

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN



BLOCKCHAIN VÀ ỨNG DỤNG

LAB 01

GVHD: TS. Nguyễn Đình Thúc

Nhóm sinh viên thực hiện

19120659 – Phạm Văn Thành

20120356 – Lê Minh Quân

20120382 – Hoàng Thu Thủy

20120386 – Lê Phước Toàn

20120389 – Nguyễn Thị Bích Trâm

Thành phố Hồ Chí Minh, tháng 12-2023

Thông tin nhóm

Nhóm 3			
Họ và tên	MSSV	Phân công	Đánh giá
Phạm Văn Thành	19120659	Xây dựng command line client	100%
Lê Minh Quân	20120356	Xây dựng tính năng xác thực giao dịch của cây Merkle	100%
Hoàng Thu Thủy	20120382	Xây dựng cấu trúc block và blockchain	100%
Lê Phước Toàn	20120386	Xây dựng cấu trúc cây Merkle	100%
Nguyễn Thị Bích Trâm	20120389	Viết báo cáo	100%

MỤC LỤC

I.	Cấu trúc chương trình	4
1.	Xây dựng blockchain	4
2.	Xây dựng cây Merkle	4
II.	Cấu trúc dữ liệu	6
1.	Cấu trúc dữ liệu Blockchain ban đầu	6
2.	Cấu trúc dữ liệu cây Merkle	6
III.	Hướng dẫn sử dụng	7
1.	Help command	7
2.	Add block command	7
a)	<i>Thêm một blockchain mới với nội dung “Brand new block”</i>	<i>8</i>
b)	<i>Thêm 3 blocks với số lượng transaction mỗi block là 3</i>	<i>8</i>
3.	Read block command	8
a)	<i>Hiển thị thông tin tất cả các block trong blockchain:</i>	<i>9</i>
b)	<i>Hiển thị thông tin của block 4</i>	<i>9</i>
4.	Verify block command	9
a)	<i>Thực hiện verify với nội dung ”Brand new block” ở tất cả block trong blockchain</i>	<i>10</i>
b)	<i>Thực hiện verify với một block cụ thể được chỉ định</i>	<i>10</i>
IV.	Tham khảo	11

I. Cấu trúc chương trình

Chương trình được chia ra thành các module sau:

- Block: cài đặt các thao tác của 1 block.
- Blockchain: các thao tác liên quan đến blockchain.
- Merkle tree: các thao tác xây dựng Merkle tree và xác thực giao dịch.

1. Xây dựng blockchain

Tạo ra một block đầu tiên (genesis block): Sử dụng hàm *NewGenesisBlock()* để tạo ra một block đầu tiên. Block này sẽ chứa một giao dịch với dữ liệu là “Genesis Block”. Hàm *NewBlock(transactions []*Transaction, prevBlockHash []byte)* được sử dụng để tạo ra block này, với tham số đầu vào là danh sách các giao dịch và hash của block trước đó.

Tính toán và thiết lập giá trị hash cho block: Sử dụng hàm *SetHash()* để tính toán và thiết lập giá trị hash cho block. Giá trị hash này sẽ dựa trên timestamp, hash của tất cả các giao dịch trong block và hash của block trước đó. Hàm *HashTransactions()* được sử dụng để tính toán giá trị hash cho tất cả các giao dịch trong block.

Tạo ra một blockchain mới: Sử dụng hàm *NewBlockchain()* để tạo ra một blockchain mới. Blockchain này sẽ chứa block genesis vừa tạo.

Thêm block vào blockchain: Sử dụng hàm *AddBlock(transactions []*Transaction)* để thêm một block mới vào blockchain. Block mới này sẽ chứa danh sách các giao dịch được truyền vào và hash của block trước đó.

Tính toán giá trị hash cho tất cả các giao dịch trong block: Sử dụng hàm *HashTransactions()* để tính toán giá trị hash cho tất cả các giao dịch trong block.

2. Xây dựng cây Merkle

Cây Merkle là một cấu trúc dữ liệu cây dùng để tóm tắt và xác thực tính toàn vẹn của các tập dữ liệu lớn một cách an toàn và hiệu quả.

Trong ngữ cảnh của Blockchain, cây Merkle có vai trò tổng hợp và xác thực các giao dịch có trong một block, đảm bảo độ tin cậy và tính toàn vẹn dữ liệu của Blockchain.

