**Task - 6 : Library Management System**

Solution by: Mahika Thakur, CSE 7TH Semester, MCKV Institute of Engineering

**Q. Explain linear search and binary search algorithms.**

Linear search is a straightforward search algorithm that checks each element in a list sequentially until the desired element is found or the list ends. It doesn't require the list to be sorted.

Steps:

1. Start from the first element of the list.
2. Compare the target value with the current element.
3. If the current element matches the target value, return its position.
4. If the current element does not match, move to the next element and repeat step 2.
5. If the end of the list is reached without finding the target value, return an indication that the value is not in the list (e.g., null or -1).

Binary search is a more efficient search algorithm that works on sorted lists. It repeatedly divides the list into halves to narrow down the possible positions of the target value.

Steps:

1. Start with the entire list.
2. Determine the middle element of the current segment.
3. Compare the middle element with the target value.
4. If the middle element matches the target value, return its position.
5. If the target value is less than the middle element, repeat the search on the left half of the list.
6. If the target value is greater than the middle element, repeat the search on the right half of the list.
7. If the segment reduces to zero, return an indication that the value is not in the list (e.g., null or -1).

**Q. i. Compare the time complexity of linear and binary search.**

**Linear Search:**

Best Case: O(1) - When the target element is the first element in the list.

Average Case: O(n) - On average, half of the list needs to be checked.

Worst Case: 𝑂 ( 𝑛 ) O(n) - When the target element is the last element or not in the list at all.

**Binary Search:**

Best Case: O(1) - When the target element is the middle element of the list.

Average Case: O(logn) - The list is repeatedly divided in half, reducing the search space exponentially.

Worst Case: O(logn) - The element is at the beginning or end of the list, requiring maximum halving steps.

**ii.Discuss when to use each algorithm based on the data set size and order.**

Linear search is ideal for small, unsorted data sets due to its simplicity and ease of implementation. It's effective when the data isn't sorted and sorting is not an option, or when search operations are infrequent. However, it's inefficient for large lists with its (O(n)) time complexity. Conversely, binary search is highly efficient for large, sorted data sets with its (O(log n)) time complexity, making it suitable for performance-critical applications and frequent searches. It requires the data to be sorted and is more complex to implement, with additional overhead for maintaining sorted order. For frequent searches in a dynamic dataset, a hybrid approach using a sorted structure like a binary search tree can be beneficial.