Aayush Shah
19BCE245
6 September 2022

BlockChain Technology

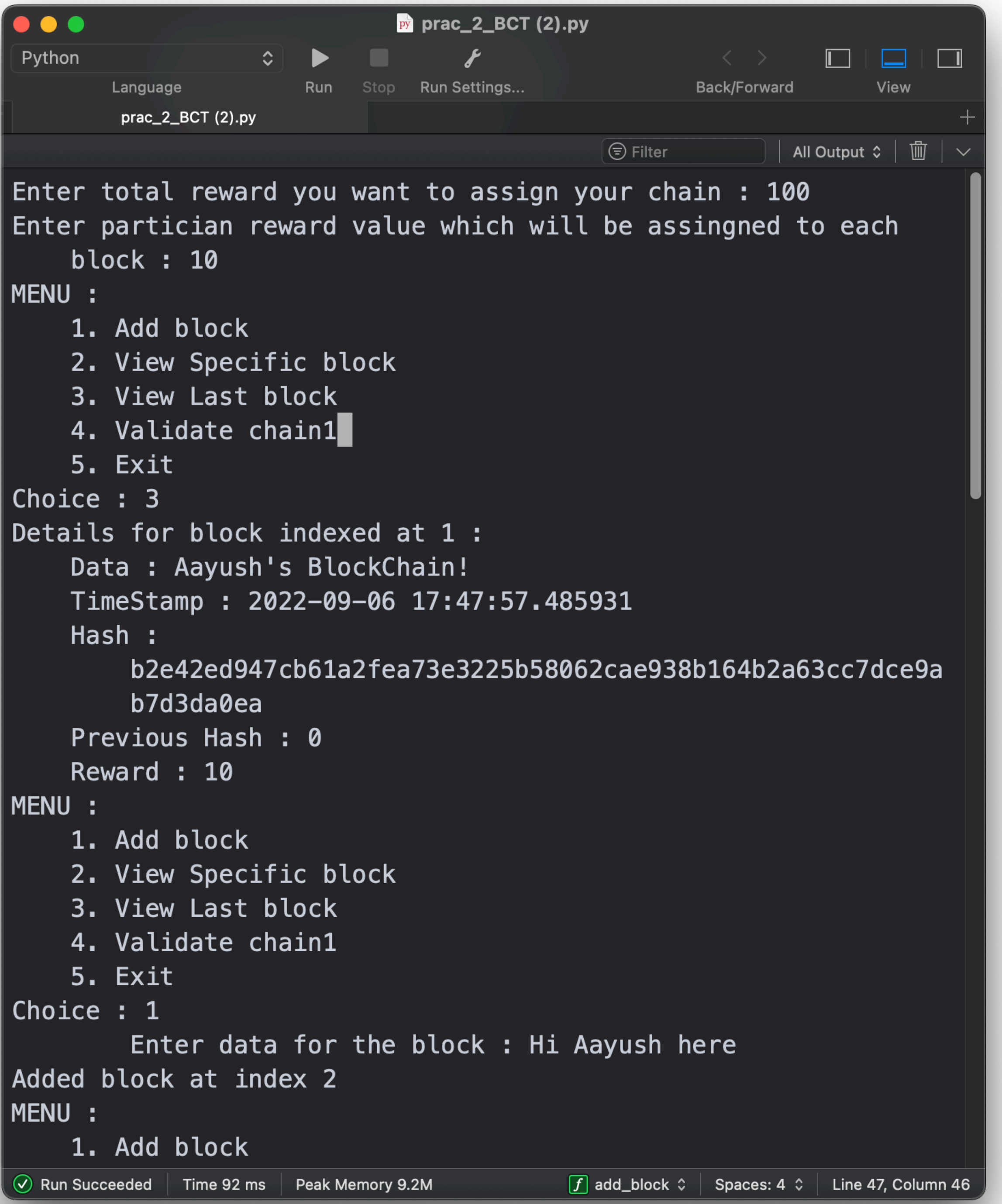
Practical 2

To create a blockchain and implement replay attacks on blockchain

- **Code :**

```
# 19BCE245 - Aayush Shah
# To create a blockchain and implement replay attacks on
blockchain

import datetime
import hashlib
import json
# import JSON
# from flask import jsonify

def compute_hash(index, previous_hash, timestamp, data):
    return hashlib.sha256((str(index) +
str(previous_hash) + str(timestamp) +
json.dumps(data)).encode('utf-8')).hexdigest()

class Block:
    def __init__(self, index, data, previous_hash,
reward):
        self.index = index
        self.data = data
        self.previous_hash = previous_hash
        self.timestamp = str(datetime.datetime.now())
        self.hash = compute_hash(self.index,
self.previous_hash, self.timestamp, self.data)
        self.reward = reward

    def print_block_details(self):
        print(f'Details for block indexed at {self.index}
: ')
        print(f'\tData : {self.data}')
        print(f'\tTimeStamp : {self.timestamp}')
        print(f'\tHash : {self.hash}')
        print(f'\tPrevious Hash : {self.previous_hash}')
        print(f'\tReward : {self.reward}')

class BlockChain:
    # chain = []
    def __init__(self, total_reward, partician):
        self.chain = []
        self.partician = partician
        self.total_reward = total_reward - partician
        genesis_block = Block(len(self.chain)
+1,'Aayush\'s BlockChain!',0, self.partician)
        self.chain.append(genesis_block)

    def add_block(self, data):
        assigned_reward = 0
        if self.total_reward-self.partician>0:
            self.total_reward -= self.partician
            assigned_reward = self.partician
        elif self.total_reward>0:
            assigned_reward = self.total_reward
            self.total_reward = 0
        new_block = Block(len(self.chain)+1, data,
self.chain[-1].hash, assigned_reward)
        self.chain.append(new_block)

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

    def get_specific_block(self,index):
        return self.chain[index]

    def print_block(self, block):
        block.print_block_details()

    def chain_validation(self):
        if self.chain[0].hash !=
compute_hash(self.chain[0].index,
self.chain[0].previous_hash, self.chain[0].timestamp,
self.chain[0].data):
            return False
        for i in range(1,len(self.chain)):
            if self.chain[i].previous_hash !=
self.chain[i-1].hash:
                return False
            if self.chain[i].hash !=
compute_hash(self.chain[i].index,
self.chain[i].previous_hash, self.chain[i].timestamp,
self.chain[i].data):
                return False
            if i != len(self.chain)-1 and
self.chain[i].hash != self.chain[i+1].previous_hash:
                return False
        return True

if __name__ == "__main__":
    total_reward = int(input('Enter total reward you want
to assign your chain : '))
    partician = total_reward
    while partician>=total_reward:
        partician = int(input('Enter partician reward
value which will be assingned to each block : '))
        if partician>=total_reward:
            print('Partician value should be less then
reward value.')
    myBlockChain = BlockChain(total_reward, partician)

    while True:
        print("""MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit""")
        choice = int(input("Choice : "))
        # try:
        if choice==1:
            data = input('\t\tEnter data for the block :
')
            myBlockChain.add_block(data)
            print(f'Added block at index 
{len(myBlockChain.chain)}')
        elif choice==2:
            index = int(input('\t\tEnter block index : 
'))
            try:

myBlockChain.print_block(myBlockChain.get_specific_block(
