<https://quizlet.com/_av6boa?x=1jqt&i=46sdbn>

**Pro and cons of merge sort alghortim**

The merge sort algorthim allows to sort a list of items in a recursive way. The main cons are the fact that is requires a huge amount of resources rather than other alghortim. But merge sort uses also a divide and conquer strategy that is faster than others sorting algorithms.It has a computational complexity of O(n log n) that makes it better than bubble sort, selection sort and insertion sort.

**Difference between Queue and Priority Queue**

Queue are FIFO (First in first out). The order of arrivial in the queue defines also the order of leaving. On the other hand priority queue have a priority that defines the order. Highest the priority of the item closer will it be located at the beginning of the queue.Higher priority leades to be removed before the others elements

**AVL advantages respect BST**

AVL trees can be consdier to be a subset of BST. AVL Trees are binary trees that try to limit as much as possible the heigh of the tree. AVL makes sure to remain balanced after there is an insertion or deletion of one item of the tree and they impose some limitations regarding the fact that the difference of high of the left and right subtree can not be more than 1

**- Depth first search (DFS)**

Depth first search is used for graphs and try to explore as much as possible every branc before exploring another one. It takes account of the discovery and finishing time, that is a counter for each node that shows how many steps where necessary to discover the node and how many to finish it (that is that all its possible connections are disocvered)

**Sequential (or linear) search**

Sequential search (also known as linear search) is the most simples and also less efficient searching alghortim. The average case O(n) time to detect if an item is present or not in a list. Linear search is based on checking one by one each element in a list and return True if it encounter it, false if it reach the end of the list and it was not detected.

**How to extract a min element**

By defintion of the BST the min element is the one on the root of the tree. When it is removed it is replaced by the last element in the list. Then it is swapped with the child that have the lowest value until both child are bigger then its value. In this way the item reaches it final correct position

**What is a binary tree**

A binary tree is a particular type of tree where each node can have at most two child nodes that are called left and right child.

**Colours of BFS and DFS**  
There are three differnt node’s colore for Breadth First Search tree and Depth First Search: white, gray and black.

White nodes are nodes not yet discovered and all the nodes are initialised as white. Gray nodes are nodes that are discovered, aka visited for the first time, by not yet explored, the have edges reaching white and/or gray nodes. Black nodes are nodes that are explored and they have edges reaching only gray and black nodes.

**Describe the stack ADT**

Stacks are a define order of items where each item can be added and removed only from one head, this is called (LIFO) that is last in, first out. A daily example of stack can be a stack of book or a stack of plates

**What is a queue?**

A queue is a defined order of items where items are added from one head and they are removed from the other one. This is also called FIFO, that is first in, first out. A daily example can be a queue of person in a grocery store.

**Bubble sort**

The bubble sorting alghortim is consider to be the easiest and also less efficient alghortim for sorting items. Bubble sort is based on confronting adjacent items two by two (that is the “bubble” of the name) and if they are out of order they are swapped. At the end of each iteration the biggest item out of order is located in the correct position and so the list requires n-1 iterations to be sorted. The average time is O(n^2)

**Delete a node in binary search tree**

Deleting a node in binary search tree can happen in three different cases. If the node have only one children then the children node subsitutes the removed node. If it has two child then we must choose the leftmost node of the right child to be moved. If the node does not have child (that is a leaf node) that it can be removed with no problems

**What is the Minimum Spanning Tree**

The minimum spanning tree is a tree that connect all the nodes in a graph without any cicles and where the weight of the graph is minimized as possible. it needs to be an undirected graph. Also a graph can have multiple spanning trees

**Adjacency matrix**

An adjacency matrix is one of the possible way to store the graph’s information (the other one is an adjacent list). An adjacency matrix is a square table where on the first row and first colums are all the vertices of the graph (so the table size is V^2) and the weight of the graphs is expressed where the two vertices line and columns intersect each other. If a grid is empty or there is 0 then it means that there is no edge between those two vertices. The adjacency matrix have some cons because it is not very efficient with sparse graph but it is efficient with dense one. It is also very easy to read and requires O(1) time for adding or removing an element.

