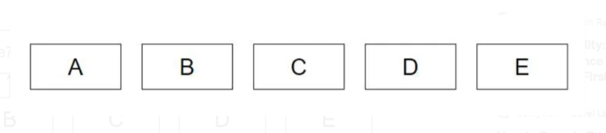
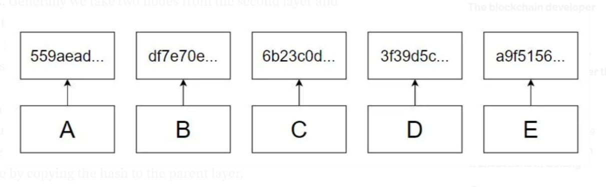
Insertion (1/6)

- Given a list of alphabets, lets create a merkel tree from it. The bottommost layer of the tree would contain all the letters as the leaf nodes.



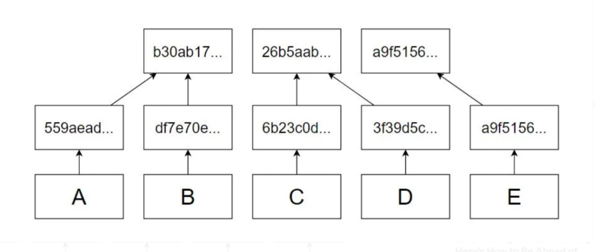
- The layer above contains its hash values.



Insertion (2/6)

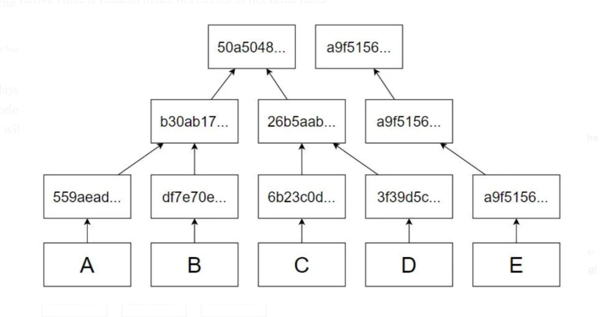
- The nodes in the layer after the second layer are contains the hash value of the child nodes. Generally we take two nodes from the second layer and combine them to form another node. We can take more than two nodes as well but binary merkel trees is the simplest of them all and increasing the degree of nodes only increase the computation and algorithms complexity.
- If we have even number of nodes, we take 2 consecutive nodes and form the parent layer. But if we have odd number of nodes, we take two consecutive nodes until one is left to form the parent layer, and then we repeat the remaining node by copying the hash to the parent layer.

Insertion (3/6)



- Similarly, the fourth layer is formed using the values of the third layer.

Insertion (4/6)



- The final layer or the root of the Merkel Tree is formed by hash value of the last two nodes remaining in top most layer. In any case, odd or even leaf nodes, we will always have two nodes in the top most layer.

Insertion (5/6)

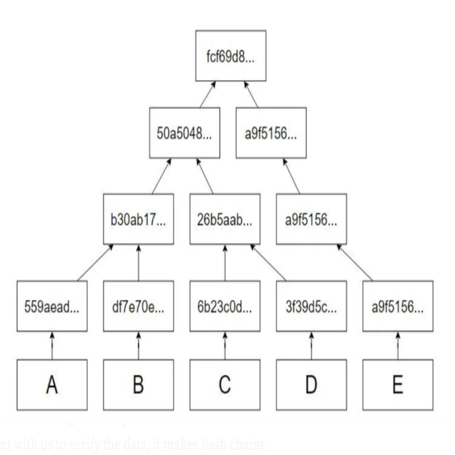


Figure: This is the Merkel tree of the given data

Insertion (6/6)

Merkle root is a representative of the Merkle tree.

1. It acts as a digital fingerprint of the data set represented by the tree.
2. Remember that even a slight change in the data will change the hash value of the leaf and this will change the hash values of all the nodes connecting that leaf node to the root node, eventually resulting change in the root hash