The hash table/ is a table of elements that have keys. A hash function/ is used for locating a position in the table The input into a hash function is a key value. The output from a hash function is an index of an array (hash table)where the object containing the key is located. The search is done in theata ( 1 ) time. The Insertion is done in theata ( 1 ) time. Collision/ when two elements in a array have the same key value in a hash function. Chaining/ Instead of having an array of objects, we have an array of linked lists, each node of which contains an object. When the elements are spread evenly (or near evenly) among the indexes of a hash table, it is called uniform hashing, can allow search in theata (1) time. Speed comes from reducing the number of collisions. A function pointer is a pointer that holds the address of a function. insert – we’ll insert at the head of the linked list –theata( 1 ) iterator – each step will be theata( 1 ). find – element is found by hashing, so it is theata( 1 ) for uniform hashing (the hash function and hash table are designed so that the length of the collision list is bounded by some small constant) retrieve –is theata( 1 ) for uniform hashing replace –is theata( 1 ) using the current position an operation to determine whether or not the list is empty –is theata( 1 ), because we just test the linked list to see if it is empty an operation to empty out the list –is theata( n ), the best we can do, since each node must be freed.

An inversion/ between any two elements when the element that comes first in the array is greater than the element that comes next. Worst case and average for insertion sort= theata(n^2). Best case for insertion sort is theata(n)time, a already sorted array Time complexities of Quicksort–best: theata( n lg n )–average: theata( n lg n )–worst: theata( n^2) A recursive function, called quicksort A non-recursive function, usually called partition In the partition function, the last element is chosen as a pivot–a special element used for comparison purposes. The time complexity of counting sort depends on both n and also the range of the elements. If the range is smaller than n, counting sort is very fast, running in theata( n ) time. The main counting sort algorithm works with elements that range in value from 0 to some positive integer k…..

HEAPS push(T)/ Adds a new value to the collection being maintained. top() / Returns a reference to the smallest element in the collection pop() /Deletes the smallest element from the collection size()/ Returns the number of elements in the collection empty()/ Returns true if the collection is empty CUCKOO Fastest lookup/insert/delete time: O(1) Advantages:Very space-efficient; values are stored in the hash table itself.Simple; no extra structures needed.Works fairly well when load factor is low.However, a low load factor wastes space.Because colliding elements remain adjacent in memory, caching behavior is exceptional.Disadvantages:Performance swiftly degrades when load factor exceeds 0.8.Collisions may cluster, and this requires traversing the hash table one element at a time to find the next available space. This may slow insertion.

CRUD hash tables insertion, accesss, deletion on average is theata(1). insertion, accesss, deletion on worst case is theata(n).

GRAPHS path/ a sequence of vertices that connect two nodes in a graph. Complete graph/ a graph in which every vertex is directly connected to every other vertex. Weighted graph/ a graph in which each edge carries a value. DFS/ travel as far as you can go, back up when you hit a dead end, can be made with a stack. BFS/ look at all the paths at the same depth before you go at a deeper level, can be made with a queue. Shortest path/ the path whose total weight is the minimum.