LINEAR SEARCH

Searching Algorithm

ALPHA MEWING SIGMA

GROUP

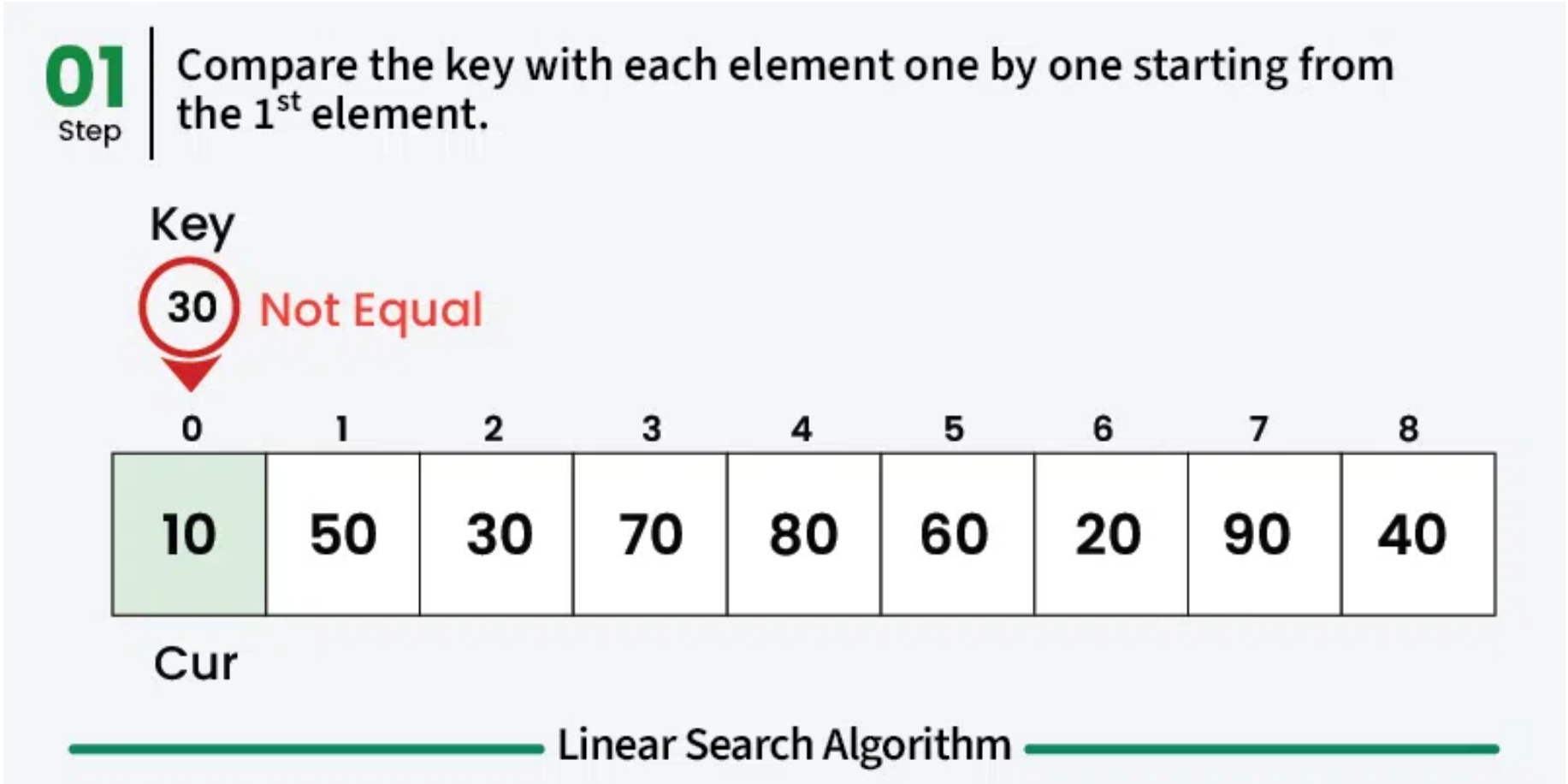
LINEAR SEARCH

In Linear Search, we iterate over all the elements of the array and check if it the current element is equal to the target element. If we find any element to be equal to the target element, then return the index of the current element. Otherwise, if no element is equal to the target element, then return -1 as the element is not found. Linear search is also known as **sequential search**.

