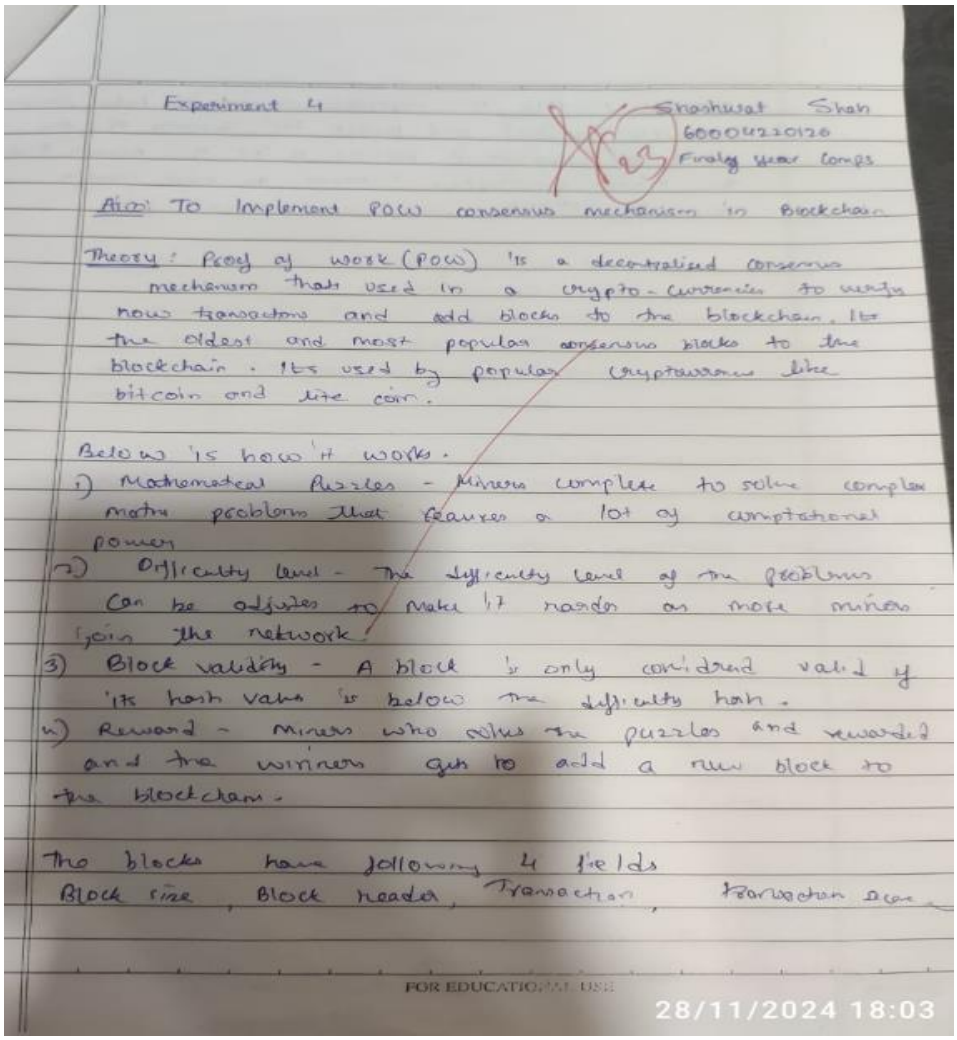
BLOCKCHAIN TECHNOLOGY

EXPERIMENT NO.04



Experiment 4

Shashwat Shah
60004220120
Final year comps

Aim: To implement POW consensus mechanism in Blockchain

Theory: Proof of work (POW) is a decentralised consensus mechanism that used in a crypto-currencies to verify new transactions and add blocks to the blockchain. Its the oldest and most popular consensus blocks to the blockchain. Its used by popular cryptocurrencies like bitcoin and lite coin.

Below is how it works.

1) Mathematical Puzzles - Miners complete to solve complex maths problems that requires a lot of computational powers
2) Difficulty level - The difficulty level of the problems can be adjusted to make it harder as more miners join the network.
3) Block validity - A block is only considered valid if its hash value is below the difficulty hash.
4) Reward - Miners who solves the puzzles and rewarded and the winners gets to add a new block to the blockchain.

