#### Atividade blockchain

#### **Ferramentas**

- Python 3.8.11;
- Biblioteca de criptografia do Python (time, json, os, hashlib sha256);

### **Objetivos**

- Criar uma aplicação de blockchain local
- Implementar um algoritmo de prova por trabalho local onde a dificuldade é definida pelo usuário da aplicação
- Armazenar os blocos validados
- Implementar uma rotina que verifica a integridade dos blocos.
- O conteúdo permitido nos blocos deverá ser definido por você.

### **Procedimentos**

Foi desenvolvido um código de uma aplicação em Python e os procedimentos de funcionamento do programa são os seguintes:

- É necessário criar no mesmo diretório do programa uma pasta com o nome de 'blockchain' para armazenar os blocos
- Ao iniciar o programa primeiramente é mandatório selecionar a opção de número um para a criação do bloco de origem.
- Em seguida na opção de número dois é possível adicionar a quantidade de blocos que desejar, lembrando sempre de inserir: remetente, destinatário, transação e dificuldade de criação do bloco
- A terceira opção do menu executa uma verificação de integridade da blockchain, averiguando se não teve adulteração de informações gravadas nos blocos.

# Metodologia e Resultados

O programa foi desenvolvido com o uso de um menu de seleção com chamada de funções. A seguir o fluxo das funções seguido de uma breve descrição:

• Ao rodar o programa é exibido um menu, criado pela função 'menu'

- Ao selecionar a opção um chama a função 'cria\_prim\_bloco' que escreve em um arquivo json as informações já pré determinadas do bloco de origem.
- Ao selecionar a opção dois chama a função 'cria\_bloco' que passa como parâmetro as variáveis: remetente, destinatário, transação e dificuldade.
   Sendo a dificuldade a quantidade de zero que tem que ter no hash
  - Primeiramente essa função acessa a pasta que guarda os blocos e extrai pela leitura do nome dos arquivos a quantidade de blocos existentes
  - starta a contagem de tempo para a verificação de desempenho
  - chama uma terceira função de nome 'minera' que passa como parâmetro o numero do ultimo bloco e a dificuldade
    - A função 'minera' abre o último bloco já criado no modo de leitura de binários e extrai para variável 'conteudo\_bloco' todas as informações gravadas no bloco
    - em seguida ao entrar em um loop while concatena em uma variável temporária o conteúdo do bloco mais o valor do nonce
    - faz o hash256 dessa variável 'temp' no modo hexadecimal
    - faz a verificação por condicional se o início do bloco inicia com a quantidade de zeros determinadas pela variável 'dificuldade' informada pelo usuário, caso essa condição tenha sido atendida a função retorna o hash mais o valor do nonce, caso contrário sai do condicional soma um na variável 'nonce' e permanece no laço while
- sdasd

A figura 1 mostra a criação do primeiro bloco, de modo que precisa-se de um nó origem para que os sucessores possam ter uma ligação pelo hash do anterior:

```
blockchainlk > blockchain
                              Ö
                                                        Python 3.9.6 (tags/v3.9.6:dl
                                                        D64)] on win32
Nome
                                  Data de modificação
                                                        Type "help", "copyright", "(
 1
                                   19/08/2021 12:39
                                                        ====== RESTART: C:\User:
                                                        Digite 1 criar blockchain
                                                        Digite 2 adiciona bloco
1 - Bloco de Notas
                                            Х
                                                        Digite 3 checa integridade
Arquivo Editar Formatar Exibir Ajuda
                                                        Digite 0 sair
                                                        Digite a opção: 1
    "remetente": "primeiro bloco",
                                                        Primeiro bloco criado
    "destinatario": "primeiro bloco",
    "transacao": "zero",
    "ant bloco": {
                                                        Digite 1 criar blockchain
        Digite 2 adiciona bloco
        "nonce": "0",
                                                        Digite 3 checa integridade
        "nome arquivo": "origem"
                                                        Digite 0 sair
                                                        Digite a opção:
```

Figura 1

A figura 2 mostra mostra a criação do segundo bloco já com o hash e o nonce do bloco anterior, observa-se que nesse caso ao alterar qualquer informação do bloco origem o hash guardado no bloco dois já não será mais válido e comprovará a alteração a blockchain:

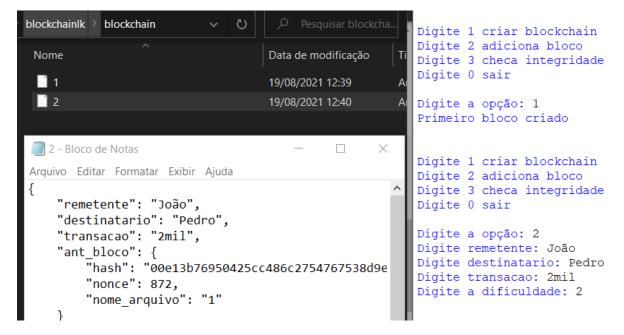


Figura 2

A figura 3 mostra a função de verificação de que os blocos não foram alterados:

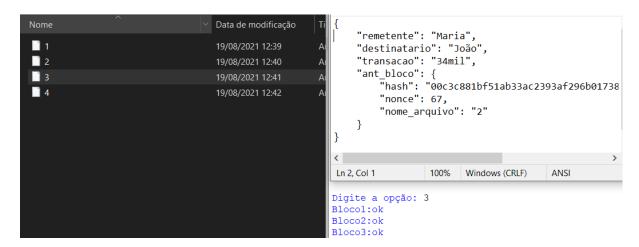


Figura 3

A figura 4 mostra que foi possível identificar uma alteração após alterar o valor da transação no bloco 3:

```
Data de modificação
                                                                 "remetente": "Maria",
19/08/2021 12:39
                                                                 "destinatario": "João",
"transacao": "4mil",
                                     19/08/2021 12:40
                                                                 "ant_bloco": {
    "hash": "00c3c881bf51ab33ac2393af296b01738
1 4
                                     19/08/2021 12:42
                                                                      "nonce": 67,
                                                                      "nome arquivo": "2"
                                                            }
                                                             Ln 4, Col 19
                                                                               100%
                                                                                       Windows (CRLF)
                                                                                                         ANSI
                                                            Digite a opção: 3
                                                            Bloco1:ok
                                                            Bloco2:ok
                                                            Bloco3:mudou
```

Figura 4

A figura 5 mostra a criação do hash com dificuldade três, ou seja, três no início:

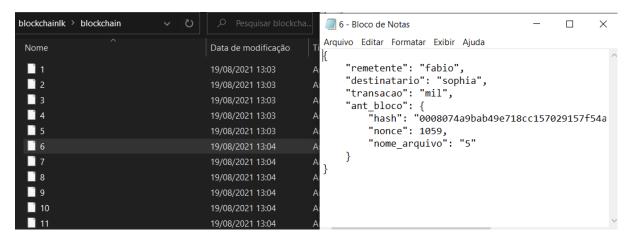


Figura 5

A figura 6 mostra a criação do hash com dificuldade quatro, ou seja, quatro no início:

```
Arquivo Editar Formatar Exibir Ajuda
                                     Data de modificação
                                                                  "remetente": "fabio",
19/08/2021 13:07
                                                                  "destinatario": "sophia",
19/08/2021 13:07
                                                                  "transacao": "mil",
                                      19/08/2021 13:07
                                                                  "ant bloco": {
                                      19/08/2021 13:07
                                                                       "hash": "0000921ebe232e2a55431e2d542b33dcc
                                                                      "nonce": 54924,
                                      19/08/2021 13:07
                                                                      "nome_arquivo": "5"
                                      19/08/2021 13:07
                                                                 }
                                      19/08/2021 13:07
8
                                      19/08/2021 13:08
                                      19/08/2021 13:08
10
                                      19/08/2021 13:08
```

Figura 6

A figura 7 mostra a criação do hash com dificuldade cinco, ou seja, cinco no início:

```
Arquivo Editar Formatar Exibir Ajuda
Nome
                                    Data de modificação
"remetente": "fabio",
                                    19/08/2021 13:11
                                                              "destinatario": "sophia",
19/08/2021 13:11
                                                              "transacao": "mil",
3
                                    19/08/2021 13:11
                                                              "ant_bloco": {
1 4
                                    19/08/2021 13:11
                                                                   "nonce": 1082016,
"nome_arquivo": "5"
1 5
                                    19/08/2021 13:11
                                                              }
                                    19/08/2021 13:12
                                    19/08/2021 13:12
                                    19/08/2021 13:12
10
                                    19/08/2021 13:12
                                    19/08/2021 13:12
```

Figura 7

A figura 8 mostra um gráfico comparativo do tempo em segundos necessários para mineração de blocos de acordo com as dificuldades de três, quatro e cinco. No gráfico também está demonstrado o tempo médio de cada dificuldade para uma amostragem de oito blocos:

# Análise do impacto da dificuldade

MÉDIA DIF3 =0,02593 DIF4 =0,30819 DIF5 = 4,41304

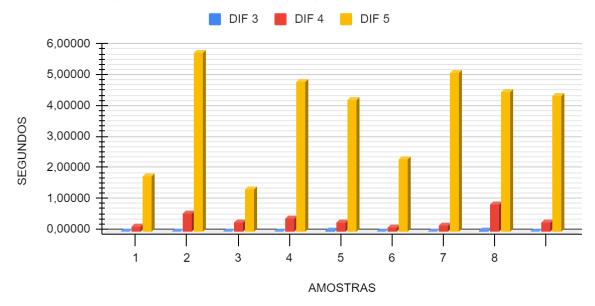


Figura 8

Observa-se um crescimento logaritmos do tempo de acordo com o acréscimo gradual da dificuldade, o que é uma característica do sistema já que tem ligação direta com a dificuldade para alteração dos blocos.

### Conclusão

E como melhoria sugere adicionar eventos de verificação try/catch e mensagens de erros como as de inserção de valores e tipos inválidos; Para evitar repetição de código sugere criar uma função que calcula o hash para ser usada na função 'minera' e na 'check\_integridade';

# **Bibliografia**

Making a menu in Python <a href="https://www.youtube.com/watch?v=63nw00JqHo0">https://www.youtube.com/watch?v=63nw00JqHo0</a>

Como Minerar Bitcoin com Python (Código em Python para Minerar Bitcoin) <a href="https://www.youtube.com/watch?v=m3k4kvX6izo">https://www.youtube.com/watch?v=m3k4kvX6izo</a>>

Blockchain with Python #2: Creating blocks & checking integrity | Python projects <a href="https://www.youtube.com/watch?v=sYwPEenxMN0">https://www.youtube.com/watch?v=sYwPEenxMN0</a>