BINARY SEARCH

Searching Algorithm

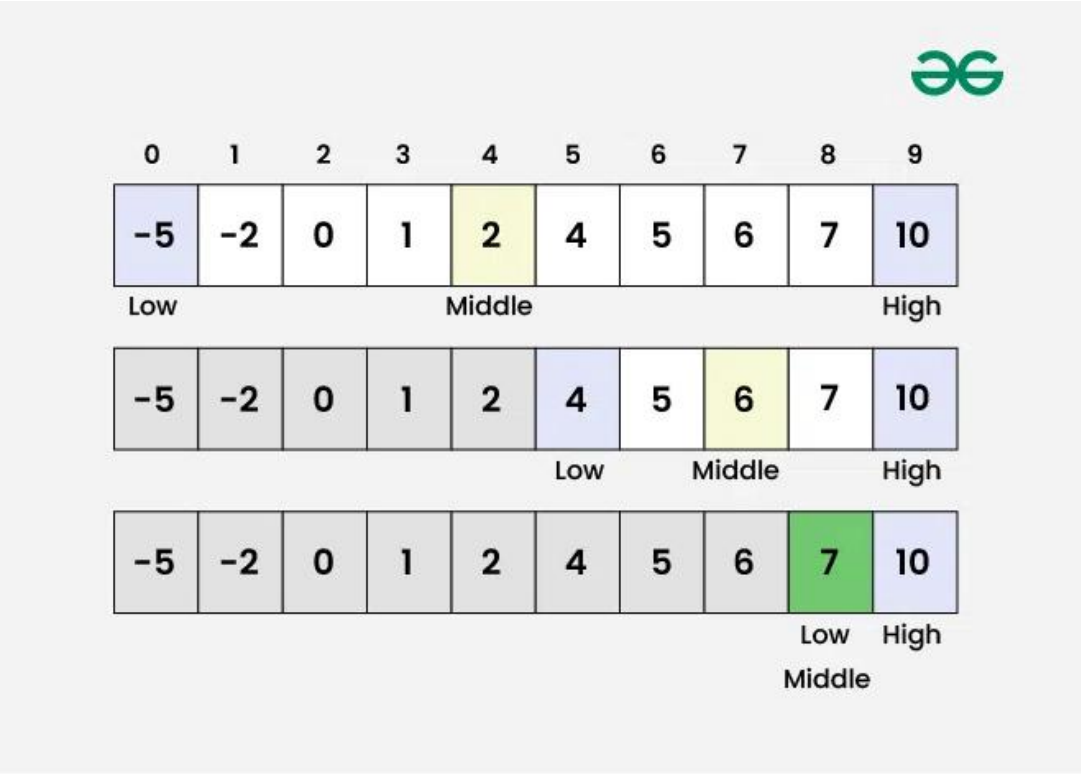
ALPHA MEWING SIGMA

GROUP

BINARY SEARCH

Definition

Binary Search is a searching algorithm that operates on a sorted or monotonic search space, repeatedly dividing it into halves to find a target value or optimal answer in logarithmic time O(log N).



So, what best condition to use Binary Search?

To apply Binary Search algorithm:

1. The data structure must be sorted.

2. Access to any element of the data structure should take constant time.

But, How it Works?

