Bitcoin Simulation Programming Assignment #2 (BONUS 5pts)

Due on Thursday, December 7th before the end of class

- **This is OPTIONAL. Not completing this assignment will NOT impact your grade in anyway. Completing the assignment will only increase your FINAL grade with maximum 5pts.**
- **All the codes should be written in C or C++ for Linux and commented appropriately for major steps. Make sure your codes do compile!**
- **The codes should be submitted through Blackboard.**
- **Late submission is not accepted.**

Requirements:

- Your program should accept two name files passed as command line arguments.
- Example of running your code:

```
yourLastname_yourFirstname_BTC blockchain.txt transaction.txt
```

- The first input ASCII file contains the headers of all the blocks in the blockchain. Example `blockchain_1.txt` (attached):

```
026765a1c8235d4ac23d2582cda3b9f5c062f805540320173eb9e9148c0dc518 704b42e4b11ca131b443c2b02a07ec0b45407f1b125027e3e68b86ace6924458 00000001
0b53181ae351f4508363cdc3e8fb3e819fb706c4ba98a3005a980a837561074a 06aa7a8cbda7ac4351c0cae116c589c2eb0ca96cb4c90844812945cb4ffe27c5 00000019
00000000000000000ad6e90c0790e83760a9d13728c23474352a2c8c7a6e0eb 2b12fcf1b09288fcaff797d71e950e71ae42b91e8bdb2304758dfcffc2b620e3 0000000f
```

- Each line represents a block with the following fields separated by space where all the values are given in hexadecimal:

Field	Size	Description	Example
Previous Block Hash	32 bytes	A reference to the hash of the previous (parent) block in the chain	`026765a1c8235d4ac23d2582cda3b9f5c062f805540320173eb9e9148c0dc518`
Merkle Root	32 bytes	A hash of the root of the merkle tree of this block's transactions	`704b42e4b11ca131b443c2b02a07ec0b45407f1b125027e3e68b86ace6924458`
Nonce	4 bytes	A counter used for the proof-of-work algorithm	`00000001`

- The first line corresponds to the latest block created in the blockchain. This block points to the previous block that is on the second line and so on.
- The second input ASCII file contains N transactions, where N is a perfect power of 2, example 2, 4, 8 etc. Example `transactions_1.txt` (attached):

```
Bob Alice 5.0
Alice Bob 1.0
John Bill 2.4
Bill Alice 1.3
John Bill 2.7
Bob John 7.9
Tom Todd 4.5
Todd Bob 12.0
```

1. Your code should check whether the blockchain is VALID. This implies that every block must be valid and the chain is not broken.
    a. A chain is valid if its hash value starts with 4 zero bits. Example: the hash of the second block is **026765a1c8235d4ac23d2582cda3b9f5c062f805540320173eb9e9148c0dc518** and starts with 0 in hex (namely 4 zero bits) thus the second block is valid.
    b. Note that the hash of a particular block is not in the header. The header only contains the hash of the previous block. To get the hash of a block one needs to concatenate the three fields, convert the hex to string and hash the result using SHA256. Please check `test.cpp` for helping function.
    c. If all the blocks are valid then the chain should be checked to see whether it is broken. Namely the first block should point to the next one and so on.
2. If the blockchain is INVALID then your code should print out the root of the merkle tree corresponding to the transactions in the second file. Each line in the file is a transaction. To get the transaction hash just pass the corresponding line through the SHA256 function. Then to get the parent of two transactions just concatenate the two corresponding hashes and pass the result through SHA256 and so on. The merkle root of the above transaction list is:

    91fc3bf900a0558a6ead455c2f2c440302d0c33d17c408dd3ce384620d452c70

3. If the blockchain is VALID then your code should print out the fields corresponding to a new block that will point to the latest block in the blockchain, has the root of the merkle tree corresponding to the transactions in the second file and the nonce should be found (mined) such that one has a valid block. Example output in this case:

    09ebbf28bacb7910d24dc1ea7a3a3480611a263e7c8163276a1a14f812d43ffd
    91fc3bf900a0558a6ead455c2f2c440302d0c33d17c408dd3ce384620d452c70
    0000000d

    the hash of this new block is given by (do not print this – just for example purposes):

    03b93c10887c7f015dd9524d39f3d763a2f43872f7555724b552de94132e0f25

A script file (Makefile) or readme file should be submitted together with your codes providing instructions on how to compile your codes. Archive all your files into a zip file named "yourLastname_yourFirstname_CSCE350.zip" and submit it through Blackboard.