The blocks have following 4 fields
Block size, Block header, Transaction, Transaction Size

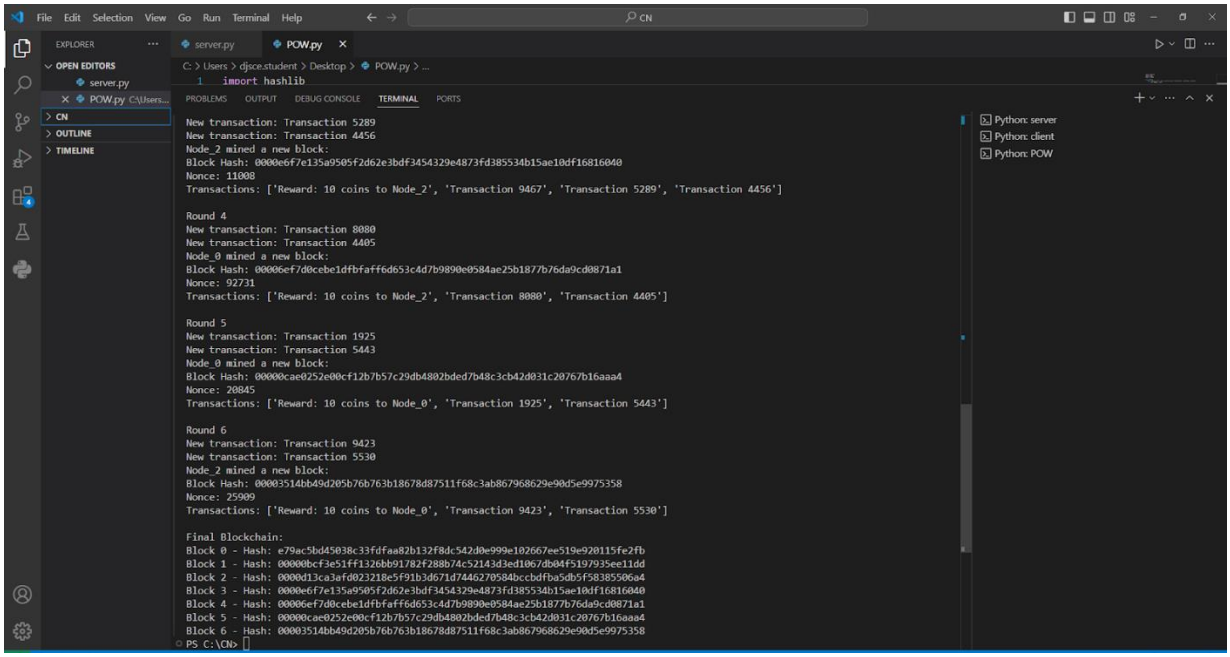
FOR EDUCATIONAL USE

28/11/2024 18:03

CODE & OUTPUT :-

```
#Simulate Consensus Mechanism
import hashlib
import time
import random

class Block:
    def __init__(self, index, previous_hash, transactions, timestamp,
nonce=0):
        self.index = index
        self.previous_hash = previous_hash
        self.transactions = transactions
        self.timestamp = timestamp
        self.nonce = nonce
        self.hash = self.calculate_hash()

    def calculate_hash(self):
        block_string =
f"{self.index}{self.previous_hash}{self.transactions}{self.timestamp}{self.non
ce}"
        return hashlib.sha256(block_string.encode()).hexdigest()

class Blockchain:
    def __init__(self, difficulty=4):
        self.chain = [self.create_genesis_block()]
        self.difficulty = difficulty
        self.pending_transactions = []

    def create_genesis_block(self):
        return Block(0, "0", [], int(time.time()))

    def get_latest_block(self):
        return self.chain[-1]

    def add_transaction(self, transaction):
        self.pending_transactions.append(transaction)

    def mine_block(self, miner_address):
        block = Block(len(self.chain), self.get_latest_block().hash,
                      self.pending_transactions, int(time.time()))

        while not block.hash.startswith('0' * self.difficulty):
```

```
            block.nonce += 1
            block.hash = block.calculate_hash()

        self.chain.append(block)
        self.pending_transactions = [f"Reward: 10 coins to {miner_address}"]
        return block

def simulate_pow_consensus(num_nodes=3, num_rounds=5):
    blockchain = Blockchain()
    nodes = [f"Node_{i}" for i in range(num_nodes)]

    for round in range(num_rounds):
        print(f"\nRound {round + 1}")

        # Simulate transactions
        for _ in range(random.randint(1, 3)):
            transaction = f"Transaction {random.randint(1000, 9999)}"
            blockchain.add_transaction(transaction)
            print(f"New transaction: {transaction}")

        # Simulate mining competition
        winner = random.choice(nodes)
        new_block = blockchain.mine_block(winner)
        print(f"{winner} mined a new block:")
