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

AIM : TO Display, what would happen if the array was not sorted, how would this impact the performance and correctness of the Binary Search algorithm

PROGRAM:

```
def binary_search(arr, key):
    low, high = 0, len(arr) - 1
    comparisons = 0

    while low <= high:
        comparisons += 1
        mid = (low + high) // 2
        print(f"Step {comparisons}: low = {low}, high = {high}, mid = {mid}")
        if arr[mid] == key:
            return mid, comparisons
        elif arr[mid] < key:
            low = mid + 1
        else:
            high = mid - 1

    return -1, comparisons  # Element not found

arr = [3, 9, 14, 19, 25, 31, 42, 47, 53]
key = 31
index, comparisons = binary_search(arr, key)
print(f"\nOutput: Index of {key}: {index}")
print(f"Number of Comparisons: {comparisons}")
```

OUYPUT:

```
Output

Step 1: low = 0, high = 8, mid = 4
Step 2: low = 5, high = 8, mid = 6
Step 3: low = 5, high = 5, mid = 5

Output: Index of 31: 5
Number of Comparisons: 3

=== Code Execution Successful ===
```

TIME COPLEXITY: O(N)