Complexity Analysis of Jump Search:

- **Time Complexity:**
    - Best Case: O(1)
    - Average Case: O(?n)
    - Worst Case: O(?n)
- **Auxiliary Space:** O(1)

Advantages of Jump Search:

- Better than a linear search for arrays where the elements are uniformly distributed.
- Jump search has a lower time complexity compared to a linear search for large arrays.
- The number of steps taken in jump search is proportional to the square root of the size of the array, making it more efficient for large arrays.
- It is easier to implement compared to other search algorithms like binary search or ternary search.
- Jump search works well for arrays where the elements are in order and uniformly distributed, as it can jump to a closer position in the array with each iteration.

Important points on Jump Search:

- Works only with sorted arrays.
- The optimal size of a block to be jumped is (? n). This makes the time complexity of Jump Search O(? n).
- The time complexity of Jump Search is between Linear Search ((O(n)) and Binary Search (O(Log n)).
- Binary Search is better than Jump Search, but Jump Search has the advantage that we traverse back only once (Binary Search may require up to O(Log n) jumps, consider a situation where the element to be searched is the smallest element or just bigger than the smallest). So, in a system where binary search is costly, we use Jump Search.