**Report**

**Theoretical Analysis**

1. **Randomized Quicksort:**
   * **Average-Case Time Complexity: O(nlogn)**
     + On average, balanced partitions are guaranteed by the random pivot selection.
     + The recurrence relation**T(n)=2T(n/2)+O(n)** leads to**O(nlogn)*.***

* **Worst-Case Time Complexity: O(n^2)**
  + It is unusual because of randomization, but it happens when the pivot is regularly the smallest or largest element.
* **Space Complexity: O(logn)** (due to recursion stack).

1. **Hash Table with Chaining:**
   * **Average-Case Time Complexity:**
     + Insertion: O(1)
     + Search: O(1)
     + Deletion: O(1)
   * **Worst-Case Time Complexity:**
     + If every key hashes to the same index, all operations deteriorate to O(n), which is rare with a competent hash algorithm.
   * **Space Complexity**: O(n), where n is the number of key-value pairs.

**Empirical Comparisons**

1. **Randomized Quicksort vs. Deterministic Quicksort:**
   * **Randomly Generated Arrays:**
     + Although Randomized Quicksort has a tiny advantage, both perform similarly.
   * **Already Sorted Arrays:**
     + Randomized Quicksort retains O(nlogn), whereas Deterministic Quicksort deteriorates to O(n^2).
   * **Reverse-Sorted Arrays:**
     + Deterministic Quicksort works poorly, much like sorted arrays.
   * **Arrays with Repeated Elements:**
     + Because random pivot selection is used, Randomized Quicksort manages repeated components more effectively.
2. **Hash Table with Chaining:**
   * **Performance:**
     + The average speed of insertion, search, and deletion is O(1).
     + When there are a lot of collisions, performance suffers, but chaining helps.

**Discussion of Results**

* **Randomized Quicksort:**
  + It consistently exhibits strong performance in all input distributions.
  + It is more dependable than Deterministic Quicksort since it avoids the worst-case situations.
* **Hash Table with Chaining:**
  + With quick average-case performance, it is effective for the majority of use cases.
  + To reduce collisions, a good hash function is necessary.