        print(f"Block Hash: {new_block.hash}")
        print(f"Nonce: {new_block.nonce}")
        print(f"Transactions: {new_block.transactions}")

    print("\nFinal Blockchain:")
    for block in blockchain.chain:
        print(f"Block {block.index} - Hash: {block.hash}")

# Run the simulation
simulate_pow_consensus()
```


File Edit Selection View Go Run Terminal Help
CN
EXPLORER
OPEN EDITORS
server.py
POW.py C:\Users...
CN
OUTLINE
TIMELINE
server.py
POW.py
C: > Users > djsce.student > Desktop > POW.py > ...
1 import hashlib
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
Python: server
Python: client
Python: POW
New transaction: Transaction 5289
New transaction: Transaction 4456
Node_2 mined a new block:
Block Hash: 0000e6f7e135a9505f2d62e3bdf3454329e4873fd385534b15ae10df16816040
Nonce: 11008
Transactions: ['Reward: 10 coins to Node_2', 'Transaction 9467', 'Transaction 5289', 'Transaction 4456']
Round 4
New transaction: Transaction 8080
New transaction: Transaction 4405
Node_0 mined a new block:
Block Hash: 00006ef7d0cebe1dfbfaff6d653c4d7b9890e0584ae25b1877b76da9cd0871a1
Nonce: 92731
Transactions: ['Reward: 10 coins to Node_2', 'Transaction 8080', 'Transaction 4405']
Round 5
New transaction: Transaction 1925
New transaction: Transaction 5443
Node_0 mined a new block:
Block Hash: 00000cae0252e00cf12b7b57c29db4802bded7b48c3cb42d031c20767b16aaa4
Nonce: 20845
Transactions: ['Reward: 10 coins to Node_0', 'Transaction 1925', 'Transaction 5443']
Round 6
New transaction: Transaction 9423
New transaction: Transaction 5530
Node_2 mined a new block:
Block Hash: 00003514bb49d205b76b763b18678d87511f68c3ab867968629e90d5e9975358
Nonce: 25909
Transactions: ['Reward: 10 coins to Node_0', 'Transaction 9423', 'Transaction 5530']
Final Blockchain:
Block 0 - Hash: e79ac5bd45038c33fdfaa82b132f8dc542d0e999e102667ee519e920115fe2fb
Block 1 - Hash: 00000bcf3e51ff1326bb91782f288b74c52143d3ed1067db04f5197935ee11dd
Block 2 - Hash: 0000d13ca3afd023218e5f91b3d671d7446270584bccbdfba5db5f58385506a4
Block 3 - Hash: 0000e6f7e135a9505f2d62e3bdf3454329e4873fd385534b15ae10df16816040
Block 4 - Hash: 00006ef7d0cebe1dfbfaff6d653c4d7b9890e0584ae25b1877b76da9cd0871a1
Block 5 - Hash: 00000cae0252e00cf12b7b57c29db4802bded7b48c3cb42d031c20767b16aaa4
Block 6 - Hash: 00003514bb49d205b76b763b18678d87511f68c3ab867968629e90d5e9975358
PS C:\CN>