index-1))
            except:
                print('# Invalid index entered!')
        elif choice==3:

myBlockChain.print_block(myBlockChain.get_previous_block(
))
        elif choice==4:
            if myBlockChain.chain_validation():
                print(f'\tChain is validated.')
            else:
                print(f'\tChain is not validated')
        elif choice==5:
            print('Thank you!')
            break
        else:
            print('# Invalid choice!')
        # except:
        #     print('# Integer value expected!')
```

• **Output :**



```
Enter total reward you want to assign your chain : 100
Enter partician reward value which will be assingned to each
    block : 10
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 3
Details for block indexed at 1 :
    Data : Aayush's BlockChain!
    TimeStamp : 2022-09-06 17:47:57.485931
    Hash :
        b2e42ed947cb61a2fea73e3225b58062cae938b164b2a63cc7dce9a
        b7d3da0ea
    Previous Hash : 0
    Reward : 10
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 1
        Enter data for the block : Hi Aayush here
Added block at index 2
MENU :
    1. Add block
```

• **Full output text :**

```
Enter total reward you want to assign your chain : 100
Enter partician reward value which will be assingned to
each block : 10
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 3
Details for block indexed at 1 :
     Data : Aayush's BlockChain!
     TimeStamp : 2022-09-06 17:47:57.485931
     Hash :
b2e42ed947cb61a2fea73e3225b58062cae938b164b2a63cc7dce9ab7
d3da0ea
     Previous Hash : 0
     Reward : 10
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 1
          Enter data for the block : Hi Aayush here
Added block at index 2
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 1
          Enter data for the block : Ok bye see you again!
Added block at index 3
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 3
Details for block indexed at 3 :
     Data : Ok bye see you again!
     TimeStamp : 2022-09-06 17:48:21.228265
     Hash :
f94cf9fb4a3fc75f7b5ca5a65dc66ddd8d243028cff3a504cbe1e541c
d522a8d
     Previous Hash :
f49b694230b0f80ac8725632ed5a61bd146e8d8f7c9d3b68a8b4035fb
1bd5c2c
     Reward : 10
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 2
         Enter block index : 2
Details for block indexed at 2 :
     Data : Hi Aayush here
     TimeStamp : 2022-09-06 17:48:09.731279
     Hash :
f49b694230b0f80ac8725632ed5a61bd146e8d8f7c9d3b68a8b4035fb
1bd5c2c
     Previous Hash :
b2e42ed947cb61a2fea73e3225b58062cae938b164b2a63cc7dce9ab7
d3da0ea
     Reward : 10
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 4
     Chain is validated.
MENU :
    1. Add block
    2. View Specific block
    3. View Last block
    4. Validate chain1
    5. Exit
Choice : 5
Thank you!
```

• Learning Outcomes :

From this practical, I learned about how to create a blockchain from scratch.

I have implemented a user-oriented menu driven function through which user can add blocks in the blockchain as well as view any specific blocks. Validation can also be checked for the blockchain in which the respective hash values will be checked for individual blocks.

Basically, every block contains the index, data, it's own hash value along with previous block's hash value and time stamps. Also each block has assigned reward tokens which will be asked to the user in the starting of the program. Total number of tokens is also asked from which the partician amount will be deducted as new block is created and that amount is assigned to that specific block.

This blockchain is usually linked-list type structure in which every block is connected through chain with each other. Moreover, Reward distribution and block chain validation functionalities are implemented here.