Work on project. Stage 1/6: Blockchain essentials

145 users solved this problem. Latest completion was **about 17 hours ago**.

Project: Blockchain

Hard 9 minutes

§1. Description

70 / 70 Prerequisites

- ✓ Introduction to OOP — Stage 1 — 10
- ✓ Units of information — Stage 1 — 12
- ✓ Computer algorithms — Stage 1 — 7
- ✓ The big O notation — Stage 1 — 7
- ✓ Data structures — Stage 1 — 4

Show all

Blockchain has a simple interpretation: it's just a chain of blocks. It represents a sequence of data that you can't break in the middle; you can only append new data at the end of it. All the blocks in the blockchain are chained together.

Check out this great video about the blockchain. It uses a different approach to reach the final result of the project, which is cryptocurrencies, but it explains the blockchain pretty well.

To be called a blockchain, every block must include the **hash of the previous block**. Other fields of the block are optional and can store various information.

Note that if you change one block in the middle, the hash of this block will also change. and the next block in the chain would no longer contain the hash of the previous block. Therefore, it's easy to check that the chain is invalid.

In the first stage, you need to implement such a blockchain. In addition to storing the hash of the previous block, every block should also have a unique identifier. The chain starts with a block whose id = 1. Also, every block should contain a timestamp representing the time the block was created. You can use the following code to get such a timestamp. This represents the number of milliseconds since 1 January 1970.

```
long timeStamp = new Date().getTime(); // 1539795682545 represents 17.10.2018, 20:01:22.545
```

By the way, since the first block doesn't have a previous one, its hash of the previous block should be 0.

The class Blockchain should have at least two methods: the first one generates a new block in the blockchain and the second one validates the blockchain and returns true if the blockchain is valid. Of course, the Blockchain should store all it's generated blocks. The validation function should validate all the blocks of this blockchain.

Also, for hashing blocks, you need to choose a good cryptographic hash function which is impossible to reverse-engineer. Insecure hash functions allow hackers to change the information of the block so that that the hash of the block stays the same, so the hash function must be secure. A good example of a secure hash function is SHA-256. You can use this implementation of the SHA-256 hashing:

/

```
import java.security.MessageDigest;

class StringUtil {
    /* Applies Sha256 to a string and returns a hash. */
    public static String applySha256(String input){
        try {
            MessageDigest digest = MessageDigest.getInstance("SHA-256");
            /* Applies sha256 to our input */
            byte[] hash = digest.digest(input.getBytes("UTF-8"));
            StringBuilder hexString = new StringBuilder();
            for (byte elem: hash) {
                String hex = Integer.toHexString(0xff & elem);
                if(hex.length() == 1) hexString.append('0');
                hexString.append(hex);
            }
            return hexString.toString();
        }
        catch(Exception e) {
            throw new RuntimeException(e);
        }
    }
}
```

Try to create at least 10 blocks in this stage. After the creation, validate the created blockchain using your validation method.

§2. Output example

The example below shows how your output might look. To be tested successfully, program should output information about first five blocks of the blockchain. Blocks should be separated by an empty line..

```
Block:
Id: 1
Timestamp: 1539810682545
Hash of the previous block:
0
Hash of the block:
796f0a5106c0e114cef3ee14b5d040ecf331dbf1281cef5a7b43976f5715160d

Block:
Id: 2
Timestamp: 1539810682557
Hash of the previous block:
796f0a5106c0e114cef3ee14b5d040ecf331dbf1281cef5a7b43976f5715160d
Hash of the block:
717242af079ccb7dd44c3f016936a81cf8ab2d4c1901243f30cbb7daa2060a0d

Block:
Id: 3
Timestamp: 1539810682558
Hash of the previous block:
717242af079ccb7dd44c3f016936a81cf8ab2d4c1901243f30cbb7daa2060a0d
Hash of the block:
28a2269bb34abd01dee9cea03400345bc9ea7322d73d3263221a47c6d970404f
```

Code Editor | IDE

Solve in IDE

IDE is opened

If you don't see your IDE opened, switch to it manually

This content was created 11 months ago and updated 7 days ago. Share your feedback below in comments to help us improve it!

Comments (2) Hints (0) Useful links (0) Solutions (0)

Share something, Sergey Kubatko

Post Please do not post solutions here

Sort by: Last posted

.... about 1 month ago Report

It says to create, at least, 10 blocks. But the test only asks for 5. Change that pls.

♡ 0 Reply

RI **Rinchin Iakovlev** 8 months ago Report

Please think of a better solution than just Ctrl+V the result in a big text chunk.

♡ 0 Reply