**- Binary search**

Binary search works only for ordered list and it is the best searching algorithm. Binary search works recursively and use a divide and conquer strategy. When we are looking for an item we choose the one that is in the center of the list. If it is what we are looking for we finish, but if it is smaller we remove the second half of the list, while if it is bigger we remove the first part of the list. The computational complexity is of O(log n)

**Self-balacing tree**

A self-balacing tree is a particular kind of binary tree that maintains keeps the tree high as small as possible. As an example AVL Tree and black-red trees.

**What are the advantage of an AVL tree respect to a generic BST**

The AVL share the same characteristics of BST but it add one requirement more. AVL trees need to have the difference of the high of the left and right subtree not greater than 1 or smaller then -1 this is called balance factors. If it wasgreater then one it would require to perform some changes than can be left or right rotation.

**What is an adjacency list**

An adjacency list is one of the two possible way to store a graph (the other one is the adjacency matrix). An adjacency list store the information in a list of the lenght of the number of vertices. It is good for sprase graph but less efficient for dense graph and it requires as size O(|V| + |E|)

**Strongly connected component of a directed graph**

Given a directed graph a strongly connected component is a subset of the graph that is strongly connected

**Height balanced**

A tree is height balanced if the difference between the left and right subtree of each node is not more than 1

**Dijkstra’s alghortim**

Dijkstra’s alghortim is used to find the most efficient path that conencts two nodes in a graph. At first the Dijkstra’s alghortim discover the starting vertex and then it chooses the edge with the lowest weight and proceeds to find the end vertex. Dijkstra take in account the total weight while BFS the number of edges. It is a greedy alghortim (that it that given a choice it chooses the best opinion available at the moment)

**Rules of red-black tree**

The red-black tree rules are. The root node and the leaves (NIL) are always black. All the nodes are either black or red. A red not must have black child. All descendant paths from a node to a NIL have the same amount of black nodes

**What is a priority queue?**

A priority queue is a queue that gives a priority to all the items enlisted. The order of the queue depends striclty on the priority and not from the arrival order. In this way the elements with an higher priority will be first in the queue and removed before the others

**Definition of complete binary tree**

A complete binary tree is a tree in which all the nodes have both the left and right child except for the last level that can be filled from left to right.

**Explain the quick sort alghortim**

The quick sort algorithm is a recursive alghortim that take advantage of the divide and conquer strategy. It is the best alghortim for sorting. At first the pivot is choosed, it is the first element of the list (it can be choose in different ways). Secondly we have a left and right marker. The left marker is located at the beginning adjacent to the pivot, while the right marker starts from the end of the list. The left marker starts checking one by one the items of the list until it finds one that is bigger than the pivot value, then the right mark start checking the items in the opposite order and it stops once it enconter an item that is smaller then the value of the pivot. After this the items of the left and right marker are swapped and they start again. We stop when the right marker is located before the left marker. When this happens the pivot and the item of the right marker swap position and now the pivot is located in the correct position.

**What is a deque?**

A deque can be consider to be a particular type of queue where the items are enqueued and dequewed from both ends (fron of rear). It is an hybrid between a stack and a queue

**Which are the two most common ways to implement a graph**

There are two ways to implement a graph: adjacent matrix and adjacent list.

The adjacent matrix is a matrix of size V\*V where V is the number of vectors in the graph. Inside the matrix there are the weights of the graph. If a cell is empty or it have a 0 then there is no edge between the two vertices. The other possile graph implementation is the adjacent list, that is a list a master list containing all the vectors and the weights.

