

# BLOCKCHAIN TECHNOLOGY

## EXPERIMENT NO.03

9570

Experiment 3

Shreshth Shah  
60004220126  
BE comp

Aim: Implement merkle tree & compute hash of the merkle root in that tree.

Theory: Merkle tree also known as hash trees are a fundamental data structure in cryptography & blockchain technology. They provide an efficient & secure way to verify the integrity of the data by organizing it into a tree-like structure where each leaf node is a hash of data & each non-leaf node is a hash of its child nodes. The root of the tree is known as merkle root & seen as a unique fingerprint of all the data in a tree.

Merkle tree structure  
A merkle tree is a binary tree where leaf nodes such as leaf nodes contain the hash of a data block which has a transaction or a record. Non-leaf nodes, each non leaf node contains the hash of the concatenation of its 2 child node hashes.

Merkle root is the topmost node in a tree which contains a hash that represents data within the tree.

The merkle root is widely used in blockchain system to efficiently & securely verify large amounts of data. It allows any change in underlying data to be quickly detected by simply recalculating & comparing merkle root.

FOR EDUCATIONAL USE

```

from typing import List
import hashlib

class Node:
    def __init__(self, left, right, value: str, content, is_copied=False) -> None:
        self.left: Node = left
        self.right: Node = right
        self.value = value
        self.content = content
        self.is_copied = is_copied

    @staticmethod
    def hash(val: str) -> str:
        return hashlib.sha256(val.encode('utf-8')).hexdigest()

    def __str__(self):
        return (str(self.value))

    def copy(self):
        return Node(self.left, self.right, self.value, self.content, True)

class MerkleTree:
    def __init__(self, values: List[str]) -> None:
        self.__buildTree(values)

    def __buildTree(self, values: List[str]) -> None:
        leaves: List[Node] = [Node(None, None, Node.hash(e), e) for e in values]
        if len(leaves) % 2 == 1:
            leaves.append(leaves[-1].copy())
        self.root: Node = self.__buildTreeRec(leaves)

    def __buildTreeRec(self, nodes: List[Node]) -> Node:
        if len(nodes) % 2 == 1:
            nodes.append(nodes[-1].copy())
            half: int = len(nodes) // 2

        if len(nodes) == 2:
            return Node(nodes[0], nodes[1], Node.hash(nodes[0].value + nodes[1].value),
nodes[0].content+"-"+nodes[1].content)

            left: Node = self.__buildTreeRec(nodes[:half])
            right: Node = self.__buildTreeRec(nodes[half:])
            value: str = Node.hash(left.value + right.value)
            content: str = f'{left.content}+{right.content}'
            return Node(left, right, value, content)

    def printTree(self) -> None:
        self.__printTreeRec(self.root)

    def __printTreeRec(self, node: Node) -> None:
        if node != None:
            if node.left != None:
                print("Left: "+str(node.left))
                print("Right: "+str(node.right))
            else:
                print("Input")

            if node.is_copied:
                print('(Padding)')
            print("Value: "+str(node.value))
            print("Content: "+str(node.content))
            print("")
            self.__printTreeRec(node.left)

```

```

        self.__printTreeRec(node.right)

def getRootHash(self) -> str:
    return self.root.value

def mixmerkletree() -> None:
    n = int(input("Enter the number of blocks: "))
    elems = []
    print("Enter the blocks: ")
    for i in range(n):
        elems.append(input())
    print("Inputs: ")
    print(*elems, sep=" | ")
    print("")
    mtree = MerkleTree(elems)
    print("Root Hash: "+mtree.getRootHash()+"\n")
    mtree.printTree()

```

mixmerkletree()

```

File Edit Selection View Go Run ... Search
EXPLORER
> NO FOLDER OPENED
> OUTLINE
> TIMELINE
> PROJECTS
> MAVEN
Merkle.py x
C:\Users> djsce.student > Desktop > Merkle.py > ...
1 from typing import List
2 import hashlib
3
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
E
Inputs:
A | B | C | D | E

Root Hash: d21efe0fe7efb677c6f27a17211a50edea42d749ad20070f744ef7c39306b727

Left: bd144551df2d311a947adbb0fc4b4bfa7bd76ae0e60095d88a8f79a3e3986af6
Right: c095c7b19fcd1f26c74113ca6a5316a9dd494b54bf8a1dc03fb16bd098580e7ca
Value: d21efe0fe7efb677c6f27a17211a50edea42d749ad20070f744ef7c39306b727
Content: A+B+C+D+E+E+E

Left: b30ab174f7459cdd48a3acd15d0c9444fec2adcfb9d579aa154c084885edd0a
Right: 84a3defdb78365472065549822bb4e2fc0a1d50a9a73109251c0320dff0cef83
Value: bd144551df2d311a947adbb0fc4b4bfa7bd76ae0e60095d88a8f79a3e3986af6
Content: A+B+C+C

Left: 559aead08264d5795d3909718cdd05abd49572a84fe55590eef31a88a08fdffd
Right: df7e70e5021544f4834bbe64a9e3789f9ebc4be81470df629cad6ddb03320a5c
Value: b30ab174f7459cdd48a3acd15d0c9444fec2adcfb9d579aa154c084885edd0a
Content: A+B

Input
Value: 559aead08264d5795d3909718cdd05abd49572a84fe55590eef31a88a08fdffd
Content: A

Input
Value: df7e70e5021544f4834bbe64a9e3789f9ebc4be81470df629cad6ddb03320a5c
Content: B

```

```

File Edit Selection View Go Run ... Search
EXPLORER
> NO FOLDER OPENED
> OUTLINE
> TIMELINE
> PROJECTS
> MAVEN
Merkle.py x
C:\Users> djsce.student > Desktop > Merkle.py > ...
1 from typing import List
2 import hashlib
3
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Left: 3f39d5c348e5b79d06e842c114e6cc571583bbf44e4b0ebfda1a01ec05745d43
Right: a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58
Value: 3bec9d7061ca48c56ee3e2b73c56ef12182b25a8b3597c10788bf4e08094d002
Content: D+E

Input
Value: 3f39d5c348e5b79d06e842c114e6cc571583bbf44e4b0ebfda1a01ec05745d43
Content: D

Input
Value: a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58
Content: E

Left: a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58
Right: a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58
Value: 4441435e9da65331ce2eccf7aca694c30accbb8289111964f9948db710915d019
Content: E+E

Input
Value: a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58
Content: E

Input
Value: a9f51566bd6705f7ea6ad54bb9deb449f795582d6529a0e22207b8981233ec58
Content: E

```