Miner ID : knguy082

**Method explanations**

* Transaction.Transaction:constructor with 3 string inputs to set as global variables
* Transaction.getSender:returns the name of person sending money
* Transaction.getReceiver:returns the name of person receiving money
* Transaction.getAmount:returns amount of money being transfered
* Transaction.toString:returns a string of the transaction information
* Block.Block:constructor for when reading blocks from file since all the information is already available sets them as global variables
* Block.Block:constructor for when making new blocks from transactions as not all the information is available, runs through methods then sets them as global variables
* Block.setHash:this sets the hash then runs nonce to get a hash starting with "00000"
* Block.nonce:this method makes a random string between 1 and 20 characters for ASCII 33 - 126, then checks if it starts with "00000" if so it continues on keeping track of loop and time (more info below)
* Block.setPHash:sets previous blocks hash
* Block.getPHash:returns previous hash
* Block.getIndex:return index of block
* Block.getTimeStamp:get the timestamp in long
* Block.getHash:get the current blocks hash
* Block.getNonce:returns the nonce
* Block.getTransaction:returns the transaction that happened in this block
* Block.toString:returns a string of all the blocks information
* BlockChain.fromFile:creates a blockchain instance from a file name
* BlockChain.BlockChain:validates the blockchain from a file to make sure it has not been tampered with
* BlockChain.validateBlockChain:validates blockchain and returns true if it is validated
* BlockChain.toFile:writes to a file containing the filename, writes the blocks so that it can be validated next time it is opened
* BlockChain.validateTransaction:this methode validates all existing blocks and their transactions to make sure that no one has ever been negative if at some point a user has gone negative this will return false even if the user is positive at the end of the blockChain
* BlockChain.getBalance:returns the balance of the user which is the input
* BlockChain.readFile:reads the file and puts the information into the correct block variable
* BlockChain.main:the main methode where everything starts
* User.User:constructor for name and balance
* User.getBalance:returns balance
* User.getName:returns name
* User.add:add amount to user

**How the nonce works**

The nonce method starts by initializing a variable for loop and start time. Then it runs a do while loop. Every loop the variable “loop” will be incremented. In every do while loop there will be a for loop. The for loop will have a pseudo random length between 1-20. In each loop a pseudo random character between 33-126 ASCII character will be chosen (I picked to go along the path of pseudo random as the professor clarified that security wasn't a priority in this assignment). This for loop will give a pseudo random number between 1-20 in length. After this the do while loop will reset if the hash does not start with 00000 resetting the nonce along with it. If the hash does start with 00000 the nonce will be set to the block and the final time will be stopped to find the total runtime.

**Statistics**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Sender** | **Receiver** | **Amount** | **Loops** | **Time(millis)** |
| 0 | lucia | khang | 20 | 443016 | 1632 |
| 1 | robert | khang | 20 | 2025807 | 6417 |
| 2 | khang | lucia | 30 | 1306846 | 4185 |
| 3 | lucia | john | 10 | 3873114 | 12247 |
| 4 | lucia | khang | 1 | 345637 | 1099 |
| 5 | khang | john | 6 | 2060473 | 6449 |
| 6 | lucia | khang | 20 | 1470573 | 4682 |
| 7 | khang | robert | 5 | 1133406 | 3562 |
| 8 | john | lucia | 6 | 1128918 | 3526 |
| 9 | john | khang | 5 | 626748 | 1953 |
| 10 | robert | khang | 3 | 74476 | 587 |
| 11 | khang | robert | 3 | 425167 | 3669 |
| 12 | khang | lucia | 10 | 1650919 | 13092 |