The pros of the adjacent matrix is that it requires O(1) time to add or delete a weight and it works efficienlty for dense graph (graphs with lots of weights and vectors) it is also very easy to read but it works unefficienlty for sprase graph and requires high memory usage O(V\*V)

On the other hand the adjacent list requires O(|V|+|E|) time and it works better for sprase graph rather then dense graphs. but requires O(|V|) to check if a vector exist and to remove it

**How merge sort works**

Given a list the merge sort alghortim split the list in sublist until all the sublist have size 1 and so they are considered to be sorted. Then the merge sort alghortim start by connecting different list and ordering them until we reach back a single list of the same size as the starer one. This alghortim is better then bubble sort, selection sort and sort but have on average lower performance than quick insertion sort. Also it is a recurisve alghortim that uses the divide and conquer strategy. It have a computational complecity of O(n log n)

**AVL tree**

An AVL tree is a particular type of binary search tree that takes in account the balance factors (BF) that state that the high difference of the left and right subtree of each node needs to be not greater then 1. It makes sure that a tree remains balance after insertion or delation

**- Breadth First Search (BFS)**

Are binary tress where each node have as left child a smaller node, and as right child a bigger node. Search, insert and delation are on average O(log n)

**Pros and cons of adjaceny matrix**

The pros of the adjacenty matrix are that it well readable and requires O(1) for searching or deleting a weight. The cons instead focus more on the size of the matrix that increases the storage of O(V\*V) and it performs poorly for sparse graph while it performs better for dense graph

**- Topological sorting**

Takes direct acyclic graph and create a linear o dering of the vertex. Multiple topoligical are possible of the same graph. To do so ou need to run DFS to compute finishing time and to sotre in decreasing order of finshing time

**Red-black tree**

Are a specific type of Binary search tree that follow some more specific rules. All the nodes needs to be either black or red, the root node and the leaves (NIL) are black and a red node must have black children. Also from a node the shortest path to the NIL have the same number of black nodes of the longest path to the NIL.The average complecity is O(log n) and it is a self balacing tree

**Definition of binary heap**

A binary heap is a binary tree that is completed (all its level are complitely filled) and it is a min/max heap that is that all the child nodes of a node needs to be smaller/bigger then the node. It can be implemented by using a list that makes easier to find the position of parent and child

**What is a list**

A list is a collection of items that are stored in a specific order. It is possible to add a new element, move it or deleting and the elements appendned in the list can be homogeneous.They old a relative posotion with repsect to the others. It can be ordered, unoreder or linked

**What is a stack**

A stack is a collection of items that are store in an order, as a stack of books or place we can add new elements only at one end and we must remove them from the same end. This is called LIFO that is last in, first out

**Law of recursion**

There must be a base case, the recursion must get closer to the base case and call itself recursively

**What is a queue?**

A queue is a collection of items that are stored in a specific order. Items are enqued from one end and dequeued from the other one (this is called FIFO, first in first out), a daily example is the queue at a supermarket.

**Is Dijkstra best alghortim to find the shortest path?**

No. The best alghortim is A\* that is otpimally efficient and use heurstic to find the shortest in a graph froma vertex to a destination vertex. Actually it depends because A\* is an alghortim that is consider to be an extension of Dijkstra alghortim and also the heurstic needs to be correct, otherwise the result must be worst than Dijkstra

**Sorting methods**

Bubble sort, selection sort, insertion sort, merge sort, quick sort

**- Shortest path**

Given a graph the shortest path from one vertex to another is the path that minimize the sum of the weight

**- AVL vs Red-black tree**

AVL trees are consider to be a subset of red-black tree because they require more balance but least to an higher number of rotation during insertion or deletion. Red black tree better if we have few edits. AVL better if we need to perform search more often then insertion or deletion

**Insert new element in min heap**

Add the new element at the end of the list (on the tree left before right), then move it upwords until it is not anymore smaller then its parent node

**Type of traversal**

Preorder, inorder, postorder

**Quick sort**

Quick sort is the most efficient sorting alghortim that we encountered.   
The quick sort alghortim start by choosing the pivot element that it the first element of the list (there can be also other way to choose it). Also the left marker is located next to the pivot element and the right marker is instead located at the end of the list. The left marker start to move towards the end of the list and it stop when it finds an element that is bigger than the pivot element, then the right marker start apporaching towards the beginning of the list and it will stops when it found an element that is smaller than the pivot element. This is called split poind and it is when all the elements before the piovt are smaller, while the one after are bigger. When this happend the element of the left and right marker are swapped and this keeps repeating until the right marker is located before the left marker. When this happens then the pivot and the right marker elements are swapped and the alcgohortim starts a new iteration. The quick sorting algorithm is recurisve.

