The idea of string similarity comes in Machine Translation, Information Extraction, Speech Recognition.

In machine translation, we might want to know how well a machine translation system does.

Suppose the machine translation system represented some sentence translated from Chinese.

We can check how many words were changed or inserted.

Here we are checking for complete words and not single characters.

Markov’s Chain.

Formula for finding number of non-adjacent subsets of given numbers.

Application of minimum edit distance.

HashMap.equals() method compares 2 hashmaps by key-value pairs.

It means both hashmap instances must have exactly same key-value pairs and both must be of same size.

The order of key-value pairs can be different and does not play a role in comparison.

For optimization:

If we are returning a boolean on some count

Instead of doing an if-else on the value of the count, return the boolean which has the comparison.

**DFS vs BFS**:

Heavily depends on the structure of the search tree and the number and location of solutions.

* If we know a solution is not far from the root of the tree, a breadth first search (BFS) might be better.
* If the tree is very deep and the solutions are rare, depth first search (DFS) might take an extremely long time, but BFS could be better.
* If the tree is very wide, a BFS might need too much memory, so it might be completely impractical.
* If solutions are frequent but located deep in the tree, BFS could be impractical.
* If the search tree is very deep we will need to restrict the search depth for depth first search (DFS),

BFS has an interesting property. It first finds all the vertices that are one edge away from the starting point, then all the vertices that are 2 edges away, and so on. This is useful if we are trying to find the shortest path from the starting vertex to a given vertex.

We start a BFS, and when we find the specified vertex, we know that path we have traced so far is the shortest path to the node. If there were a shorter path, the BFS would have found it already.

BFS can be used for finding the neighbour nodes in peer to peer networks like BitTorrent, GPS systems to find nearby locations, social networking sites to find people in the specified distance.

Breadth First Search is generally the best approach when the depth of the tree can vary, and you only need to search part of the tree for a solution.

For example, finding the shortest path from a starting value to a final value is a good place to use BFS.

Depth First Search is commonly used when you need to search the entire tree.

It's easier to implement (using recursion) than BFS, and requires less state:

While BFS requires you store the entire 'frontier', DFS only requires you store the list of parent nodes of the current element.

**IMP NOTE**:

In the collections we always use object types and not primitive types.

If we have List<Integer[ ]>

Arrays are of fixed size. So, to iterate over arrays we should use for-each loop.

Elements have to added in the array at the time of creation.

Adding elements into a list can be done anytime.

When thinking of using for-loops, think about why not while loop?