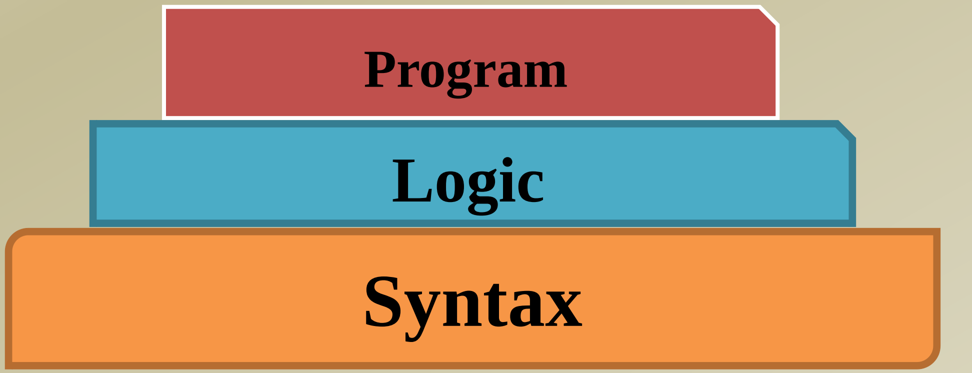
Code at Random (OPC) PVT. LTD.