**Insertion sort**

Sorting alghortim that have a computational complexity of O(N^2) and that is better than bubble sort and insertion. This alghortim work by having sorted a sublist and sorting one by one each element. At first there is only one element and the list is considered sorted

**Merge sort**

Merge sort in an alghortim that work recursively and take advantage of the divide and conquer strategy. It has a computational complexity of O(n log n). At first the merge sort alghortim split the list in half by creating two of them, then it recursively keep doing this until there are n sublist of size 1. Later it will start mering and adding all the list back togheter by sorting them until there is only one list of size 1.Consist in taking two smaller sorted list and combine them toghter in a single sorted new list.

**- Kosaraju alghortim**

**Run DFS and compute finishing time of each vertexm then compute G^t by reversing the direction of all the edges and consider the vertices in decreasing finishing time. Each tree in the forest computed is strongly connected component**

**What is an Abstract Data Type (ADT)?**

An abstract data type is a collection of items that are organised in a specific order is a type for object whose behaviour is defined by a set of values and a set of operations

**Minimum spanning tree**

Given an undirected graph the mimum spanning tree is a tree that is able to connect all the nodes of the graph by having the lowest al possible weight of the edges and that is acyclic

**Prim’s alghortim**

Prim’s alghortim focus on fidning the most efficient way to connect all the nodes. To do so it focus on conencting two nodes with the lowest possible weight (from all the discovred nodes).It is a greedy alghortim and it find always the lowest weight

**Defintion of percetly height balaced**

A perfeclty height balanced tree is a tree where from all the nodes the left and right subtree have the same hiehgt

- **Min or max hepify**

Is the process of reshpaing a binary tree in a min or max heap

**Search methods**

Sequential (linear) search

Hashing

**Selection sort**

When a list is given the selection sort alghortim work by checking the list and finding the biggest item, then it is moved to the end of the list. Instead of the bubble sort that requires a lot of swap the selection sort requires only one swap for each interation. The computational compexity is O(N^2)

**Hashing**  
Hashing is a searching alghortim that storage all the items in a hash list and then it can recall them in O(1) computational compexity but only when the number of entries are large. To do it needs an hash fucntion that is able to create a relationship between the items and the hash location in the list. As an example the remainder hash function work by finding the remainder betwen an item and the index of the hash function. The hash function might not be perfect and so it is required to take in account the possibilibity of a conflict between two items

**Linked list**

A linlked list is a particualr type of list where the order of the items is not bases on the location in the list but rather in a relative way by the element before and after and so it does not needs contingous memory positioning

**Pros and cons of adjaceny list**

The pros of adjaceny list is the size that is O(|v| + |E|) and it requires O(1) to add a vertex or a weight. Adjaceny list are also efficient for sprase graph (graph with few nodes and edges). We also need to take in account that the ajdacny list are worst for dense graph (it is better in this case to use a matrix) and it is less easy to read also removing an edge or vertiex if requires O(|V|)

**What is a recursion**  
A recursion is a program that is able to recall itself during the code. It have three rules, it needs to recall itself, get close each interation to the base case and change its state to more towards the base case. Recursion alghortim use a divide and conquer strategyt. It is focus on breaking a problem in smaller pieces until it can be solved trivially

**Definition of strongly connected**

Given a graph it is strongly connected if it is possible to find a path from each node to all the others nodes of the graph

**A\* alghortim**

The A\* alghortim uses heursist and it is consier to be an implementation of Dijastra alghortim. It is base on heuristic that gives an idea about how to find the shortest path.It is considered optiammly efficient

**- Definition of binary search tree**

It can havw 2 subtree max, if the key is less then the parent key is in the left subtrer, if it id bigger it in the right subtree