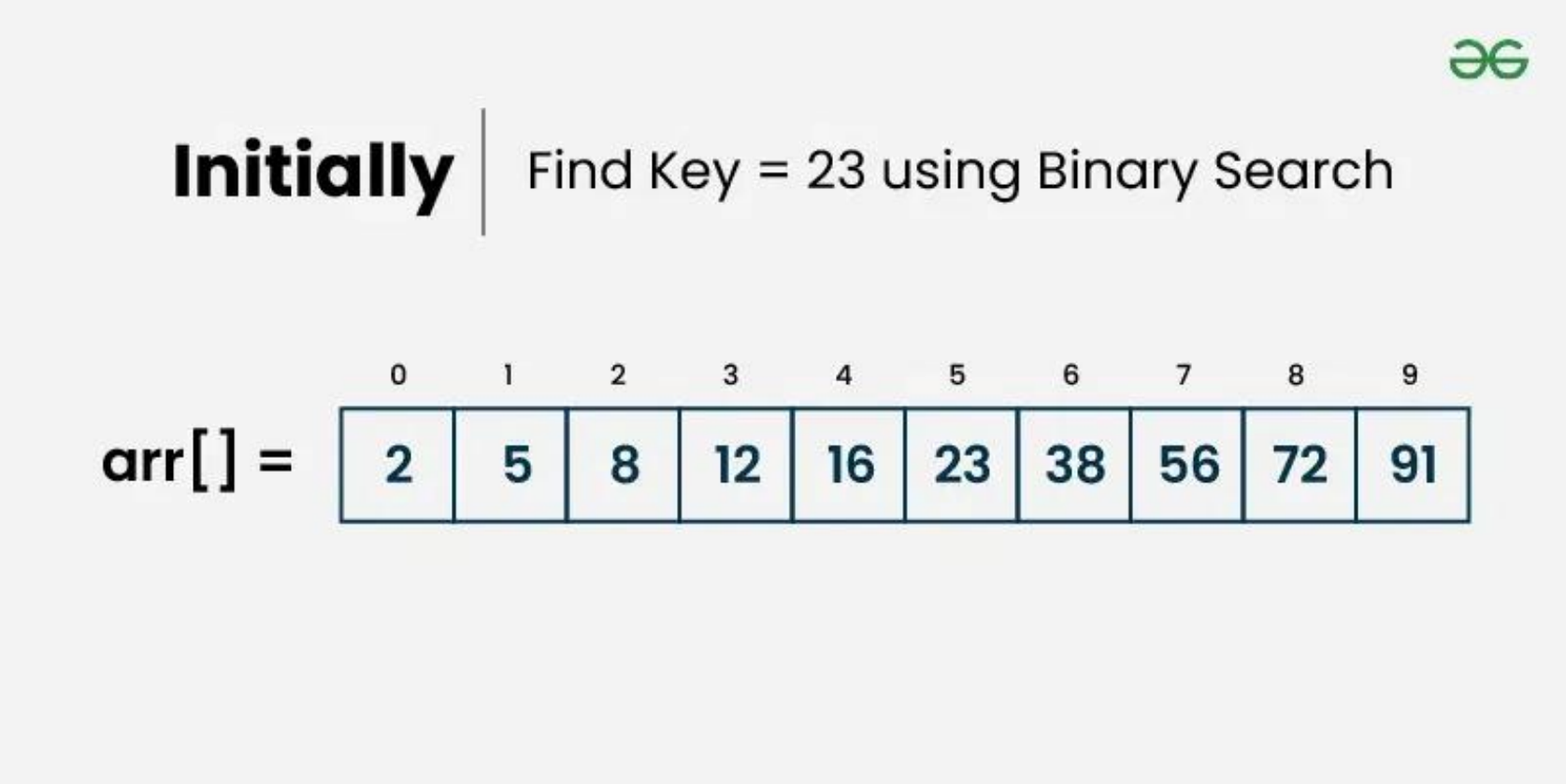
Binary Search Algorithm

- Below is the step-by-step algorithm for Binary Search:
- Divide the search space into two halves by **finding the middle index "mid"**.
- Compare the middle element of the search space with the **key**.
- If the **key** is found at middle element, the process is terminated.
- If the **key** is not found at middle element, choose which half will be used as the next search space.
  -> If the **key** is smaller than the middle element, then the **left** side is used for next search.
  -> If the **key** is larger than the middle element, then the **right** side is used for next search.
- This process is continued until the **key** is found or the total search space is exhausted.

How does Binary Search Algorithm work?

To understand the working of binary search, consider the following illustration:

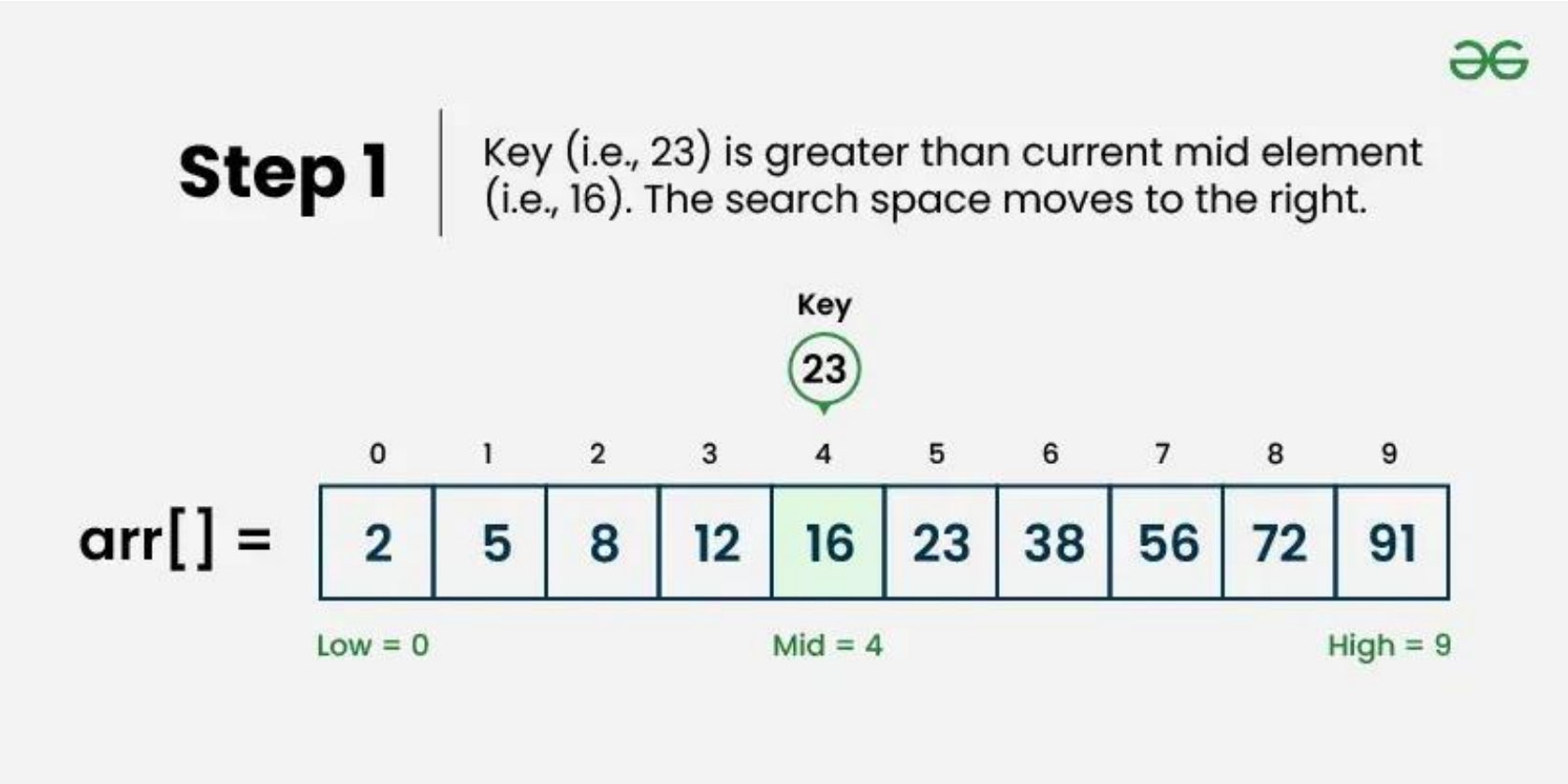
Consider an array **arr[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91}**, and the **target = 23**.



How does Binary Search Algorithm work?

To understand the working of binary search, consider the following illustration:

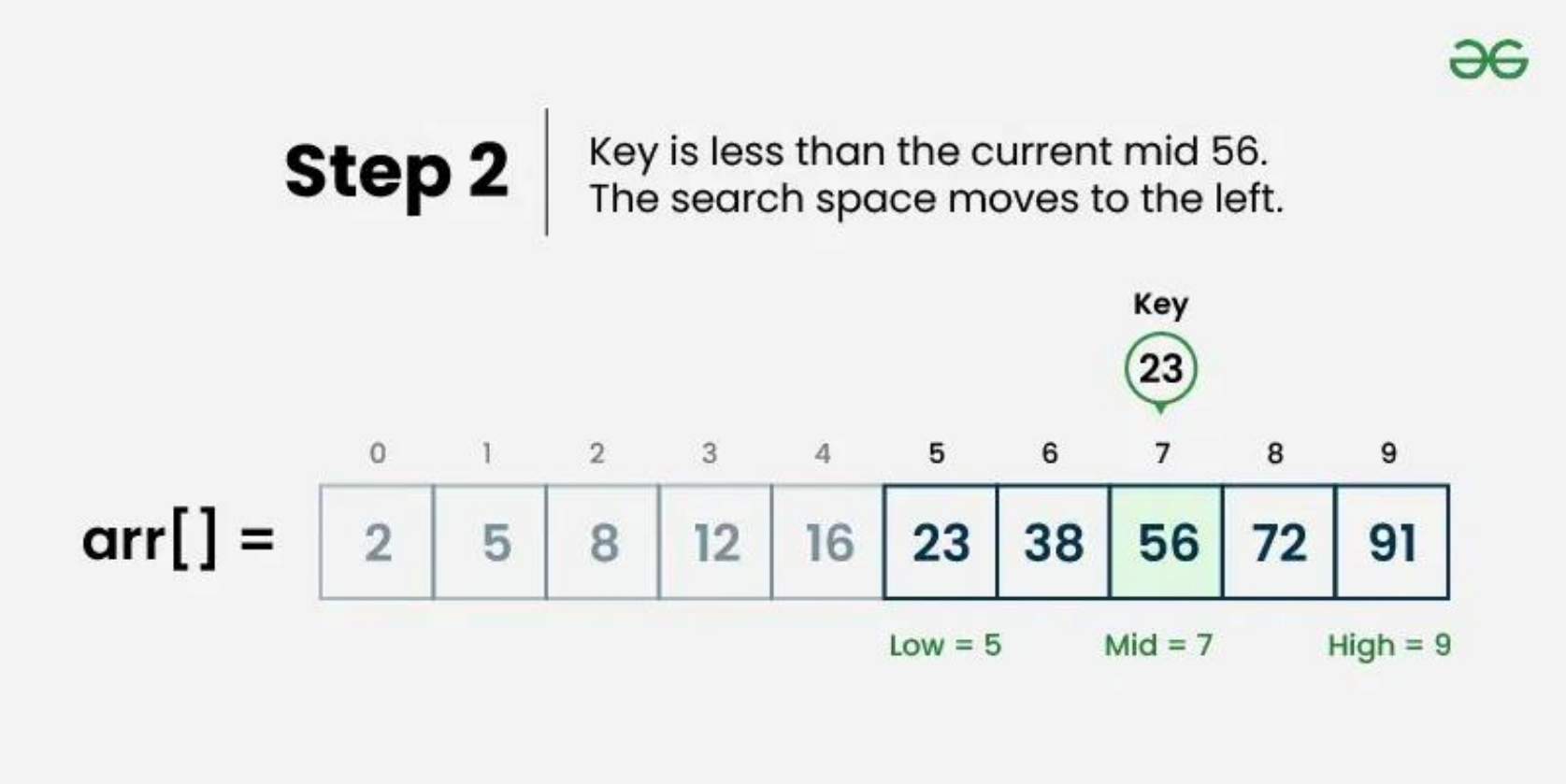
Consider an array **arr[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91}**, and the **target = 23**.



How does Binary Search Algorithm work?

To understand the working of binary search, consider the following illustration:

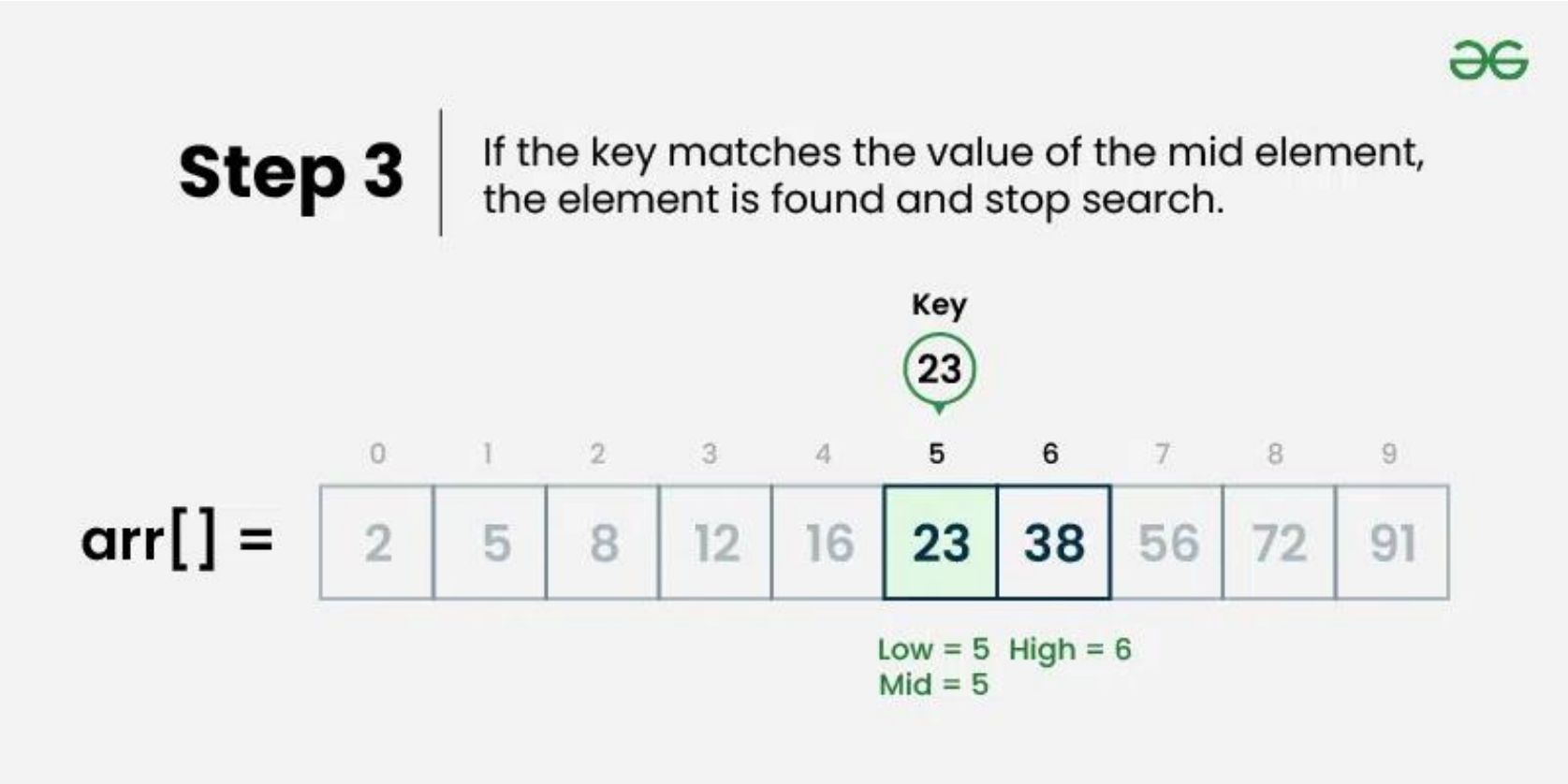
Consider an array **arr[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91}**, and the **target = 23**.



How does Binary Search Algorithm work?

To understand the working of binary search, consider the following illustration:

Consider an array **arr[] = {2, 5, 8, 12, 16, 23, 38, 56, 72, 91}**, and the **target = 23**.



How to Implement Binary Search Algorithm?

The **Binary Search Algorithm** can be implemented in the following two ways

- Iterative Binary Search Algorithm
- Recursive Binary Search Algorithm

Iterative Binary Search Algorithm: O(log n) Time and O(1) Space

Time Complexity:

-> Best Case: O(1)

-> Average Case: O(log N)

-> Worst Case: O(log N)

Auxiliary Space:

O(1), If the recursive call stack is considered then the auxiliary space will be O(log N). then

Here's the Implementation in Java

Iterative Binary

```
static int binarySearch(int arr[], int x) {
        int low = 0, high = arr.length - 1;
        while (low <= high) {
            int mid = low + (high - low) / 2;
            // Check if x is present at mid
            if (arr[mid] == x)
                return mid;
            // If x greater, ignore left half
            if (arr[mid] < x)
                low = mid + 1;
            // If x is smaller, ignore right half
            else
                high = mid - 1;
        }
        // If we reach here, then element was
        // not present
        return -1;
    }
```

Recursive Binary Search Algorithm:

```
// A recursive binary search function. It returns
    // location of x in given array arr[low..high] is present,
    // otherwise -1
    static int binarySearch(int arr[], int low, int high, int x) {
        if (high >= low) {
            int mid = low + (high - low) / 2;
            // If the element is present at the
            // middle itself
            if (arr[mid] == x)
                return mid;
            // If element is smaller than mid, then
            // it can only be present in left subarray
            if (arr[mid] > x)
                return binarySearch(arr, low, mid - 1, x);

            // Else the element can only be present
            // in right subarray
            return binarySearch(arr, mid + 1, high, x);
        }
        // We reach here when element is not present
        // in array
        return -1;
    }
```

OUR TEAM

Muhammad Andhika

NIM. 20250040092

Regith Rayabi

NIM.20250040075

Syifa Nurul Fadilah

NIM.20250040087

M. Nizar Wirapradana

NIM.20250040070

ANY QUESTION?

THANKS FOR YOUR ATTENTION