CSI 2110 Assignment 2 - Blockchain

Miner ID: snaya091

Class and Method Descriptions:

Blockchain: A List with elements of type Block

Methods Used:

`public static Transaction promptTransactionDetails (Scanner scanner)`

- Prompts user to input details of Transaction.
- Has checks to insure numeric value is inputted for amount transferred.
- Outputs object of type Transaction.

`public static void promptFileSaveOptions (Scanner scanner, BlockChain blockChain)`

- Prompts user to input file details for output file in which blockchain will be saved
- Used in Main method with option #3
- Does not have a return type

`public static void main (String [] arg)`

- Calls validation method to validate input blockchain from file
- Prompts user for inputs on actions required
- Repeats in while loop until user breaks it by inputting “3” (exits program)

`public BlockChain(List<Block> listOfBlocks)`

- Constructor of Blockchain that initiates the attribute of <List> of elements Block

`public int size()`

- Returns number of blocks in blockchain

`private static Transaction getTransaction(String sender, String receiver, String amountStr)`

- Returns object of type Transaction after the String amountStr is parsed into an integer type

`public static BlockChain fromFile(String fileName)`

- Goes through a file and stores transactions in block and blocks in a blockchain
- Uses Buffer Reader to read file

`public void toFile(String fileName)`

- Writes to file name using an Iterator and a BufferedWriter

`public boolean validateBlockchain()`

- Creates hashmap of User and Balance
- Verifies if Blockchain is valid using multiple conditions
- Returns boolean value (True or False)

`public int getBalance(String username)`

- Returns Balance of corresponding username

`private String getRandomAsciiString(int nbOfChars)`

- Generates and returns a random ASCII String

`public void add(Block block)`

- Adds block to blockchain
- Contains Proof-of-work (will be explained)

Block: Object representing a transaction in a Blockchain

Methods Used:

- Consists of 2 Constructors, 6 Getters (1 for each parameter), 2 Setters (Hash and Nonce), toString()

Transaction: Object representing a movement of bitcoins between Sender and Receiver

Methods Used:

- Consists of 1 Constructor, 3 Getters (1 for each parameter), toString()

Proof-of-Work (Generation of Nonce) Algorithm:

A random nonce is generated using the getRandomAsciiString method. The characters are chosen from 94 available ones. String Builder is used to create and append the random characters chosen together.

The lowest number of characters for Nonce is set to 3 and the max is set to 20. The algorithm within the add method uses all the possible combinations of 94 characters for the amount of characters for the Nonce (amount of combinations is therefore 94 to the power of length of nonce). The program generates a random nonce value (starts with nonce of length 3). From this random nonce, it generates a Hash (using Sha1 class provided). If the hash generated starts with 00000, the block is added to the list of blocks with that particular nonce and hash value. If it does not generate a hash starting with 00000, a new Nonce is generated at the same length as before and the process is repeated. The length of the nonce is only changed after the number of nonces generated crosses the amount of possible combinations for that length (ex. For length 3, it would be 94^3 = 830584 combinations). This entire process repeats until a nonce that generates a desired Hash is found. In the 10 test transactions, the nonce never crossed length of 4. It was of length 4, 6/10 times and was of length 3, 4/10 times.

Table 1: Transaction Statistics

Transaction	Sender	Receiver	Amount	Nonce	# of Hash Trials
1	bitcoin	sanat	100000	:\%	402486
2	sanat	lucia	1000	(^#T	1412535
3	lucia	bob	10	w.1V	1233746
4	sanat	bob	100	w`Ha	1093521
5	bitcoin	lucia	100	ucfj	949622
6	robert	sanat	1	'#Ye	4025475
7	sanat	satoshi	1000	|9#	201652
8	satoshi	mario	500	PqT	2264
9	satoshi	luigi	100	;BG?	2713441
10	luigi	sanat	1	f,H	156576

Average # of Hash Trials = 12191318 / 10 = **1219131.8 Hash Trials**