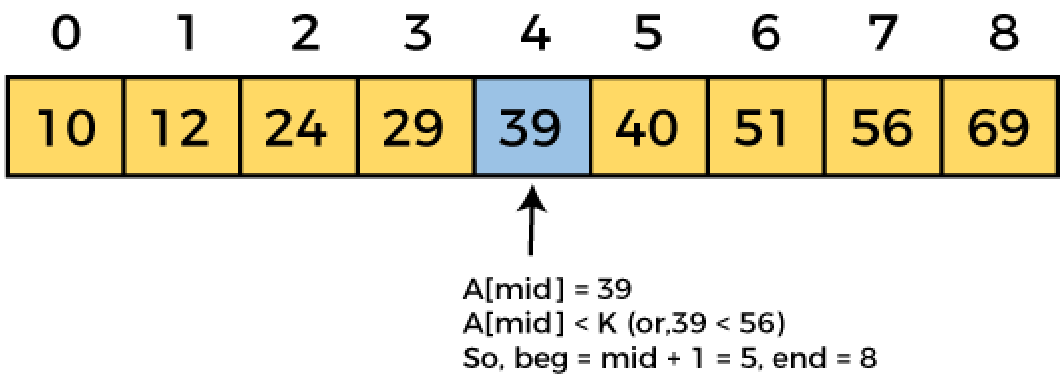
C.Binary Search

Suppose let the given array:



0	1	2	3	4	5	6	7	8
10	12	24	29	39	40	51	56	69

Let the element to search is, K = 56
We have to use the below formula to calculate the mid of the array
Low=0
High=8
Mid=(low+high)/2=(0+8)/2=4

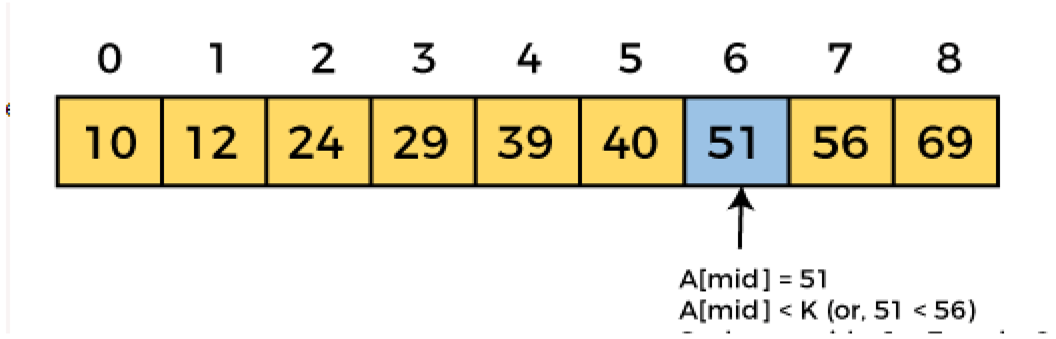


0	1	2	3	4	5	6	7	8
10	12	24	29	39	40	51	56	69

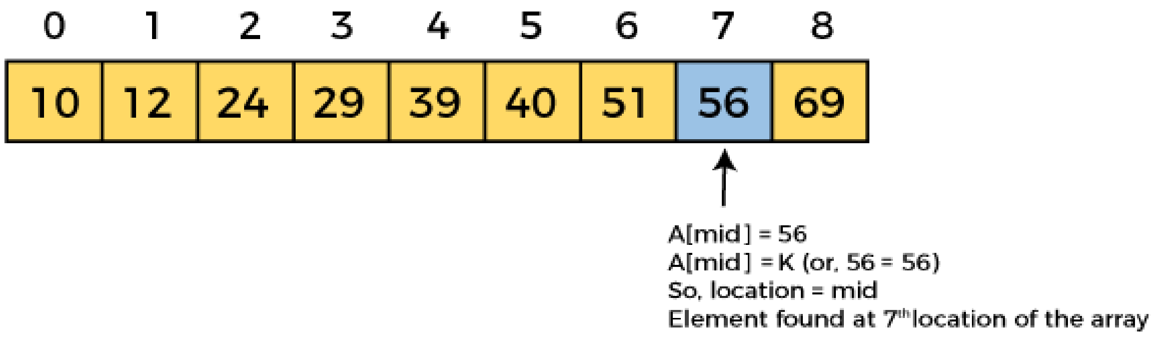
A[mid] = 39
A[mid] < K (or,39 < 56)
So, beg = mid + 1 = 5, end = 8

Now low=mid+1 and high=8

mid=(low+high)/2=(5+8)/2=6



Low=mid+1=7   high=8
Mid=(7+8)/2=7



Time Complexity: The worst case time complexity is O(log2 n).

D.Fibonacci Search :

Given a sorted array arr[] of size n and an element x to be searched in it. Return index of x if it is present in array else return -1.

```
Input:  arr[] = {2, 3, 4, 10, 40}, x = 10

Output:  3

Element x is present at index 3.


Input:  arr[] = {2, 3, 4, 10, 40}, x = 11

Output:  -1

Element x is not present.
```

Fibonacci Search is a comparison-based technique that uses Fibonacci numbers to search an element in a sorted array.
Similarities with Binary Search:
1. Works for sorted arrays
2. A Divide and Conquer Algorithm.

3. Has Log n time complexity.
Differences with Binary Search:
1. Fibonacci Search divides given array in unequal parts
2. Binary Search uses division operator to divide range. Fibonacci Search doesn't use /, but uses + and -. The division operator may be costly on some CPUs.
3. Fibonacci Search examines relatively closer elements in subsequent steps. So when input array is big that cannot fit in CPU cache or even in RAM, Fibonacci Search can be useful.

Conclusion:
We have implemented the python program for searching techniques.

A	**P**	**J**	**Total**	**Dated Sign**
3	4	3	10	