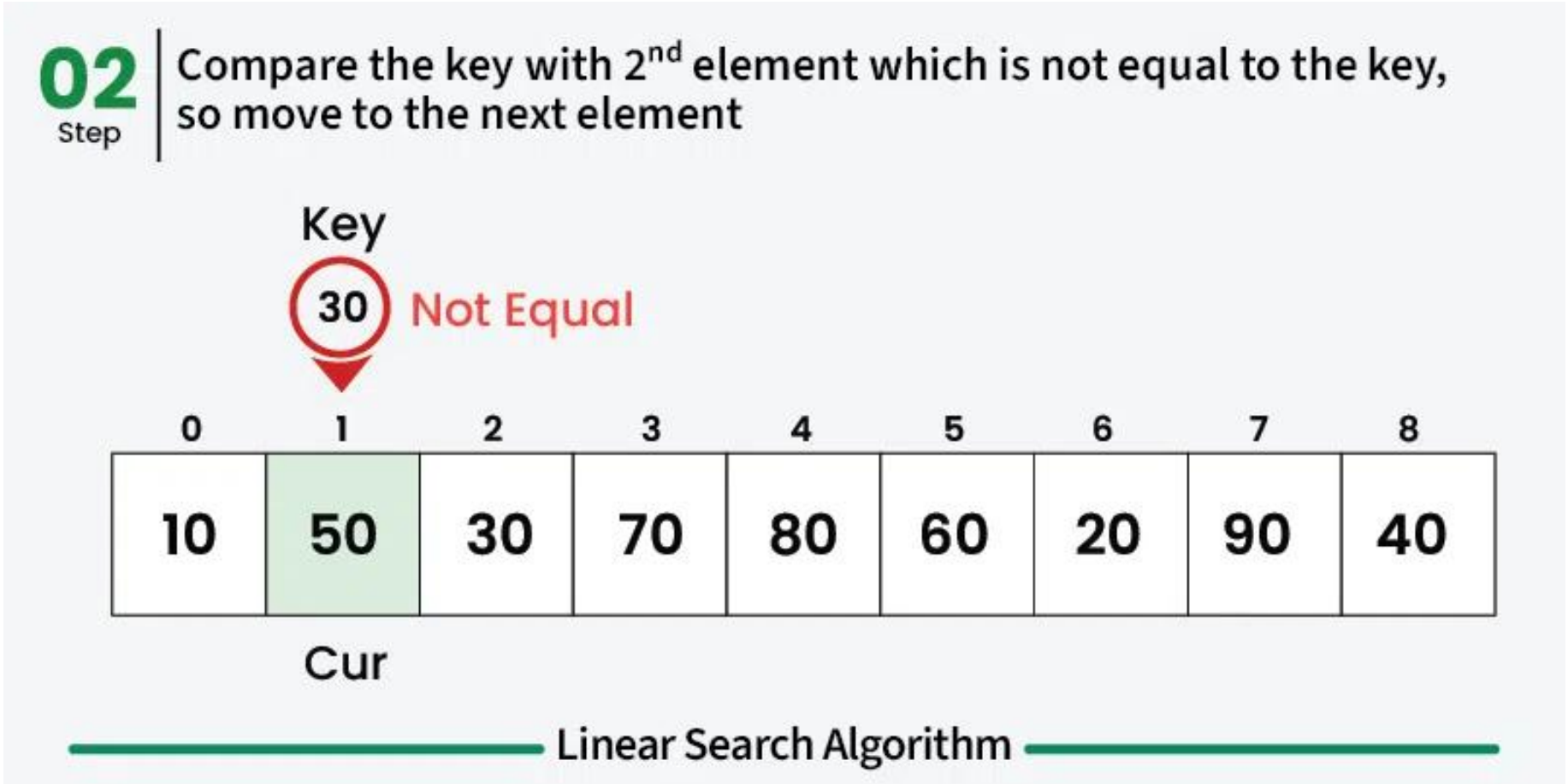
Given an array, **arr[]** of **n** integers, and an integer element **x**, find whether element **x** is **present** in the array. Return the **index** of the first occurrence of **x** in the array, or **-1** if it doesn't exist.

