CS61065 Theory and Applications of Blockchain

Assignment - 1 (Autumn 2022)

Deadline: September 01 2022 EOD

Each student needs to solve and upload this assignment individually. Please check the submission instructions at the end.

We have a zero-tolerance policy on plagiarism. Any form of plagiarism will lead to zero marks for all the cases that will get detected.

There are no restrictions on the programming languages to be used. We would prefer Python, but you can choose your favorite programming language.

Part A

Write a program to verify the Merkle root for a block. Each block can have **T** transactions. The program has to construct the merkle tree for these T transactions, and find the root hash of the merkle tree.

Input:
The test case contains **B** blocks. The first line of the input will be the integer **B**. This is followed by inputs for **B** blocks. First line of each block input contains an integer **T**, the number of transactions in that block. This is followed by **T** lines, each a string of length **S** representing a transaction (the string will have lowercase characters only). These **T** lines are followed by a md5 hash, denoting the merkle tree root formed of these T transactions. Then follows the input for the next blocks.

Output:
For each block in the test case, print **Valid** / **Invalid**. If the merkle tree root provided in the test case is correct, print "Valid", otherwise print "Invalid".

NOTE: While taking md5 hash, convert it to the hexadecimal string representation.

Constraints:
$0 < T \leq 50$
$S = 20$

Example:

```
5
1
ylrpbnsyyjpkqmujxhvm
cb2b070a0cd4265fe84e149e31fd35ce
3
ysaqiblcmbaawxdixzrs
pnbcgrefcuqxnftimghe
tzniehbjkldzxucqtufw
02569e697adb12fe7bd60f688d75566a
2
vkqcjncnswfsdzddyiwl
fuewvxwdvypokxdtzxcl
06ca63ce836c169c869ef2dcb060fab0
3
xbrffuhusuvbbwyftler
yfpahwrtmgpbiysguxmu
uesjywnfcqsgggpkwkvg
75b5eca04a5285d339b6c5ab6f82c02b
5
rditxaunslsgmcujqgzs
khhwesslqwluflrdlubt
zcxrxrqlrvwdspsvgxdk
ogdatjvoejjgbyhpocou
piwwizxjpudjzhfjboqz
e2aa32f996404a240f0e2f58c06e5ace
```

Outcome

```
Valid
Valid
Valid
Invalid
Valid
```

Explanation

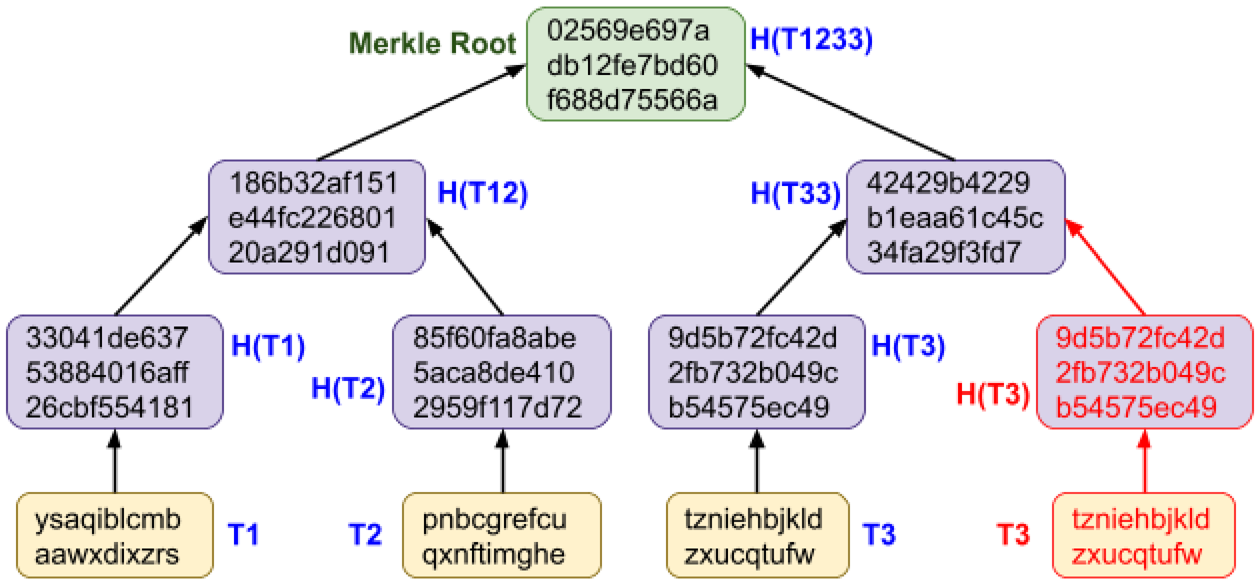

Fig.3.

