Exercises

Arreglo con numeros los numero mas 2 o menos 2 esta dentro de la matriz con hashtable

All permutations of a string, take 1 character and add it to each position of the previous small string starting with one char until all chars are used

Find elements duplicated in 2 arrays, send one to a map and then look each from 1 array in that map

Es primo pag 51

Hashtable con hashcode pag 88

Examples page 90

Cv

Have the most interesting part

Accomplished x by implementing y which lead to z

Reduced object rendering time by 75% by implementing distributed caching leading to a 10% reduction in log time

Not years of experience more expert fluent proficiency, etc

Develop projects in new languages

From mi previous projects:

Challenges, mistakes/failures, enjoyed, leadership, conflicts, what youd do differently.

1-2 Projects to talk about in detail

Challenging interaction

Team work 2 nice 1 was struggling he rarely participate in meetings and struggled to complete his components. So It was shifting more work on us and we were not sure if we could count on him

First I wanted to understand why, he was struggling in understanding the big picture and how the components interact making him to go slow, so we sat and had a deep going through the principal component and going in detail, after he undertood how they fit together then he started delivering quickly and on time.

BIG O

Is the language and metric we use to describe the efficiency of algorithms

Ways to describe a scenario best case, worst case and expected case

If your algorithm is in the form do this then when you are don do that then you add the runtimes O(A+B)

If your algorithm is in the form do this for each time you do that then you multiply runtimes O(A\*B)

When you see a problem where the number of element in the problem space gets halved each time that will likely be a O(log N) runtime

Hashtable

Is a data structure that maps keys to values for the highly efficient lookup.

1 compute hashcode, 2 map hashcode to an index in the array, store the key and and value in the index

2 different keys could have the same hashcode and 2 different hash codes that map to the same index.

To retrieve the value pair by its key, compute the hashcode from the key, compute the index for that hashcode and in that index search through the linked list for the value with this key

Arraylist, resizable array which doubles its size when it is full.

String

String sentence

Sentence = sentence + “fer”

On each concatenation a new copy of the string is created and there the 2 strings are copied over char by char, String builder avoids this issue creating a resizable array of all the strings.

LinkedList

is a data structure that represents a sequence of nodes, single each node points to the next, double linked list gives each node pointers to the next and previous nodes.

If you want to access one node you need to traverse the nodes

The benefit if to add and remove items from the beginning of the list in constant time

Stack

Uses lifo

Queue

Fifo

Tree

Data structure composed of nodes

Trees have a root node with zero or more child nodes, each child has zero or more child nodes

Nodes may or may not be in a particular order, can have any data type as value and may or may not have links back to their parent nodes

Binary trees

Each node has up to 2 children

Node is called leaf if it has no children

Binary search tree

Every node fits a specific ordering property; all descendants from n must be on left side and bigger on the right side

Complete binary tree

Every level of the tree is fully filled except for the last level, its filled left to right

Full binary tree

Binary tree which every node has zero or two children

Perfect binary tree

It Is both full and complete, all leaf nodes will be the same level

Binary tree traversal

In order Traversal, visit left branch, then current node then right branch

When performed on a binary search tree it visits the nodes in ascending order

Preorder traversal

Visits the current node before its children, root is always the first node visited

PorsOrder traversal

Visit the root after its children, root always the last node visited

Min heap

A complete binary tree where each node is smaller than its children, the root therefore is the smallest element in the tree

We have 2 operations, insert and extract, when we insert we start at the bottom from the right side, we then swap the new element with its parent until we find and appropriate spot

Extract: remove the minimum element (root) then swap it with the last element in the heat, then bubble it down with its children swapping places until finding the right spot

Trie

Variant of an n-ary tree in which characters are stored at each node, each path down the tree may represent a word

\*nodes sometimes called null nodes are just to indicate complete word

Usually used to store the entire language for quick lookups, many problems involving list of valid words use trie as an optimization. In situations when we look for the tree on related prefixes repeatedly we might pass around the reference in the current node in the tree, so we can start from the node rather than from the root each time.

Graph

A collection of nodes with edges between some of them

Can be directed (one way street) or indirected (2 way street)

If there is a path between every pair of vertices it is called connected graph

A graph can or not (acyclic graph) have cycles

Adjacency list

The most common way to represent a graph, every vertex or node stores a list of adjacent vertices, in an indirected graph, an edge like (a,b) would be stored twice, one in a´s adjacent vertice and once in b´s adjacent vertices.

Class Graph{

Public Node[[ nodes;

Class Node {

Public String name;

Public Node[[ children

Adjacency matrices

It is an NXN matrix where n is the number of nodes, where true value at matrix[i[ [j[ indicates an edge from node I to node j.

In an undirected graph an adjacency matrix will be symmetric, in a directed graph it will not necessarily be

Graph search

Depth first search

We start at the root or another arbitrarily selected node and explore each branch completely before moving on to the next branch, we go deep first before we go wide.

Preferred if we want to visit all nodes in the graph (bit simpler)

Key to validate if the node has been visited to avoid infinite loop

Breadth first search

We start at the root or another arbitrarily selected node and explore each neighbor before going on to any of their children, we go wide first before we go deep.

If we want to find the shortest path or any path between 2 nodes bfs is generally better,

Object Oriented Design

Who is gonna use it, how it is going to be used, where, how, when, why

Define core objects

Ex: restaurant : table, guest, order, meal, employee, host, etc.

Analyze their relationships.

Which objects are members from another objects, any inheritance?, many to many or one to many relationship.

Ex: party array of guests, server and host inherit from employee, each table has a party but a party can have multiple tables.

Consider key actions that the objects will take and how they relate to each other.

Deck of cards

Deck class with actions shuffle, deal card, deal hand

Class Hand with array of cards, add card

Class Card wih value and suit (type card), methods is available, mark available, mark unavailable