- ***Input***: *arr[] = [1, 2, 3, 4], x = 3*
  Output: *2*
  Explanation: *There is one test case with array as [1, 2, 3 4] and element to be searched as 3. Since 3 is present at index 2, the output is 2.*
- ***Input***: *arr[] = [10, 8, 30, 4, 5], x = 5*
  Output: *4*
  Explanation: *For array [10, 8, 30, 4, 5], the element to be searched is 5 and it is at index 4. So, the output is 4.*
- ***Input***: *arr[] = [10, 8, 30], x = 6*
  Output: *-1*
  Explanation: *The element to be searched is 6 and its not present, so we return -1.*

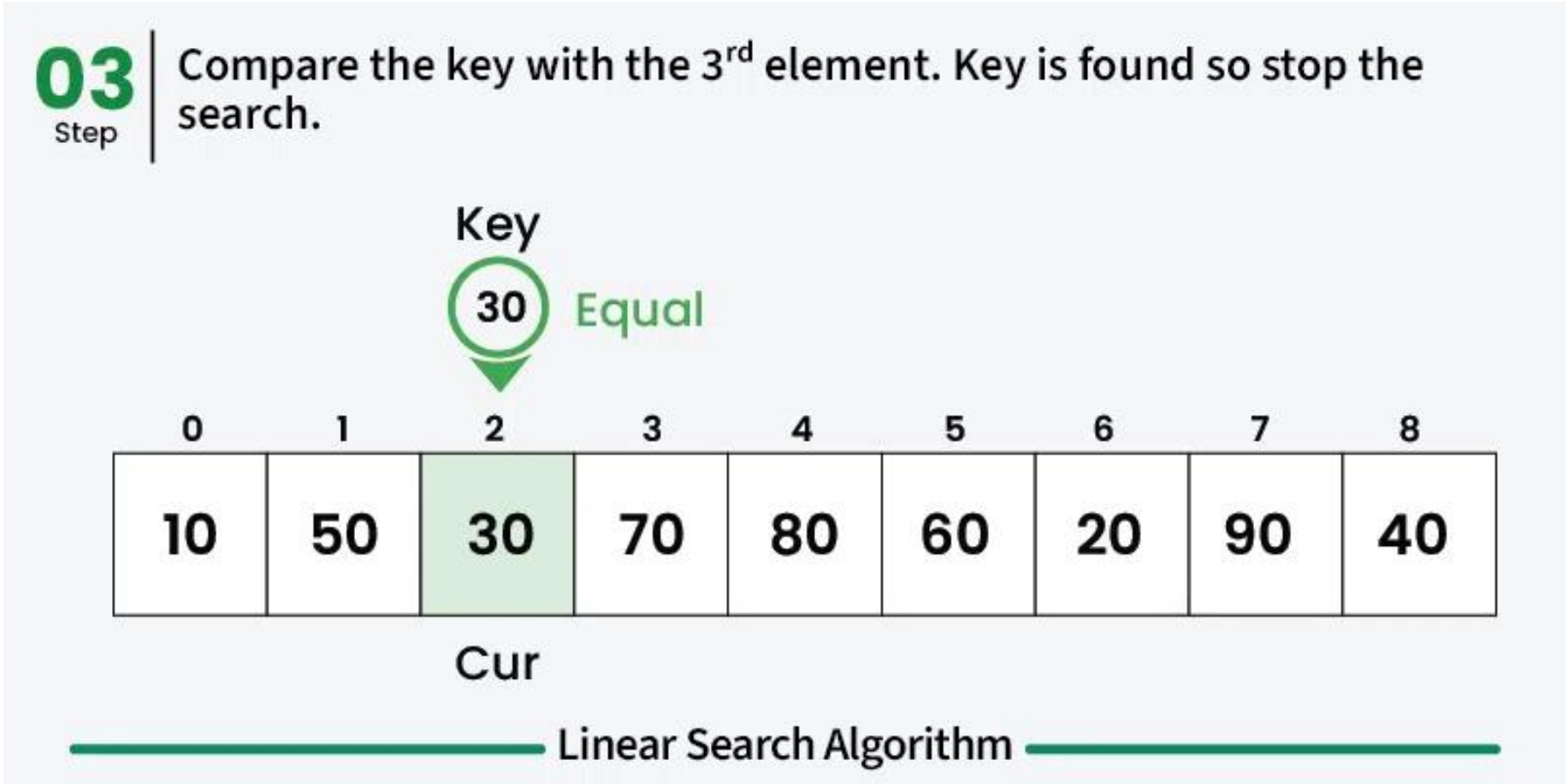
For example: Consider the array **arr[] = {10, 50, 30, 70, 80, 20, 90, 40}** and **key** = 30



