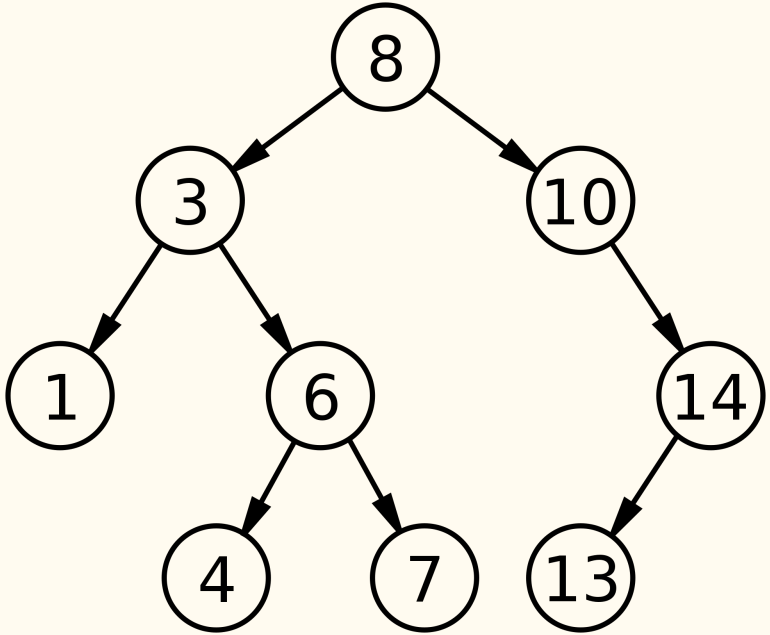
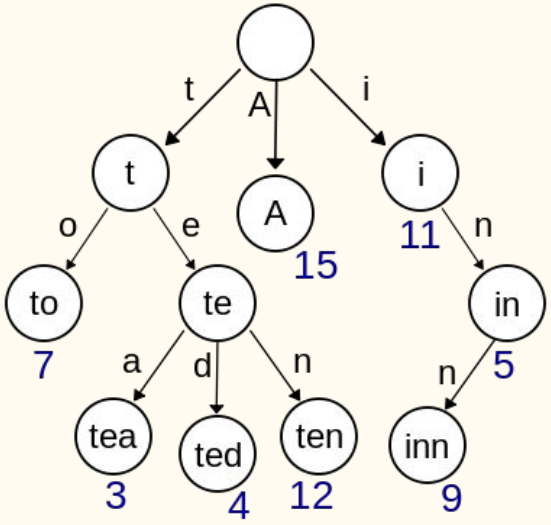
Side Note - Trees



Trie Data Structures

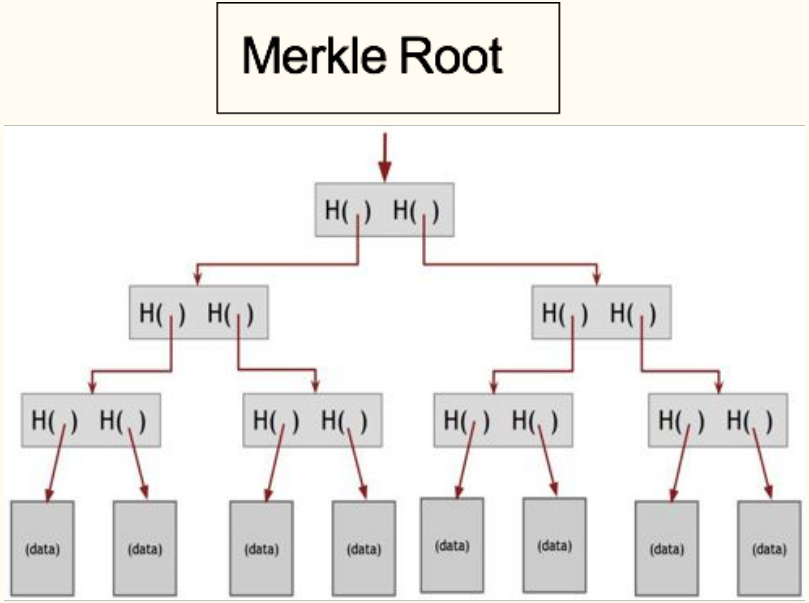
- A trie, also called digital tree, radix tree or prefix tree is a kind of search tree
- An ordered tree data structure used to store a dynamic set or associative array where keys are usually strings



Merkle Trees | An Introduction

- Primary purpose is to prove the consistency of data and is essentially a tree of hashes
  - Blobs of data are hashed at the bottom
  - Hashes are then hashed together into a binary like tree
- Merkle trees allow for proof-of-membership, that is, its easy to tell that certain data blocks are members of the merkle tree.
- To prove inclusion of data in the Merkle tree, provide the root and its intermediate hashes
- To fake the proof you would need to find hash preimage, second preimage resistance meets this qualification. Thus very difficult.

Merkle Example



Source: Princeton Textbook Figure 1.7

Merkle Tree Advantages

1. Data consistency / Verification
2. Merkle tree proofs are computationally fast and easy
3. Merkle tree proofs require only a small chunk of data to be broadcasted across a network
4. Integral for distributed systems

Ethereums Trie Data Structures

- Ethereum has 4 trie data structures
  - Receipt tree
  - State tree
  - Storage Tree
  - Transaction Tree
- The root of each trie is a Keccak 256-bit hash
- Receipt tree, State tree and Transaction Tree all exist within the block header
- Storage tree root lives within the RLP encoded data value within the State tree