Quá trình xác thực giao dịch:

- Được thực hiện bởi client và một full node.
- Client sẽ gửi truy vấn Merkle path đến full node
- Khi có Merkle path thì thực hiện tái tạo lại Merkle root hash và block hash nhằm xác thực.

Quá trình tạo ra Merkle path được thực hiện như sau:

1. Lặp qua danh sách các leaf nodes ở trong cây Merkle để tìm node có hash trùng với hash của giao dịch cần xác thực, nếu không tìm thấy thì trả về mảng rỗng.
2. Nếu tìm thấy thì tiếp tục lặp qua danh sách các non-leaf nodes để tìm ra node cha.
 - Nếu là node con bên trái, thêm node con bên phải vào Merkle path. Đánh dấu node hiện tại là node con bên trái.
 - Nếu là node con bên phải, thêm node con bên trái vào Merkle path. Đánh dấu node hiện tại là node con bên phải.
3. Gán node hiện tại bằng node cha và lặp lại bước 2 đến khi nào node hiện tại là node gốc.

Điểm yếu về hiệu năng: do lặp qua tất cả các non-leaf node mỗi khi tìm được node cha nên thuật toán không tối ưu.

Sau khi có Merkle path thì lặp qua các phần tử của nó, nếu là node con bên trái thì nối phần tử của Merkle path vào bên phải và ngược lại.

Khi lặp hết các phần tử của Merkle path thì có được Merkle root hash. Kết hợp với các trường header đã có của block, ta tạo ra block hash và đem đi so sánh với hash hiện tại của block.

Nếu bằng nhau thì chứng tỏ giao dịch đó có tồn tại trong block. Trong trường hợp Merkle path là rỗng hoặc block hash tạo ra không bằng block hash đã có thì kết luận giao dịch không tồn tại trong block.

II. Cấu trúc dữ liệu

1. Cấu trúc dữ liệu Blockchain ban đầu

Transaction: Đây là cấu trúc dữ liệu chứa thông tin về một giao dịch. Mỗi giao dịch bao gồm dữ liệu được lưu trữ dưới dạng mảng byte.

Block: Mỗi block trong blockchain chứa thông tin về thời gian tạo block (Timestamp), danh sách các giao dịch (Transactions), hash của block trước đó (PrevBlockHash) và hash của chính nó (Hash).

Blockchain: Blockchain là một danh sách liên kết các block. Mỗi block trong danh sách này chứa thông tin về một số giao dịch và liên kết đến block trước đó thông qua hash. Một blockchain mới sẽ được tạo ra với block đầu tiên là block genesis.

2. Cấu trúc dữ liệu cây Merkle

Cấu trúc của một cây Merkle là một cây nhị phân gồm ba thành phần chính:

- Leaf Nodes: những nodes này lưu lại toàn bộ các giá trị Hash của tất cả giao dịch có trong một block và cũng là các nodes cơ sở để xây dựng nên cây Merkle.
- Non-Leaf Nodes: những nodes này lưu các giá trị Hash được tính toán dựa trên giá trị Hash của các nodes con của nó – Hash (Hash (child 1) | Hash (child 2)). Giá trị Hash thường được tính toán bằng thuật toán SHA-256.
- Root Node: node cao nhất của cây Merkle, giá trị Hash duy nhất được lưu bên trong nó gọi là Merkle Root, đại diện cho toàn bộ tập dữ liệu được lưu trữ bên trong cây Merkle.

Các nodes trong cây Merkle có cấu trúc như sau:

- Left Node: node con bên trái
- Right Node: node con bên phải
- Data: lưu dữ liệu đã được hash của giao dịch

III. Hướng dẫn sử dụng

Chương trình sử dụng packet “Flag” trong Golang để xử lý tham số dòng lệnh một cách tiện lợi. Các command và chức năng được sử dụng trong chương trình bao gồm:

