**Batch: A2 Roll No.: 16010121045**

**Experiment No. 1**

**Title:** Implementation of PoS and POW in Blockchain

**Objective:** Implementation of PoS and POW in Blockchain

# Expected Outcome of Experiment:

|  |  |
| --- | --- |
| **CO** | **Outcome** |
|  |  |

**Books/ Journals/ Websites referred:**

# Implementation Details:

*import* hashlib

*# initializing string*

string = "pargat"

nonce = 10000

*while*(True):

new\_string = string + str(nonce)

result = hashlib.sha256(new\_string.encode("utf-8"))

print(result.hexdigest())

nonce += 1

*if*(result.hexdigest()[0:3] == "0000"):

*break*

print(result.hexdigest())

print(nonce)

# PoW:



# PoS:

*import* random

arr = ['P0', 'P1', 'P2', 'P3', 'P4']

def randSelection(arr):

print(arr[random.randint(0, 4)])

def voteSelection(arr):

voteArr = [0, 0, 0, 0, 0]

*for* i *in* range(len(arr)):

*while* True:

newVote = int(input("Who does " + arr[i] + " vote for (just enter their number): "))

*if* 0 <= newVote <= 4:

*break*

print("Enter a valid input again!\n")

voteArr[newVote] += 1

print(arr[voteArr.index(max(voteArr))])

def coinAgeSelection(arr):

ageArr = [0, 0, 0, 0, 0]

*for* i *in* range(len(arr)):

*while* True:

newAge = int(input("What is the coin age of " + arr[i] + ": "))

*if* newAge >= 0:

*break*

print("Enter a valid input again!\n")

ageArr[i] = newAge

*for* t *in* range(5):

print(arr[ageArr.index(max(ageArr))])

*for* j *in* range(len(ageArr)):

print(ageArr[j], "\n")

ageArr[j] = ageArr[j] + 1

ageArr[ageArr.index(max(ageArr))] = 0

randSelection(arr)

voteSelection(arr)

coinAgeSelection(arr)

A screenshot of a computer program

Description automatically generated

**Conclusion:-**

## In this experiment, we learnt about PoS (Proof of Stake) and PoW (Proof of Work) in respect to blockchain technology, and how each of them is useful in their own way.