**Definition of stongly connected component of a direct graph**

A strongly connected component of a direct graph is a subgraph that is strongly connected.

**Sequential (or linear) search**

A sequential or linear search is a search alghortim that hav a average search case of of O(n) and works by checking all the items in a list in order one-by-one if the item that we are looking for is detected the alghortim will return true, instead if it reaches the end of the list without finding the item it will return false. It is the easiest search alghortim

**Definition of height balanced**

An height balanced graph is a type of graph where all its nodes have the difference of the left and right subtree height not more than 1 and not less than -1

**Dijkstra’s algorithm**

The dijkstra’s algorithm is an alghortim that works for directed and undirect graph. It is based on finding the shortest path from one node to another that have the sum of the weight minimized. When a path from one node to another is found with the Dijkstra algorithm it is possible to connected also all the others nodes and create a spanning tree. This tree is not guaranteed to be the minimum spanning tree. It is a greedy alghoritm, so it chooses always the best possible outcome

**What is the difference between a Queue and a Priority Queue?**

A queue is a collection of item that in entirely orderd by the arrival of the items. Queue are an Abstract Data Type (ADT) that is FIFO (First In, First Out) that have two ends, one for appending one for removing. On the other hand a Priority Queue is still an ADT but it is takes into account the priority that each items have. In this way the order of remova of a Priority Queue differs substantially of the one of a Queue becuase the items with the highest priority will be removed before the others.

**Rule of red-black tree**

1. A node can be either black or red
2. A red not must have a black node has child
3. The root and the leaf ndoes (called NIL) must be black
4. Given a node the shortest path for reach the leaf node si half the lenght of the longest one and contains the same number of black nodes.

**What is a priority queue?**

A priorty queue can be consider as a particuar type of queue. It is based on the idea that the items are orderd not by the order of arrival (as it is done with the “strandard” queue) but by their priority. By doing so the item that is the next one in line to be removed from the priority queue is the one with the heighest priority

**What are the pros and cons of the merge sort algorithm?**

The merge sort algorithm it is a recursive alghorithm that takes advantage of the divide and conquer strategy. Since this it requires more space to run property but it is also more efficient that bubble sort, insertion sort and selection sort. On the other hand it is less efficient than Quick sort that is also a recursive alghortim

**Complete binary tree**

A complete binary tree is a tree where each node can have at maximum two child nodes called left and right node. Also to be complete all it’s level must be filled except for the last one that needs to be filled from left to right. A subset of the complete binary tree is the perfeclty height balanced tree, where all it’s level are filled and so the amount of nodes of the tree is always equal to 2^(n+1) -1 where n is the height of the tree

**How the quick sort alghortim works**

The quick sort alghortim is a recursive sorting algorithm that takes advantage of the divide and conquer strategy. The pivot is, usually, the first item of the list (there can be also others way to choose it), while the left marker starts adjacent to the pivot and the right marker starts at the end of the list. At first the left markers move towards the end of the list and it stops as soon as it encounters an item that is more then the pivot items, then the right markers move toward the pivot and it stops as soon as it finds an item that is less than the pivot. When this happens the left and right items are swapped location. And this procedure is repeated until the right marker is before the left markers. When this happens the piovt item is swapped with the right item and we have found the split point. Now we know that the on the left size of the piovot there will be oly smaller items, while on the right there will be only bigger

**What is an adjacecy list**

An adjancy list is one of the two possible ways to store a graph. It works by saving in a master list all the nodes and all the weight of the edges between the nodes. It is optimally efficient O(|V|+|E|) and it works well for sprase (with few edges and nodes) graph. On the other hand, it is less efficient for dense graph and it requires O(V) time to remove a vertex or to verify its existence

**Whar is a queue?**

A queue is an abstract data type (ADT) that works by saving items in a precise order. Rather than a stack that is LIFO a queue is FIFO, that means First In, First Out. So the first item added will be also the first one to be removed. A common daily example of queue is the one in a grocery shop. It has two ends, one for adding items and one for removing it. A particuar kind of queue is the priority queue that order the items depending on their priority.