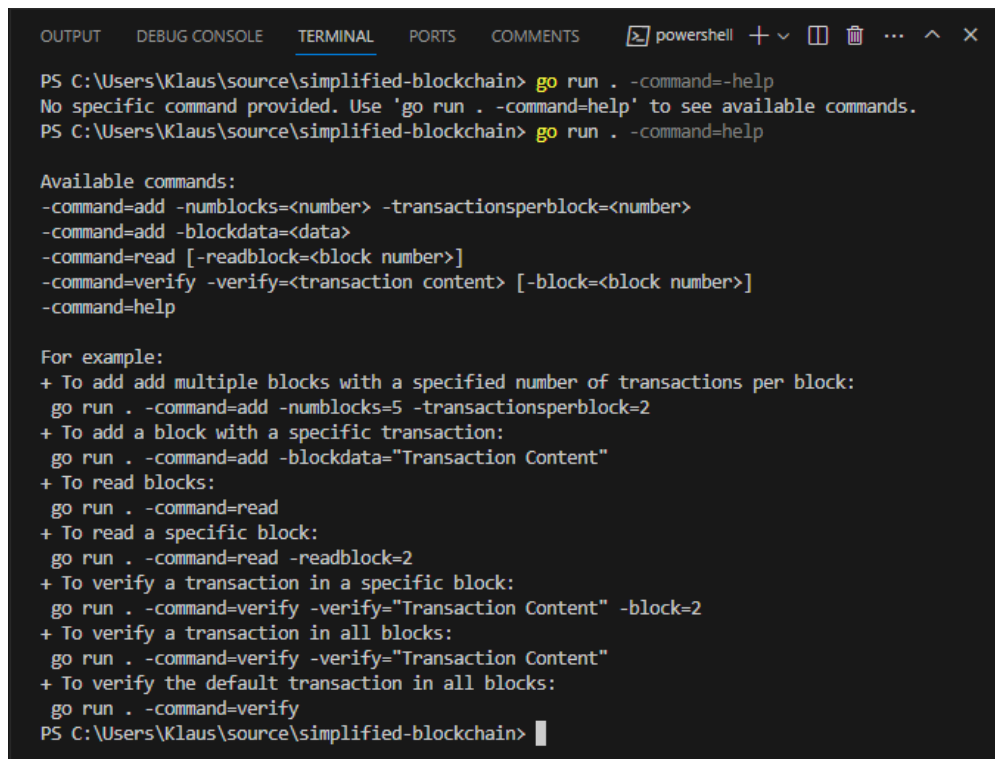
1. Help command

Command: ‘*help*’

Chức năng: Liệt kê danh sách các câu lệnh khả dụng kèm ví dụ cách dùng tương ứng

Sử dụng: ‘*go run . command=help*’

Kết quả :



```
OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS  powershell + - - - - - X
PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=-help
No specific command provided. Use 'go run . -command=help' to see available commands.
PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=help

Available commands:
-command=add -numblocks=<number> -transactionsperblock=<number>
-command=add -blockdata=<data>
-command=read [-readblock=<block number>]
-command=verify -verify=<transaction content> [-block=<block number>]
-command=help

For example:
+ To add add multiple blocks with a specified number of transactions per block:
go run . -command=add -numblocks=5 -transactionsperblock=2
+ To add a block with a specific transaction:
go run . -command=add -blockdata="Transaction Content"
+ To read blocks:
go run . -command=read
+ To read a specific block:
go run . -command=read -readblock=2
+ To verify a transaction in a specific block:
go run . -command=verify -verify="Transaction Content" -block=2
+ To verify a transaction in all blocks:
go run . -command=verify -verify="Transaction Content"
+ To verify the default transaction in all blocks:
go run . -command=verify
PS C:\Users\Klaus\source\simplified-blockchain> |
```

2. Add block command

Command: ‘*add*’

Các tham số đi kèm:

- **blockdata:** nội dung của transaction
- **numblock:** số lượng block muốn thêm vào blockchain
- **transactionsperblock:** số lượng transaction trong mỗi block

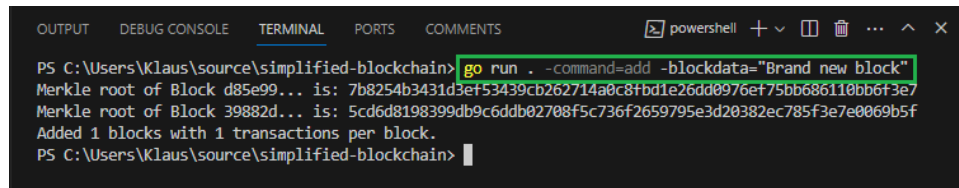
Chức năng: Thêm một block mới vào block chain với nội dung data người dùng nhập hoặc thêm số lượng block cùng với số lượng transaction theo thông tin người dùng nhập.

