

163. You are given a sorted array 3,9,14,19,25,31,42,47,53 and asked to find the position of the element 31 using Binary Search. Show the mid-point calculations and the steps involved in finding the element. Display, what would happen if the array was not sorted, how would this impact the performance and correctness of the Binary Search algorithm?

Input : N= 9, a[] = {3,9,14,19,25,31,42,47,53}, search key = 31

Output : 6

Test cases

Input : N= 7, a[] = {13,19,24,29,35,41,42}, search key = 42

Output : 7

Test cases

Input : N= 6, a[] = {20,40,60,80,100,120}, search key = 60

Output : 3

PROGRAM :-

```
def binary_search(arr, target):
```

```
    low = 0
```

```
    high = len(arr) - 1
```

```
    while low <= high:
```

```
        mid = (low + high) // 2
```

```
        if arr[mid] == target:
```

```
            return mid
```

```
        elif arr[mid] < target:
```

```
            low = mid + 1
```

```
        else:
```

```
            high = mid - 1
```

```
    return -1
```

```
# Sorted array
```

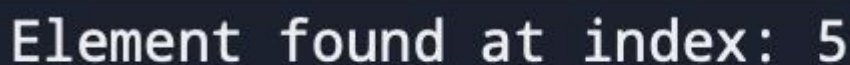
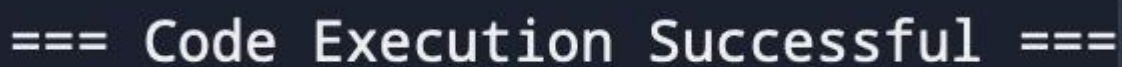
```
arr = [3, 9, 14, 19, 25, 31, 42, 47, 53]
```

```
target = 31
```

```
result = binary_search(arr, target)
```

```
print("Element found at index:", result)
```

OUTPUT:-

A screenshot of a terminal window with a dark background. The text "Element found at index: 5" is displayed in a light-colored, monospaced font.A screenshot of a terminal window with a dark background. The text "=== Code Execution Successful ===" is displayed in a light-colored, monospaced font, with a vertical cursor line at the end.

TIME COMPLEXITY:-O(log N)