CSI Assignment 2 Report

Miner ID: nshar082 Nischal Sharma Student ID: 300023475

Class Transaction

- This class stored the information of sender, receiver, and amount.
- public Transaction(String sender, String receiver, String amount). This is the constructor and it takes the required inputs.
- getSender(), getReceiver(), getAmount(), toString() are the 3 getter methods that returns sender, receiver, and amount variables along with toString() that returns String format.

Class Block

- This class stores the blocks for the block chains. It contains the index of block **(index)**, time stamp of block creation **(timestamp)**, transaction class information **(transaction)**, nonce value **(nonce),** hash of the previous block **(previousHash)**, and block's **(hash)**.
- There are 2 constructors.
    - Block(int index, Transaction transaction, String previousHash) when block wants to be created
    - Block(int index, Transaction transaction, String nonce, String previousHash, long timestamp, String hash) when reading from a block
- computeHash() is the method that will generate random nonce values to get a hash that starts with "00000" for the block.
- getIndex(), getTimeStamp(), getTransaction(), getNonce(), getPreviousHash(), getHash(), toString() are getter functions for the block and with To String function to output string.

Class BlockChain

- main(String[] args) is the main function for this class. It takes input for file name and reads from file which adds it to the blockchain instance if itself. Then it validates the blockchain from the file and asks for transactions. if the transaction balance is wrong, it will ask again. The transaction is created when all values are right, and it will add the transaction to the block which is then added to the blockchain. It then prompts user if they want another transaction. Typing "yes" or "no" will lead to output. After that, the block chain is validated, and the block chain is converted into a text file appending the name of the input file and adding my miner id "_nshar082" in the file name.
- BlockChain(ArrayList<Block> blockList) is the constructor that contains the blocks. The class variable is ArrayList<Block> which the constructor takes as a parameter.
- fromFile(String fileName) takes the name of file to read from and converts them into blocks. These blocks are then added into an Array List of blocks called blockchain. After that it returns this Array List of Blocks. It uses Buffered Reader to read the file.
- toFile(String fileName) takes the file name as a parameter and writes the information regarding the blocks in the block chain in a text file. It uses Print Writer to write into the text files.
- getList() is a getter function that returns the blocklist which is the Array List that holds the collection of blocks.
- validateBlockChain() is a function that returns Boolean value. It checks 3 different things for the block chain. It checks if the index is the same as the block number for every block. If there is any error, it will return false. Then, it will check for the hash compatibility. For every block, it will take the block information and use SHA-1 to

generate the hash code for that block. It will then check if the hash code is the same as the previous hash stored for the next block in the list of block chains. If they are different, then the text file has been tampered with and it will return false. Finally, it will check the total balance of the block chain. It takes the users and puts them in an array list. Removes the duplicates and calls the function getBalance(String username) for every user in the block chain. If any of the users other than bitcoin has a negative balance, it will return false.

- getBalance(String username) is a function that will check the balance of the user in the blockchain. It takes every block and checks if the user sent or obtained bitcoins. It will total the sent as a negative value and obtained as positive. It will then result a value.
- add(Block block) is a setter function that adds the block into the Array List of block chains blockList.

Proof of work:

1. The method computeHash() computes the Hash for the block which is found in class Block. The method starts with a infinite while loop which only stops after nonce value has been found. This means the runtime for this loop is O(n).
2. Inside the loop, another for loop is done which loops from at least 5 to maximum of 10 times. This loop is random and it finds the size for a random nonce string. This will make the runtime of the loop O(n^2).
3. Once the nonce value is generated, it will leave the loop and set the block nonce value to the random string generated in the loop above. It will use SHA-1 and find an experimental hash value. If the hash value generated starts with "00000" then the block hash value is set to that hash and the while loop is broken using break; If that is not so, then the loop starts all over again. The runtime is still O(n^2).

Overall, the runtime is O(n^2) for the algorithm to generate a random nonce value.

Table of statistics: Trials for all 10 transactions done.

Trial Index	Number of Trials
1	**684388**
2	**552952**
3	**159698**
4	**374825**
5	**2037686**
6	**310850**
7	**39601**
8	**111574**
9	**1790411**
10	**3690490**
Median	**975247.5**
Average (Mean)	**463888.5**

The trials were performed in bitcoinBank_nshar082.txt. Each transaction took approximately one or two seconds to process. All the validated text files are stored in /Validated/ directory. It includes all test transactions done by class mates (miners).