Assignment 2

CSI2110 - Data Structures and Algorithms

Fall 2018

School of Electrical Engineering and Computer Science

University of Ottawa

Course Coordinator: Lucia Moura

Said Ghamra: 300008217

Email: sgham022@uottawa.ca

Miner ID: sgham022

Submission Date: October 22, 2018

This project contains 4 classes: **Sha1**, **Transaction**, **Block**, and **BlockChain**.

- **Class Sha1:** The class Sha1 is responsible for generating the Sha1 hash of the string representation of the block.

- **Class Transaction:** The class Transaction is used to store the information about the sender, receiver, and the amount of bitcoin transferred. It also contains all the necessary methods.

- **Class Block:** The class Block is used to store the index of the block, timestamp object, transaction object, nonce, previous hash, and the hash of the block. The primary function of the Block class is to generate the hash of the block (using the generateHash method) , which should contain five zeros in the beginning, by randomly generating a nonce that varies in length from 1 to 20 characters that are in the integer ASCII range of [33,126].

  Nonce Generation:

  In order to randomly generate the nonce, which occurs in the generateHash method, a String variable was used. The string variable starts at length 1, i.e. containing 1 random ASCII character in the range of [33,126]. The program then generates the hash of the block with the current value of the nonce (set to the value of the String variable) and checks if the hash contains five zeros in the beginning. If it doesn't, the program resets the String variable and adds 2 random characters and repeats the process all the way up to 20 characters. If no suitable value for a nonce is found, the program resets the String variable and increments it with 1 to 20 characters until a suitable nonce is found. The program checks the hash of the block at every step of the way and stops when a nonce value is found that results in a hash with five zeros in the beginning.

- **Class BlockChain:** The class BlockChain is primarily used to store all the blocks in an ArrayList. This class also contains a main method that follows the sequence of execution specified in the assignment report. When the Class BlockChain is ran, it first prompts the user for the filename of a text file containing all the information of a blockchain. The method validates the file name and makes sure it exists in the projects main directory. The method then reads the file (using the fromFile method) and generates a blockchain object based on the information present in the text file. It then checks whether the blockchain is valid or not (using the validateBlockChain method). If the blockchain is not valid, it will print so and the program terminates. However, if the blockchain is valid, the user will be prompted with whether he/she would like to add a transaction or not. If the user wants to add a new transaction, the program prompts the user for the name of the sender and receiver and for the amount of bitcoin to be transferred. The program then validates the transaction and if it is valid it gets added to the blockchain. The program will keep prompting the user whether they would like to add a transaction until the user

inputs no. The blockchain is then exported to a text file with a specific file name and format.

Notes:

- A detailed description of every method can be found in every class if further clarification is needed.

- It is also worth mentioning that a transaction was programmed not to accept any amount that is less than or equal to zero. It doesn't make sense for a user to send zero bitcoins to someone else.

- When a new block is created, the hash value found in the text file is not passed through to the block object. Instead, when a new block is created, the correct hash is generated and is stored in the hash variable inside that block object. In order to compare the hash values found in the text file and the correct hash for the block stored inside the block object, an ArrayList was used in the class BlockChain that stores all the hashes given for every block inside the text file. The program then proceeds to compare the hashes in the validateBlockChain method.

The following table contains the amount of trials needed in order to generate the nonce for every transaction that results in a hash with five zeros in the beginning:

The average number of trials is 1,020,650 trials approximately.

Transaction Number	Amount of Trials
1	1651814
2	3894400
3	7343
4	122260
5	2372706
6	269490
7	1161825
8	106522
9	401638
10	218495