Código da Blockchain

Código completo e comentado:

https://github.com/leandromarques1/Blockchain_DisciplinaCloud_2021.1/blob/main/myBlockchain.py

Foi criado um Código de Blockchain, escrito em **Python,** para exemplificar o funcionamento de uma Blockchain aplicada a um sistema de Rastreabilidade:

A cada bloco adicionado na blockchain deverá conter os seguintes dados:

- **Origem do Produto:** de onde o produto está saindo
- **Destino do Produto:** para onde o produto está indo
- **Código do Produto:** código de identificação do produto
- **Empresa:** empresa que responsável pela etapa da logística, representada por aquele Bloco
- **Timestamp:** momento da linha temporal em que foi criado bloco
- **Hash:** código criptografado que contem as informações de todo o bloco
- **previousHash:** hash do bloco anterior; é importante na construção do bloco a ser adicionado

Origem
Destino
Código
Empresa
Timestamp
previousHash
Hash

Condições a serem obedecidas para Validação a ser efetuada pela cadeia:

- **Origem** de um bloco deve ser igual ao **Destino** do bloco anterior
- **Código** de um bloco deve ser igual ao código do bloco anterior

```python
from hashlib import sha256
import json
from datetime import datetime

block_chain = []    #Criar a estrutura de dados para a Blockchain
                    (inicialmente vazia)

def get_time():
  return datetime.utcnow().timestamp()

#função para gerar o Timestamp (e assim ver
a hora que transação foi efetivada)

def isValidHashDifficulty(hash, difficulty):
  count = 0
  for i in hash:
    count += 1
    if(i != '0'):
      break
  return count > difficulty

#   DIFFICULTY: forma de validar o hash.
#   No nosso caso, um hash será válido se
começar com quatro "0" (por exemplo, "0000abc…")
#   Quanto maior essa dificuldade, mais tempo
vai levar para encontrar um hash válido.
```

```
#NONCE: simples número inteiro que
vamos incrementar sempre que um hash
não for válido.
# em resumo, o Nonce é um iterador:
contador p/ fazer iterações até achar
um HASH válido
```

```
#função para gerar o Hash
def generate_hash(block):
    nonce = 0
    block["nonce"] = nonce
    hash = sha256(json.dumps(block).encode('utf-8')).hexdigest()
    while(not isValidHashDifficulty(hash, 4)):
        nonce = nonce + 1
        block["nonce"] = nonce
        hash = sha256(json.dumps(block).encode('utf-8')).hexdigest()
    return hash
```

```
#função p/ adicionar blocos na Blockchain
def add_block(block):
  if(len(block_chain) == 0): #se block_chain estiver vazia (1º bloco a ser adicionado)
    block["timestamp"] = get_time()
    block["hash"] = generate_hash(block)
  else:
    block["timestamp"] = get_time()
    last_block = block_chain[-1]
    while block["origem"] != last_block["destino"]:
      origem = input("Origem errada! Por favor, digite novamente: ")
      if origem == last_block["destino"]:
        block['origem'] = origem

    while block["codigo"] != last_block["codigo"]:
      codigo = input("Código do Produto errado! Digitar novamente: ")
      if codigo == last_block["codigo"]:
        block['codigo'] = codigo

    block["previous_Hash"] = last_block["hash"]
    block["hash"] = generate_hash(block)

  block_chain.append(block)  #Após todas as validações serem feitas, adicionar bloco na Blockchain
```

#validação dos blocos anteriores
#validação de ORIGEM e DESTINO

#validação de Código do Produto

```python
# Esperando um Evento
# Evento --> informação de adicionar ou não o bloco na Blockchain
while 1:
    print("\n#########################################")
    resp = input("Deseja adicionar alguma informação?(S/N) ")
    if resp == 's' or resp == 'S':
        data = {}   #bloco vazio, p/ colocar informações

        origem = input("Origem do Produto: ")
        destino = input("Destino do Produto: ")
        codigo = input("Codigo do Produto: ")
        empresa = input("Empresa responsável: ")

        data['origem'] = origem
        data['destino'] = destino
        data['codigo'] = codigo
        data['empresa'] = empresa

        json_data = json.dumps(data)
        add_block(data)
        print("=== Cadastro realizado com sucesso! ===")

    else:
        break
```