1. Sequential Search:

Definition: Sequential search, also known as linear search, involves searching through a list one item at a time until the desired item is found or the end of the list is reached.

Algorithm: Iterate through each element of the list, comparing it with the target value until a match is found.

Complexity: O(n) in the worst-case scenario, where 'n' is the number of elements in the list.

2. Binary Search:

Definition: Binary search is an efficient search algorithm used on sorted lists. It halves the search space by repeatedly dividing the list into two halves until the target value is found.

Algorithm: Compare the target value with the middle element of the sorted list. If they match, return the index. If the target is less than the middle element, search the left half; otherwise, search the right half.

Complexity: O(log n) in the worst-case scenario, where 'n' is the number of elements in the list.

3. Hash Table:

Definition: A hash table is a data structure that stores key-value pairs, allowing for efficient lookup, insertion, and deletion operations.

Hash Functions: Hash functions map keys to indices in the hash table array, ideally distributing keys evenly across the table.

Collision Resolution Strategies: Techniques for handling collisions that occur when two keys hash to the same index, including chaining, open addressing, and rehashing.

Implementation: Hash tables can be implemented using arrays and linked lists or other data structures to handle collisions.

4. Sorting Algorithms:

Insertion Sort: Iteratively builds a sorted array by shifting elements as necessary to insert each unsorted element into its correct position.

Bubble Sort: Repeatedly iterates through the list, swapping adjacent elements if they are in the wrong order until the list is sorted.

Quick Sort: Divides the list into smaller sublists based on a pivot element, recursively sorts the sublists, and then combines them.

Two-Way Merge Sort: Divides the list into two halves, recursively sorts each half, and then merges them in sorted order.

Heap Sort: Builds a heap data structure from the list, repeatedly extracts the maximum element from the heap and rebuilds the heap until the list is sorted.

Comparison and Analysis: Sorting algorithms differ in their efficiency, best-case, average-case, and worst-case time complexities, and suitability for different types of data.