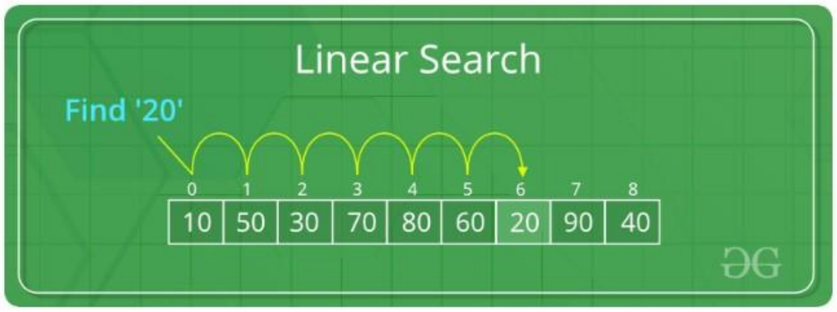
Linear Search

Linear Search means to sequentially traverse a given list or array and check if an element is present in the respective array or list. The idea is to start traversing the array and compare elements of the array one by one starting from the first element with the given element until a match is found or end of the array is reached.



Examples :

```
Input : arr[] = {10, 20, 80, 30, 60, 50,
                     110, 100, 130, 170}
          key = 110;
Output : 6
Element 110 is present at index 6

Input : arr[] = {10, 20, 80, 30, 60, 50,
                     110, 100, 130, 170}
           key = 175;
Output : -1
Element 175 is not present in arr[].
```

Problem: Given an array **arr[]** of **N** elements, write a function to search a given element **X** in arr[].

Algorithm:

- Start from the leftmost element of arr[] and one by one compare X with each element of arr[].
- If X matches with an element, return the index.
- If X doesn't match with any of elements and end of the array is reached, return -1.

Function:

```

// Function to linearly search the element X in the
// array arr[] of N elements
int search(int arr[], int N, int X)
{
    // Pointer to traverse the array
    int i;

    // Start traversing the array
    for (i = 0; i < N; i++)
    {
        // If a successful match is found,
        // return the index
        if (arr[i] == X)
            return i;
    }

    // If the element is not found,
    // and end of array is reached
    return -1;
}

```

Time Complexity: O(N). Since we are traversing the complete array, so in worst case when the element X does not exists in the array, number of comparisons will be N. Therefore, *worst case time complexity of the linear search algorithm is O(N)*.