Sử dụng: `go run . -command=add -blockdata=<data>`

Hoặc `go run . -command=add -numblocks=<number> -transactionsperblock=<number>`

Kết quả :

a) *Thêm một blockchain mới với nội dung “Brand new block”*



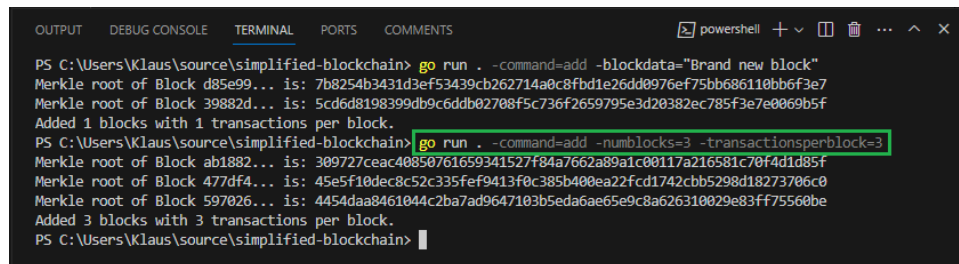
```

PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=add -blockdata="Brand new block"
Merkle root of Block d85e99... is: 7b8254b3431d3ef53439cb262714a0c8fbd1e26dd0976ef75bb686110bb6f3e7
Merkle root of Block 39882d... is: 5cd6d8198399db9c6ddb02708f5c736f2659795e3d20382ec785f3e7e0069b5f
Added 1 blocks with 1 transactions per block.
PS C:\Users\Klaus\source\simplified-blockchain>

```

=> Chương trình sẽ tự hiểu số lượng transaction = 1 và thêm vào blockchain

b) *Thêm 3 blocks với số lượng transaction mỗi block là 3*



```

PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=add -blockdata="Brand new block"
Merkle root of Block d85e99... is: 7b8254b3431d3ef53439cb262714a0c8fbd1e26dd0976ef75bb686110bb6f3e7
Merkle root of Block 39882d... is: 5cd6d8198399db9c6ddb02708f5c736f2659795e3d20382ec785f3e7e0069b5f
Added 1 blocks with 1 transactions per block.
PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=add -numblocks=3 -transactionsperblock=3
Merkle root of Block ab1882... is: 309727ceac40850761659341527f84a7662a89a1c00117a216581c70f4dd85f
Merkle root of Block 477df4... is: 45e5f10dec8c52c335fef9413f0c385b400ea22fcd1742cbb5298d18273706c0
Merkle root of Block 597026... is: 4454daa8461044c2ba7ad9647103b5eda6ae65e9c8a626310029e83ff75560be
Added 3 blocks with 3 transactions per block.
PS C:\Users\Klaus\source\simplified-blockchain>

```

=> Lúc này nội dung transaction sẽ được tự động đánh số theo thứ tự: Transaction 1, Transaction 2, Transaction 3, ...

3. Read block command

Command: `'read'`

Các tham số đi kèm:

- **readblock:** chỉ định một block cụ thể để đọc

Chức năng: Hiểm thị thông tin của một hay tất cả các block trong blockchain

Sử dụng: `go run . -command=read [-readblock=<block number>]`

Kết quả :

a) Hiển thị thông tin tất cả các block trong blockchain:

```

PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=read

reading Block: all

Blockchain:

Block 0
Prev. hash:
> Data: Genesis Block
Hash: d85e991bf2e37ca6374cb7b254a042a4e4cc4add4c59db2e470ace6d87cd96a

Block 1
Prev. hash: d85e991bf2e37ca6374cb7b254a042a4e4cc4add4c59db2e470ace6d87cd96a
> Data: Brand new block
Hash: 39882defed0cc8df7f1023a51af0895f8d9fa341a4da7c86bce158e8b0b65bde

Block 2
Prev. hash: 39882defed0cc8df7f1023a51af0895f8d9fa341a4da7c86bce158e8b0b65bde
> Data: Transaction 1 > Data: Transaction 2 > Data: Transaction 3
Hash: ab18825c2ef4634055a22d3069a7c606a6bbe7e8df9f156afaa7992f07a4b562

Block 3
Prev. hash: ab18825c2ef4634055a22d3069a7c606a6bbe7e8df9f156afaa7992f07a4b562
> Data: Transaction 4 > Data: Transaction 5 > Data: Transaction 6
Hash: 477df4fd441955f43d8cd70558e2c3d0a0e3cb75be4497e412ed295b2a916f7e

Block 4
Prev. hash: 477df4fd441955f43d8cd70558e2c3d0a0e3cb75be4497e412ed295b2a916f7e
> Data: Transaction 7 > Data: Transaction 8 > Data: Transaction 9
Hash: 5970268b9ba5e5dcb5a5a05a49f41e1b46e67c5411976108febce6ee4bc9055
PS C:\Users\Klaus\source\simplified-blockchain>
  