For example: Consider the array **arr[] = {10, 50, 30, 70, 80, 20, 90, 40}** and **key** = 30



For example: Consider the array **arr[] = {10, 50, 30, 70, 80, 20, 90, 40}** and **key** = 30



Time Complexity:

Best Case: O(1)

In the best case, the key might be present at the first index.

Worst Case: O(N) where N is the size of the list.

In the worst case, the key might be present at the last index i.e., opposite to the end from which the search has started in the list.

Average Case: O(N)

Space Complexity:

Auxiliary Space: O(1)

as except for the variable to iterate through the list, no other variable is used.

Applications of Linear Search Algorithm:

- **Unsorted Lists:** When we have an unsorted array or list, linear search is most commonly used to find any element in the collection.
- **Small Data Sets:** Linear Search is preferred over binary search when we have small data sets with
- **Searching Linked Lists:** In linked list implementations, linear search is commonly used to find elements within the list. Each node is checked sequentially until the desired element is found.
- **Simple Implementation:** Linear Search is much easier to understand and implement as compared to Binary Search or Ternary Search.

Advantages of Linear Search Algorithm:

- Linear search can be used irrespective of whether the array is sorted or not. It can be used on arrays of any data type.
- Does not require any additional memory.
- It is a well-suited algorithm for small datasets.

Disadvantages of Linear Search Algorithm:

- Linear search has a time complexity of O(N), which in turn makes it slow for large datasets.
- Not suitable for large arrays.

When to use Linear Search Algorithm?

- When we are dealing with a small dataset.
- When you are searching for a dataset stored in contiguous memory.

Java Implementation

```
public static int search(int arr[], int N, int x)
    {
        // Iterate over the array in order to
        // find the key x
        for (int i = 0; i < N; i++) {
            if (arr[i] == x)
                return i;
        }
        return -1;
    }
 public static void main(String args[])
    {
        int arr[] = { 2, 3, 4, 10, 40 };
        int x = 10;
        int result = search(arr, arr.length, x);
        if (result == -1)
            System.out.print(
                "Element is not present in array");
        else
            System.out.print("Element is present at index "
                             + result);
    }
}
```

Output

Present at Index 3

Thanks for Your Attention

OUR TEAM

Muhammad Andhika

NIM. 20250040092

Regith Rayabi

NIM.20250040075

Syifa Nurul Fadilah

NIM.20250040087

M. Nizar Wirapradana

NIM.20250040070