**Data Structures Used:**

1. **Hash Table with Separate Chaining:**
   * The hash table uses an array of linked lists (chaining) to handle collisions. Each array element is a sorted linked list.
2. **Linked List (SortedList):**
   * Each linked list is sorted during insertion for efficient search and traversal within the chain.

**Algorithms Used:**

1. **Hashing:**
   * The key is hashed using the modulo operation (key % arraySize) to compute the index in the hash table.
   * This distributes keys across buckets (linked lists) to reduce clustering.
2. **Insertion in a Sorted Linked List:**
   * The linked list is traversed to find the correct position to maintain sorted order. The node is inserted at that position.
3. **Search in a Sorted Linked List:**
   * The linked list is traversed sequentially, stopping if the key is found or the current key is greater than the search key.
4. **Deletion in a Linked List:**
   * The list is traversed to find the node to delete, and pointers are adjusted to remove it from the chain.

**Time Complexity:**

Let nn be the number of keys and mm be the size of the hash table.

1. **Hash Function Calculation:**
   * **Time Complexity:** O(1)O(1)
   * The modulo operation is constant time.
2. **Insertion:**
   * **Best Case:** O(1)O(1)  
     When the linked list is empty (no collisions), insertion occurs in constant time.
   * **Worst Case:** O(n/m)O(n/m)  
     When all keys hash to the same bucket (poor hash distribution), insertion involves traversing the linked list of length n/mn/m.
   * **Average Case:** O(1+α)O(1 + \alpha), where α=n/m\alpha = n/m is the load factor.
3. **Search:**
   * **Best Case:** O(1)O(1)  
     When the key is found in the first node of the chain.
   * **Worst Case:** O(n/m)O(n/m)  
     When the key is at the end of the chain.
   * **Average Case:** O(1+α)O(1 + \alpha).
4. **Deletion:**
   * **Best Case:** O(1)O(1)  
     When the key is the first element of the chain.
   * **Worst Case:** O(n/m)O(n/m)  
     When the key is at the end of the chain.
   * **Average Case:** O(1+α)O(1 + \alpha).
5. **Display Table:**
   * **Time Complexity:** O(n)O(n)  
     Each bucket is traversed, printing all keys.