Fig.3 is the Merkle Tree of the 2nd block of the example. The markle root in Fig.3 matches with the given value “02569e697adb12fe7bd60f688d75566a”. Therefore, the output for that block is “Valid”.

There are only three (odd number) transactions in this block, therefore T3 is considered twice (marked with red).

Part B

Write a program to construct blocks from a set of transactions and then verify whether the given previous block header hash is correct or not. The program has to construct B blocks, each of which are connected to the previous block using the previous block header hash, thus forming a linked list like structure, - the blockchain. The first block (genesis block) contains only one given transaction. The rest of the B blocks consist of T given sets of transactions. For each block, find the hash of the block header and verify if the previous block header hash is correct or not.

For each block, the block version is '02000000'.

Hash of the block header = Hash(block version + previous block header hash + merkle root)
*merkle root computed in Part A.

NOTE: Here Hash denotes md5 hash. While taking md5 hash, convert it to the hexadecimal string representation.

Example:
Block version: “02000000”
Previous block header hash: “b94950dbb12a8f1b829a6d0f348e75c0”
Merkle root: “328ae47c50e99e2aaac8f05af0874787”
Hash of current block header:
md5(“02000000b94950dbb12a8f1b829a6d0f348e75c0328ae47c50e99e2aaac8f05af0874787”)

Input:
The test case contains **B** blocks. The first line of the input will be the integer **B**. This is followed by information about B blocks. First line of each block contains the **T** number of transactions of a block. This is followed by **T** lines, each a string of length **S** representing a transaction (the string will have lower case characters only). These **T** lines are followed by a **md5 hash**, denoting the hash of the previous block header.

The first block (genesis block) is not included in the input. Therefore, the first block in the input is actually the second block in the blockchain. The genesis block is formed of only one transaction. This is the coinbase transaction with the string **'coinbase'**.
Hash of the coinbase block header = Hash(block version + merkle root)

Output:
For each block in the test case, print **Valid** / **Invalid**. If the previous block header hash provided in the test case is correct, print “Valid”, otherwise print “Invalid”.

NOTE: You have to construct the blocks only with the transactions given as input. The previous block header hash given in the input is only for you to match with the actual hash your are getting for the previous block and print Valid / Invalid.

Constraints:
0 < T <= 50
S = 50
Block header contains only version, previous block header hash and merkle root.
Block Version = ‘02000000’
Coinbase Txn = ‘coinbase’
For the coinbase block, the previous block header hash is None or ‘’.

Complete Example of Input & Output:

Input

```
5
1
ylrpbnsyyjpkqmujxhvm
69e8f1dd247471c70fd4d9d0f3ea7789
3
ysaqiblcmbaawxdixzrs
pnbcgrefcuqxnftimghe
tzniehbjkkdzxucqtufw
4b310daef2e8455c56fb81354735dfd9
2
vkqcjncnswfsdzddyiwl
fuewvxwdvypokxdtzxcl
ccc2da9688172a8f9554cdedd003cfe2
3
xbrffuhusuvbbwyftler
yfpahwrtmgpbiysguymu
uesjywnfcqsgggpkwkvg
5d101f82bf134eca2091973c2b7af918
5
rditxaunslsgmcujqgs
khhwesslqwluflrdlubt
zcxrxrqlrvwdspsvgxdk
ogdatjvoejjgbyhpocou
piwwizxjpudjzhfjboqz
468e1c2487c24f7646df804472153081
```

Output

```
Valid
Valid
Invalid
Valid
Invalid
```

Explanation

The file given 1st previous block header hash “69e8f1dd247471c70fd4d9d0f3ea7789” is equal to the previous block header hash (i.e. coinbase block header hash) of the 1st block data. Therefore, the 1st outcome is “Valid”.

Submission Instructions:

Both Part A and Part B of this assignment have to be submitted in HackerRank as well as Moodle. If you do not have a HackerRank ID, please create one.

For Moodle submission, use CSE Moodle: https://moodlecse.iitkgp.ac.in/moodle/login/index.php. We have created a course page there. You can join the course page using the key **TAB22STU** .

Make sure you upload the same code in both the portals.

HackerRank Contest Link:
https://www.hackerrank.com/cs61065-2022-assignment-1

Fill up the following Google form to map your roll number with your HackerRank ID:
https://forms.gle/cGCYsatLvKWr9YMw8