```

block mặc định

block thêm vào lần 1

Các block được thêm vào lần 2 với 3 transaction

b) Hiển thị thông tin của block 4

```

> Data: Transaction 7 > Data: Transaction 8 > Data: Transaction 9
Hash: 5970268b9ba5e5dcb5a5a05a49f41e1b46e67c5411976108febce6ee4bc9055
PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=read -readblock=4

reading Block: 4

Block 4
Prev. hash: 477df4fd441955f43d8cd70558e2c3d0a0e3cb75be4497e412ed295b2a916f7e
> Data: Transaction 7 > Data: Transaction 8 > Data: Transaction 9
Hash: 5970268b9ba5e5dcb5a5a05a49f41e1b46e67c5411976108febce6ee4bc9055
PS C:\Users\Klaus\source\simplified-blockchain>
  
```

4. Verify block command

Command: 'verify'

Các tham số đi kèm:

- **verify:** Nội dung transaction cần verify
- **block:** chỉ định một block cụ thể để thực hiện verify

Chức năng: Kiểm tra transaction có trong block hay không; kết quả trả về sẽ là **true** hoặc **false**

Sử dụng:

`go run . -command=verify -verify=<transaction content> [-block=<block number>]`

Kết quả :

a) Thực hiện verify với nội dung "Brand new block" ở tất cả block trong blockchain

```

OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  COMMENTS
powershell + - - - - -

Block 4
Prev. hash: 477df4fd441955f43d8cd70558e2c3d0a0e3cb75be4497e412ed295b2a916f7e
> Data: Transaction 7 > Data: Transaction 8 > Data: Transaction 9
Hash: 5970268b9ba5e5dcb5a5a05a49f41e1b46e67c5411976108febcec6ee4bc9055
PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=verify -verify="Brand new block"

Verifying transaction in Block 0
Check whether transaction with content 'Brand new block' is in block d85e991bf2e37ca6374cb7b254a042a4e4cc4add4c59
db2e470ace6d87cd96a
Verify result: false

Verifying transaction in Block 1
Check whether transaction with content 'Brand new block' is in block 39882defed0cc8df7f1023a51af0895f8d9fa341a4da7
c86bce158e8b0b65bde
Verify result: true

Verifying transaction in Block 2
Check whether transaction with content 'Brand new block' is in block ab18825c2ef4634055a22d3069a7c606a6bbe7e8df9f1
56afaa7992f07a4b562
Verify result: false

Verifying transaction in Block 3
Check whether transaction with content 'Brand new block' is in block 477df4fd441955f43d8cd70558e2c3d0a0e3cb75be449
7e412ed295b2a916f7e
Verify result: false

Verifying transaction in Block 4
Check whether transaction with content 'Brand new block' is in block 5970268b9ba5e5dcb5a5a05a49f41e1b46e67c5411976
108febcec6ee4bc9055
Verify result: false
PS C:\Users\Klaus\source\simplified-blockchain>

```

b) Thực hiện verify với một block cụ thể được chỉ định

```

OUT Focus folder in explorer (ctrl + click)  PORTS  COMMENTS
powershell + - - - - -

PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=verify -verify="Brand new block" -block=4

Verifying transaction in Block 4
Check whether transaction with content 'Brand new block' is in block 5970268b9ba5e5dcb5a5a05a49f41e1b46e67c5411976
108febcec6ee4bc9055
Verify result: false

PS C:\Users\Klaus\source\simplified-blockchain> go run . -command=verify -verify="Brand new block" -block=1

Verifying transaction in Block 1
Check whether transaction with content 'Brand new block' is in block 39882defed0cc8df7f1023a51af0895f8d9fa341a4da7
c86bce158e8b0b65bde
Verify result: true
PS C:\Users\Klaus\source\simplified-blockchain>

```

IV. Tham khảo

- [Going the distance \(jeiwan.net\)](http://jeiwan.net)
- [Data Validation — Symbol Documentation](#)
- [cbergoon/merkletree: A Merkle Tree implementation written in Go. \(github.com\)](https://github.com/cbergoon/merkletree)
- <https://pkg.go.dev/flag>
- <https://www.tutorialspoint.com/golang-program-to-implement-a-merkle-tree>
- <https://steemit.com/utopian-io/@tensor/building-a-blockchain-with-go---part-8---the-merkle-tree>
- <https://www.simplilearn.com/tutorials/blockchain-tutorial/merkle-tree-in-blockchain>