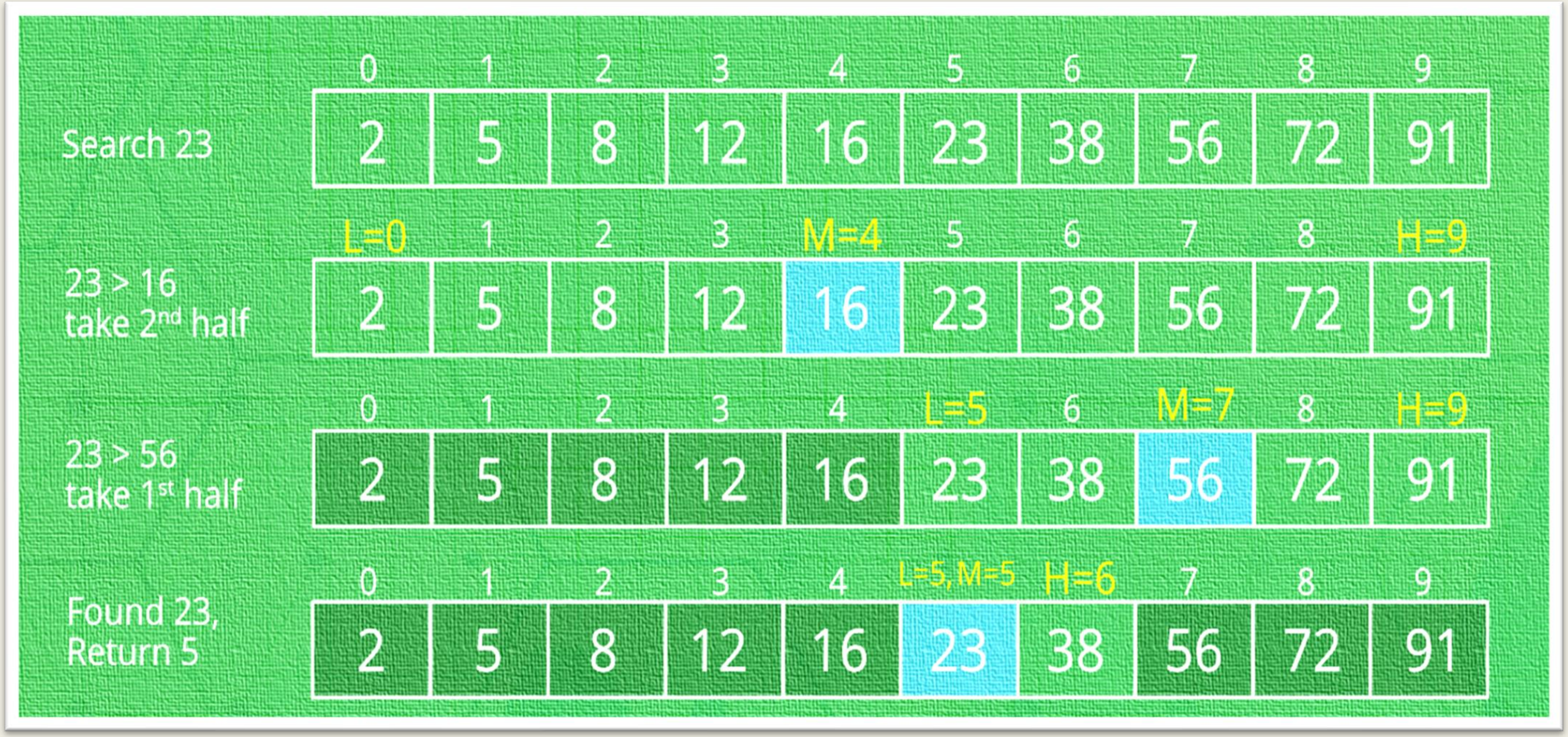
Binary Search in JAVA


Program

Logic

Syntax

How to Use Binary Search

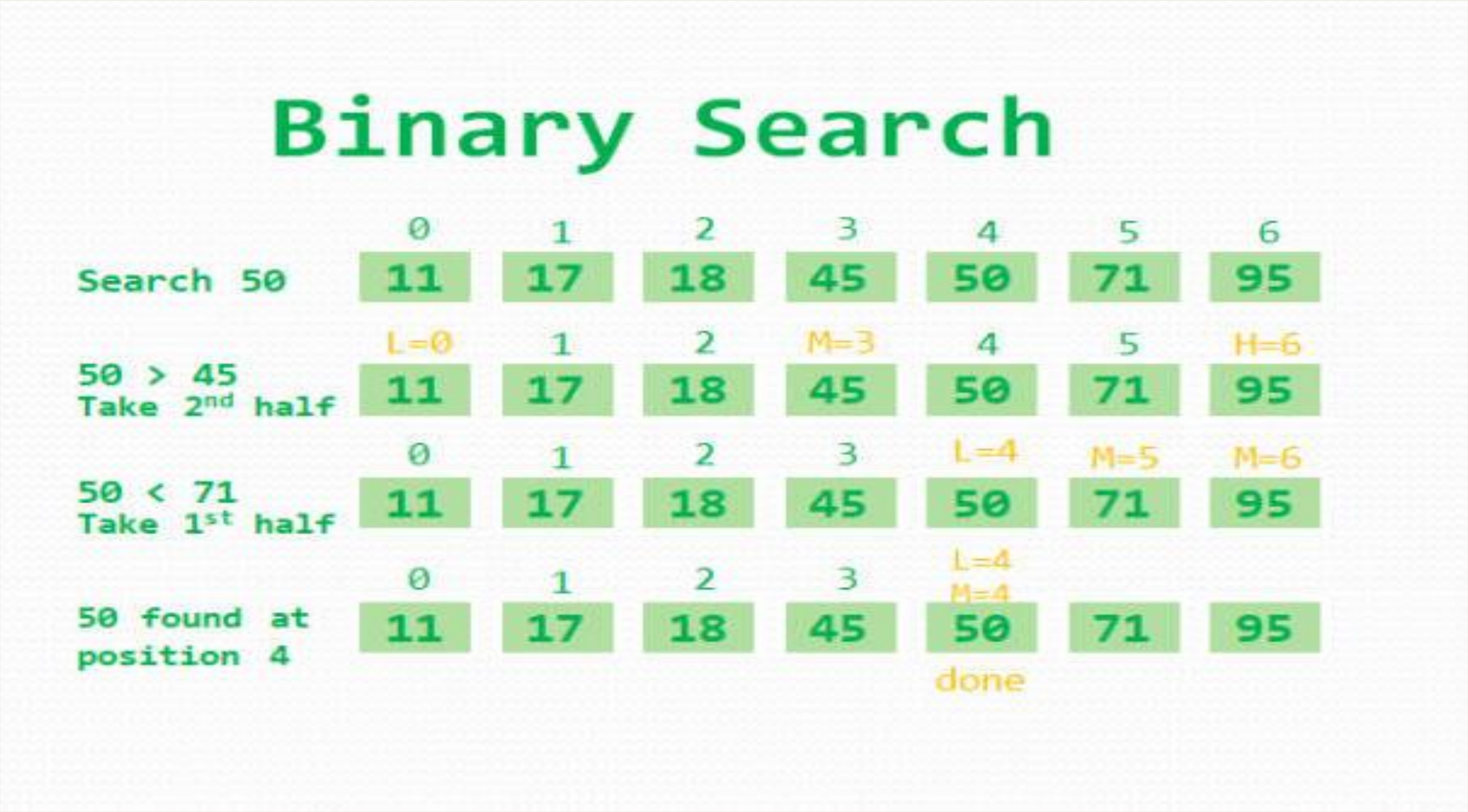


	0	1	2	3	4	5	6	7	8	9
Search 23	2	5	8	12	16	23	38	56	72	91
	L=0	1	2	3	M=4	5	6	7	8	H=9
23 > 16 take 2nd half	2	5	8	12	16	23	38	56	72	91
	0	1	2	3	4	L=5	6	M=7	8	H=9
23 > 56 take 1st half	2	5	8	12	16	23	38	56	72	91
	0	1	2	3	4	L=5, M=5	H=6	7	8	9
Found 23, Return 5	2	5	8	12	16	23	38	56	72	91

Algorithm of Binary Search

- This searching technique follows the divide and conquer strategy.
- Binary Search Algorithm is a very efficient technique for searching but it needs some order on which partition of the array will occur.
- It works only on the sorted arrays. To perform the binary search, follow these steps:
    - Find the middle element of the array → mid = (low+high)/2.
    - Compare x with the middle element.
    - If x matches with the middle element, we return the mid index.
    - Else If x is greater than the mid element, then x can only lie in the right half subarray after the mid element. So we recur for the right half.
    - Else (x is smaller) recur for the left half.

Example of Binary Search



Binary Search

	0	1	2	3	4	5	6
Search 50	11	17	18	45	50	71	95
	L=0	1	2	M=3	4	5	H=6
50 > 45 Take 2nd half	11	17	18	45	50	71	95
