Searching

- Searching is a process of finding a particular element among several given elements.
- The search is successful if the required element is found.
- Otherwise, the search is unsuccessful.

Searching Algorithms
The searching of an element in the given array may be carried out in the following two ways-

1. Linear Search
2. Binary Search

Linear Search

- Linear Search is the simplest searching algorithm.
- It traverses the array sequentially to locate the required element.
- It searches for an element by comparing it with each element of the array one by one.
- So, it is also called as **Sequential Search**.

Linear Search Algorithm is applied when

- No information is given about the array.
- The given array is unsorted or the elements are unordered.
- The list of data items is smaller.

Algorithm for linear searching

1. Input: A set of data in array a, and variable x. i.e., the target element
   A[1….n];
   item=x;
   location=0;
2. Search the list to find the target element:

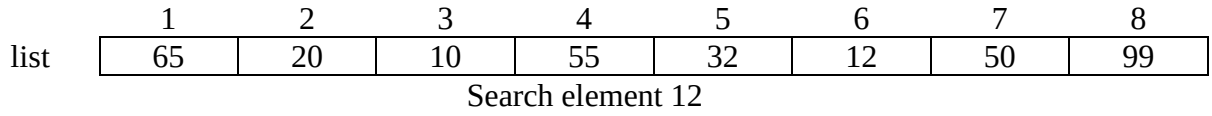
```
for (i=1; i<=n; i++)
{
    if (A[i]==item)
        {
            print "Found";
            location=i;
            stop searching;
        }

}
```

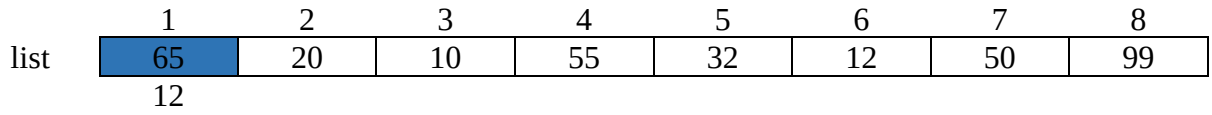
3. if (i>n) print " Not Found";
4. Output: Found or Not Found.

Example

Consider the following list of elements and the element to be searched...

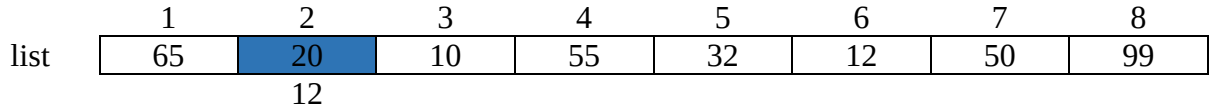


Step 1: Search element (12) is compared with first element (65)



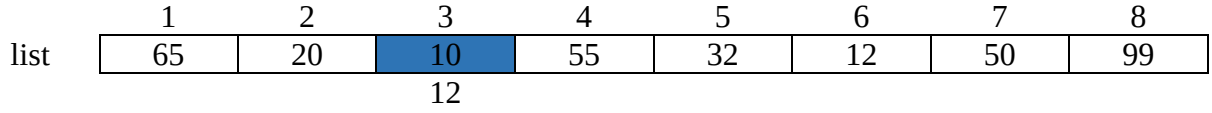
Both are not matching. So, move to next element.

Step 2: Search element (12) is compared with next element (20)



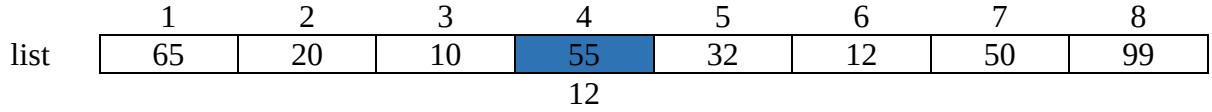
Both are not matching. So, move to next element.

Step 3: Search element (12) is compared with next element (10)



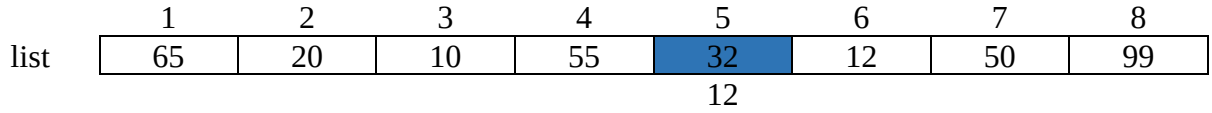
Both are not matching. So, move to next element.

Step 4: Search element (12) is compared with next element (55)



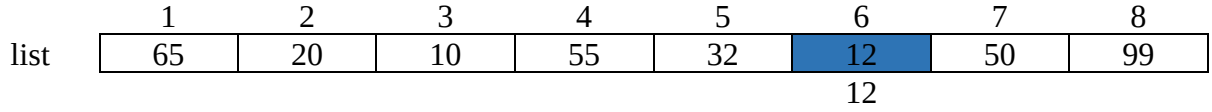
Both are not matching. So, move to next element.

Step 5: Search element (12) is compared with next element (32)



Both are not matching. So, move to next element.

Step 6: Search element (12) is compared with next element (12)



Both are matching. So, we stop comparing and display element found at index 6.

Binary Search

- Binary Search is one of the fastest searching algorithms.
- It is used for finding the location of an element in a linear array.
- It works on the principle of divide and conquer technique.

Binary Search Algorithm can be applied only on Sorted arrays.

So, the elements must be arranged in-

- Either ascending order if the elements are numbers.
- Or dictionary order if the elements are strings.

To apply binary search on an unsorted array,

- First, sort the array using some sorting technique.
- Then, use binary search algorithm.

Algorithm for binary searching

1. Input A[1…..m], x; //A is the array with size m and x is the target element.

2. first=1, last=m

3. while(first<=last)

{

mid = (first+last)/2;

i. if (x=A[mid]), then print mid; //target element=A[mid] or target element is //in index mid

break(stop searching);

ii. else if(x<A[mid]), then last=mid-1;

iii. else first=mid+1;

}

4. if(first>last), print “Not Found”;

5. Output: mid or “Not Found”;

Example

Consider the following list of elements and element 15 has to be searched in it using Binary Search Algorithm.

1	2	3	4	5	6	7
3	10	15	20	35	40	60

Step-01:

- To begin with, we take first=1 and last=7.
- We compute location of the middle element as-

  mid = (first + last) / 2 = (1 + 7) / 2 = 4

- Here, A[mid] = A[4] = 20 ≠ 15 and first < =last.
- So, we start next iteration.

Step-02:

- Since A[mid] = 20 > 15, so we take last = mid – 1 = 4 – 1 = 3 whereas beg remains unchanged.
- We compute location of the middle element as-

  mid = (first + last) / 2 = (1 + 3) / 2 = 2

- Here, A[mid] = A[2] = 10 ≠ 15 and first < =last.
- So, we start next iteration.

Step-03:

- Since A[mid] = 10 < 15, so we take first = mid + 1 = 2 + 1 = 3 whereas end remains unchanged.
- We compute location of the middle element as-

  mid = (first + last) / 2 = (3 + 3) / 2 = 3

- Here, A[mid] = A[3] = 15 which matches to the element being searched.
- So, our search terminates in success and index 3 is returned.

Differences between Linear search and Binary search

- A linear search scans one item at a time, without jumping to any item. In contrast, binary search cuts down your search to half as soon as you find the middle of a sorted list.
- In linear search, the worst case complexity is O(n), where binary search making $O(\log_2 n)$ comparisons.

- Time taken to search elements keep increasing as the number of elements is increased when searching through linear process. But binary search compresses the searching period by dividing the whole array into two half.
- Linear search does the sequential access whereas Binary search access data randomly.
- Input data needs to be sorted in Binary Search and not in Linear Search.
- In linear search, performance is done by equality comparisons. In binary search, performance is done by ordering comparisons.
- Binary search is better and quite faster than linear search.
- Linear search uses sequential approach. But, binary search implements divide and conquer approach.
- Linear search is quick and easy to use, but there is no need for any ordered elements. Where binary search algorithm is tricky, and elements are necessarily arranged in order.
- The best case time in linear search is for the first element that is O(1). And the other side O(1) is the middle element in binary search.
- Linear search can be implemented in an array as well as in linked list, but binary search can't be implemented directly on linked list.
- Binary search is efficient for the larger array. If the amount of data is small, then linear search is preferable because this searching process is fast when data is small.