**What is a deque?**

A deque can be considered the hybrid of a stack and a queue. It have two different ends and it possible to add and remove items from both ends. It is known also as a double endned queue

**What are the two most common ways to implement a graph?**

The two most common ways to implement a graph are adjacent matrix and adjancent list. The adjacent matrix is a matrix of size (V\*V) with expressed in the grid the weight of the edges between the nodes. If a grid is empty or there is a 0 it means that there is no edge between the two nodes. The adjacent matrix is well readable and requires O(1) to add or remove an edge, it also works well fro dense graph (graph that are full of edges or nodes). On the other hand adjacent matrix have a less efficent performance for sparse graph and they requires an higher capacity.

Adjacent list is the other way to save a graph. It saves the edges and the weights in the master list and it is more efficient for sparse graph. And requires O(1) to add an edge. Adjacent list have a lower performance for dense graph to remove and get a node. Also it is particullary memory efficient because it requires only O(|V|+|E|)

**Describe (schematically) how the Merge Sort algorithm works**

The merge sort alghortim is a recursive algorithm that takes advatange of the divide and conquer strategy. It starts with a list than then is recursively splitted in half until it reaches n list of size one, those list are considered to be sorted. Then the merge strategy is applied and all the list starts to be re-merged toghter and while doing this the sorting is completed

**AVL Tree**

AVL are a particualr kind of binary search tree (if the item is smaller go left, if bigger go rigth) that keeps the tree balanced after insertion or deletion. To do so it uses the balance factor (called BF) that is equal to the difference between the left and right subtree of the node. The balance factor of all nodes is calculated every time it is added or removed a node from an AVL tree and if a node results in having a balance factor that is less then -1 or more then +1 we apply rotation to reduce the comlexityof the tree.

**Breadth First Search (BFS)**

Breadth First Search (BFS) are applied to graph to find the shortest path from a node to another. It follows the idea to explore all the vertices at distance k before fiding all the one that are at distance k+1. It all the nodes are discored it produces a breath first tree. Also they are usually represented with 3 colors: white, gray and black. White means that the node is not discovered and all the nodes are initialised as white. Gray nodes instead are the one that are discovered but not yet explored. While black nodes are nodes that are explored and can have edges only to black or gray nodes.

**Pros and cons of adjaceny matrix**

Adjaceny matrix are one of the two possiblw way that we have studied on how to save a graph (the other one is adjacent list). Adjacent matrix have the advatange of beeing well readable and also they requires O(1) to check the existence of an edge, also they have a better performance in dense graph (graphs that are full of nodes and edges). On the other hand they have a lower performance in sparse graph and they require bigger storage space of O(V\*V)

**Topological sorting**

Takes Direct Acycling graph and produces a linear ordering of all its vertices. Multiple topological are possible on the same graph

**Red-black tree**

Red-black tree are a type of Binary Search Tree (BST) that are particually efficient for adding or removing a node. Red-black tree (since they are vinary search tree) try to minimize the height of the tree. It follows some rules that defines the structure of the tree

1. a node can be either black or red
2. the root node and the leaf nodes (NIL) are always black
3. a red tree must have black child
4. the shortes descendand path from a black node to the leaf have 1/2 the lenght of the biggest one

They have also an average complexity of O(log n), they are self-balacing tree and have virtual leaf node (NIL) that do not have data or rules

**Defintion of binary heap**

A binary heap is a binary tree that is complete (all it’s level are filled except for the last one that is filled from left to right) and they are also min or max heap: that is that that given a node all the child are bigger/smaller. Also it can be implemented by using a list that makes easier to find the position of parent/child

**Minimum spanning tree**

Given an undirected graph the minimum spanning tree is the acyclic tree that connects all the nodes and mimimize as much as possible the sum of all the weights that connects the nodes. Also a graph can have multiple spanning tree, but only one minimum spanning tree