Level 5 - Blockchain

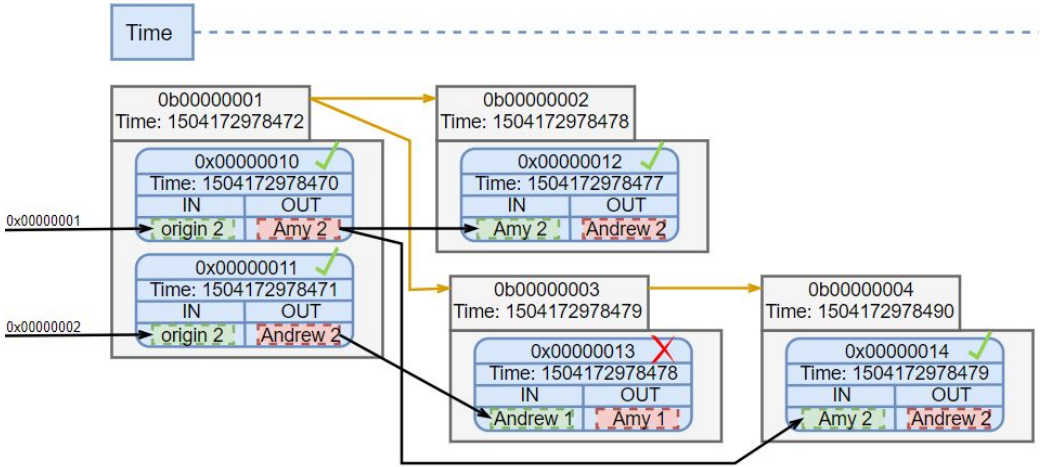
CODING CONTEST

Definitions

- Funds are passed on through transaction linking
- Transactions need to be included in blocks to be booked

Task

- Report the transactions in the longest chain of valid blocks
    - If there are multiple, the one that reached their last block earlier
- Report the longest chain of blocks



Blocks

CODING CONTEST

Blocks

- Have exactly one predecessor
  - Except block **0b00000000**, which is the root block (**not included** in level input/output)
  - The predecessor block must have been generated before
- Can carry 0-20 transactions, empty blocks are possible
- Included transactions have to be submitted **at latest at the time** the block is generated
- A block is only valid if all included transactions are valid
  - Criteria for valid transactions as defined in level 3
  - Inputs of transactions must be outputs on the respective chain of predecessor blocks

Blockchains

CODING CONTEST

Blockchain

- Blocks are chained together by pointing to their predecessors
- All transactions carried by a block need to have valid inputs on the blocks branch
- The root block is **0b00000000**, it is **not** included in the input and should not be included in the output
- A valid blockchain consists of valid blocks only
- A valid blockchain's first block has the root block as a predecessor

Data format

CODING CONTEST

Input

`<NumberOfTransactions>` the number of transactions in the banking system
NumberOfTransactions lines: `<TransactionID> <NumberOfInputs>` NumberOfInputs * InputElement `<NumberOfOutputs>`
NumberOfOutputs * OutputElement `<TransactionSubmitTime>`
InputElement: `<InputTransactionID> <InputTransactionOwner> <InputTransactionAmount>`
OutputElement: `<OutputTransactionOwner> <OutputTransactionAmount>`
`<NumberOfBlocks>` the number of blocks in the banking system

NumberOfBlocks lines: `<BlockId> <PreviousBlockId> <NumberOfTransactions>` NumberOfTransactions *
`<TransactionId> <BlockCreationTime>`

Data format

CODING CONTEST

Output

`<NumberOfTransactions>` the number of transactions in the longest valid blockchain
NumberOfTransactions lines: `<TransactionID> <NumberOfInputs>` NumberOfInputs * InputElement `<NumberOfOutputs>`
NumberOfOutputs * OutputElement `<TransactionSubmitTime>`

InputElement: `<InputTransactionID> <InputTransactionOwner> <InputTransactionAmount>`
OutputElement: `<OutputTransactionOwner> <OutputTransactionAmount>`

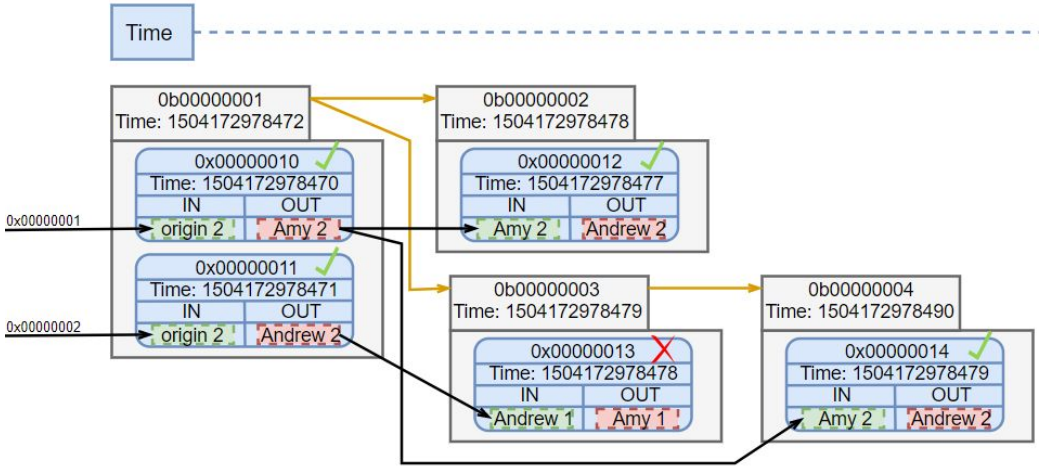
`<NumberOfBlocks>` the number of blocks in the longest valid blockchain
NumberOfBlocks lines: `<BlockId> <PreviousBlockId> <NumberOfTransactions>` NumberOfTransactions * `<TransactionId> <BlockCreationTime>`

Example

CODING CONTEST

Input

```
5
0x00000010 1 0x00000001 origin 2 1 Amy 2 1504172978470
0x00000011 1 0x00000002 origin 2 1 Andrew 2 1504172978471
0x00000012 1 0x00000010 Amy 2 1 Andrew 2 1504172978477
0x00000013 1 0x00000011 Andrew 1 1 Amy 1 1504172978478
0x00000014 1 0x00000010 Amy 2 1 Andrew 2 1504172978479
4
0b00000001 0b00000000 2 0x00000010 0x00000011 1504172978472
0b00000002 0b00000001 1 0x00000012 1504172978478
0b00000003 0b00000001 1 0x00000013 1504172978479
0b00000004 0b00000003 1 0x00000014 1504172978490
```



Output

```
3
0x00000010 1 0x00000001 origin 2 1 Amy 2 1504172978470
0x00000011 1 0x00000002 origin 2 1 Andrew 2 1504172978471
0x00000012 1 0x00000010 Amy 2 1 Andrew 2 1504172978477
2
0b00000001 0b00000000 2 0x00000010 0x00000011 1504172978472
0b00000002 0b00000001 1 0x00000012 1504172978478
```


Level 5