**Explain linear search and binary search algorithms :-**

**Linear Search and Binary Search Algorithms**

Both linear search and binary search are fundamental algorithms used to find an item in a list. Each has its own advantages and use cases depending on the nature of the data and the requirements of the search operation.

**1. Linear Search**

**Description**:

* Linear search is the simplest search algorithm. It works by iterating through each element in a list one by one until the target value is found or the end of the list is reached.

**How It Works**:

1. Start at the beginning of the list.
2. Compare the target value with the current element.
3. If the target value matches the current element, return the index of the element (or the element itself, depending on implementation).
4. If the target value does not match, move to the next element.
5. Repeat steps 2-4 until the target value is found or the end of the list is reached.
6. If the end of the list is reached without finding the target value, return null or indicate that the item is not found.

**Time Complexity**:

* **Best Case**: O(1) – If the target value is found on the first attempt.
* **Average Case**: O(n) – On average, half of the elements will be checked.
* **Worst Case**: O(n) – The target value is not present, or it is at the end of the list.

**Advantages**:

* Simple to implement.
* Works on unsorted lists.

**Disadvantages**:

* Inefficient for large lists due to linear time complexity.

**Example**:

java

Copy code

public static int linearSearch(int[] array, int target) {

for (int i = 0; i < array.length; i++) {

if (array[i] == target) {

return i; // Target found at index i

}

}

return -1; // Target not found

}

**2. Binary Search**

**Description**:

* Binary search is a more efficient search algorithm that works on sorted lists. It repeatedly divides the search interval in half, comparing the target value to the middle element.

**How It Works**:

1. Start with the entire list and determine the middle element.
2. Compare the target value with the middle element.
3. If the target value matches the middle element, return the index of the middle element (or the element itself).
4. If the target value is less than the middle element, narrow the search to the left half of the list.
5. If the target value is greater than the middle element, narrow the search to the right half of the list.
6. Repeat steps 1-5 until the target value is found or the search interval is empty.

**Time Complexity**:

* **Best Case**: O(1) – If the target value is the middle element on the first attempt.
* **Average Case**: O(log n) – Each comparison halves the search space.
* **Worst Case**: O(log n) – The search space is reduced logarithmically until the target value is found or the search interval is empty.

**Advantages**:

* Much faster than linear search for large, sorted lists.
* Efficient due to logarithmic time complexity.

**Disadvantages**:

* Requires the list to be sorted. If the list is not sorted, binary search cannot be used.
* More complex to implement compared to linear search.

**Example**:

java

Copy code

public static int binarySearch(int[] array, int target) {

int low = 0;

int high = array.length - 1;

while (low <= high) {

int mid = low + (high - low) / 2;

if (array[mid] == target) {

return mid; // Target found at index mid

} else if (array[mid] < target) {

low = mid + 1; // Search in the right half

} else {

high = mid - 1; // Search in the left half

}

}

return -1; // Target not found

}

**Summary**

* **Linear Search**:
  + **Works On**: Unsorted and sorted lists.
  + **Time Complexity**: O(n).
  + **Use Case**: Simple and effective for small or unsorted lists.
* **Binary Search**:
  + **Works On**: Sorted lists.
  + **Time Complexity**: O(log n).
  + **Use Case**: Efficient for large, sorted lists where quick search performance is required.

Choosing between linear and binary search depends on whether the data is sorted and the size of the dataset. For large datasets where sorting is feasible, binary search offers significant performance advantages. For smaller or unsorted datasets, linear search is straightforward and effective.