Exercise 3

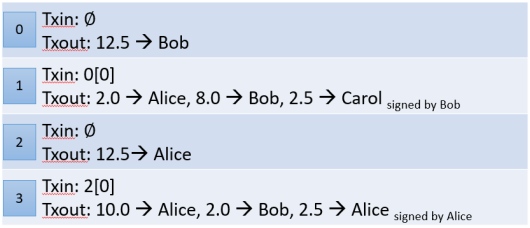
A. Transactions

1. Consider the following transactions in a transaction based ledger. Check if the transactions are valid. If valid calculate the balances of each person.

(a)

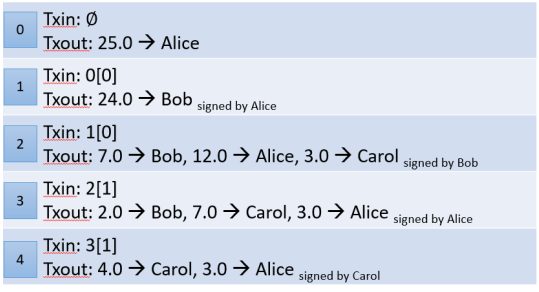
Solution:
Transactions are valid
Alice = 7.0 Bob = 12.0 Carol = 6.0

(b)



Solution:
Transactions are not valid. At Tx3 the Txin 2[0] has 12.5 coins whereas the Txouts sum up to 14.5 coins. Even though Alice has a balance of 14.5 coins Tx3 is not valid as $\sum Txin < \sum Txout$. To make the transaction correct, Tx3 would not only have to use Txin 2[0], but also Tx1[0].

(c)



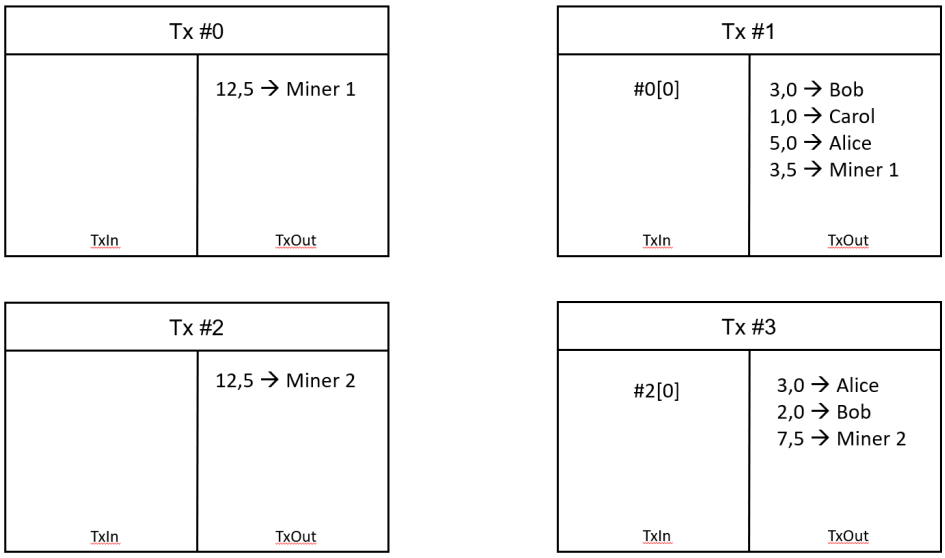
Solution:
Transactions are valid
Alice = 6.0 Bob = 9.0 Carol = 7.0
In this case at Tx1 Alice, and at Tx2 Bob both do not redeem the full amount remaining from the respective Txin. Those balances can be claimed by miners as a transaction fee.

(d)

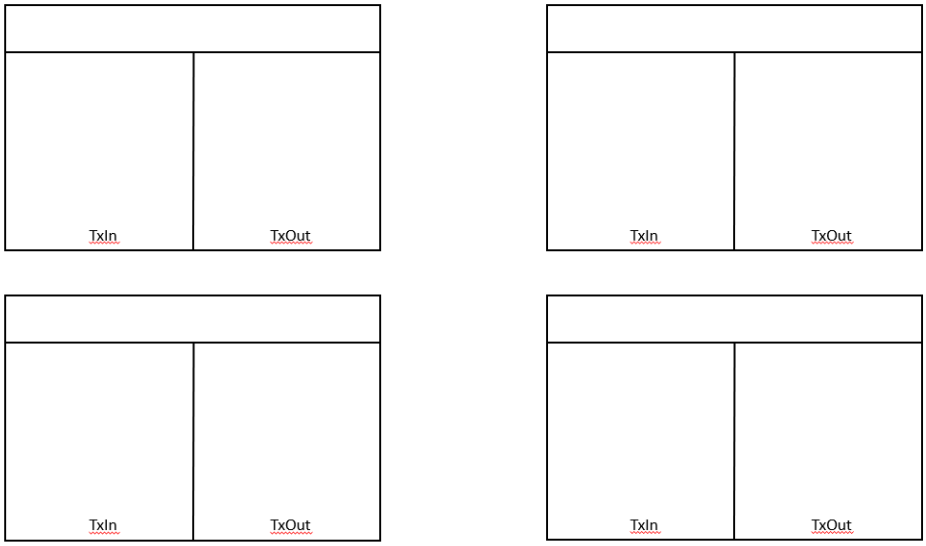
0	Txin: Ø Txout: 25.0 → Carol
1	Txin: 0[0] Txout: 6.0 → Bob, 6.0 → Alice, 13.0 → