SEARCHING

Searching is the process of finding the location of given element in the linear array. The search is said to be successful if the given element is found ie the element does exists in the array otherwise unsuccessful.

There are two appeoaches to search operation:

(1) Linear search
(2) Binary search.

The algorithm that one chooses generally depends on organization of the array elements. If the elements are in random order, then one have to use linear search technique,
and if the elements are already sorted, then it is preferable to use binary search technique. These two search techniques are:

(1) LINEAR SEARCH

Given the array a, the only way to search for given element item is to compare item with each element of a one by one. This method, which traverses a sequentially to locate item is called linear search or sequential search.

Algo: LinearSearch(a, n, item, loc)

Begin
for i = 0 to (n-1) by 1 do

```
if (a[i] == item) then
    set loc = i
    exit
  endif
endfor
  set loc = -1
End.
```

Analysis of Linear Search

In the best possible case, the item may occur at first position. In this case, the search operation terminates in success with just one comparison. However, the worst case occurs when either the item is present at last position or missing from the array. In the former case, the search terminates in success with n comparisons.

In worst case the linear search is O(n) operations.

BINARY SEARCH

Suppose the elements of the array A are sorted in ascending order (if the elements are numbers) or dictionary order (if the elements are strings). The best searching algorithm, called binary search, is used to find the location of the given element.

Complexity = $O(\log_2 n)$

Consider the following sorted array A with 7 elements

3, 10, 15, 20, 35, 40, 60

and we want to search element 15.

Soln

a[0]	a[1]	a[2]	a[3]	a[4]	a[5]	a[6]
3	10	15	20	35	40	60

Here, beg = 0, end = 6 and compute the location of the middle element as

$$mid = (beg + end)/2 = (0+6)/2 = 3$$

Since a[mid], ie a[3] ≠ 15 and beg < end.
We start next iteration,
as a[mid] = 20 > 15, therefore we take
end = mid − 1 = 3 − 1 = 2, where as beg remains unchagged. Thus,

$$mid = (beg + end)/2 = (0+2)/2 = 1$$

Since a[mid], ie a[1] ≠ 15, and beg < end.
we start end iteration.
As a[mid] = 10 < 15, therefore, we take

beg = mid + 1 = 1 + 1 = 2, where as
end remains unchanged.

Since beg = end, again compute location of middle element as

$$mid = (beg + end)/2 = (2+2)/2 = 2$$

Since a[mid] ie a[2] = 15, the search terminates on success.
The element 15 is found at index 2.

Alg:

Algo: BinarySearch(a, n, item, loc)

Here a is an linear array of size n. The algorithm finds the location of the element item in sorted linear array a. If search ends in success it sets loc to the index of the element; otherwise, it sets loc to -1.

Here, Variables beg and end to keep track of the first element and last element of the array to be searched, and Variable mid is used to as index of the middle element of the array.

```
Begin
    set beg = 0
    set end = n-1
    set mid = (beg + end)/2
    while ((beg ≤ end) and (a[mid] ≠ item))
                                    do
        if (item < a[mid]) then
            set end = mid - 1
        else
            set beg = mid + 1
        end if
        set mid = (beg + end)/2
    end while
        if (beg > end) then
            set loc = -1
        else
            set loc = mid
        end if
End
```