	0	1	2	3	L=4	M=5	M=6
50 < 71 Take 1st half	11	17	18	45	50	71	95
	0	1	2	3	L=4 M=4		
50 found at position 4	11	17	18	45	50	71	95
					done		

WAP to input n numbers in an array and a number to search in the array. Perform the **Binary Search** and check whether the number is present in the array or not. If present, print the first occurrence of the number.

```
import java.util.*;
class binary_search {
    public static void main(String Args[]) {
         int L ,i , mid , flag=0;
       Scanner sc = new Scanner(System.in);
       System.out.println("Enter the size of the array");
       L= sc.nextInt();
       int A[] = new int[L];
       System.out.println("Enter the elements of the array");
       for(i =0; i< L ; i++) {
        A[i] = sc.nextInt( );
      }
      System.out.println("Enter the element to be searched");
      int n = sc.nextInt();
      int low=0, high=L-1;
      mid=(low+high)/2;
    while(low<=high){
        mid = (low+high)/2;
       if(A[mid]>n)
            h = mid-1;
 else if(A[mid]<n){
                low=mid+1;
   }
         else{
             flag=1;
             break;
         }
     }
     if(flag==0){
        System.out.println("Element not found");
     }
     else{
         System.out.println("Element found at index "+mid);
     }
   }
}
```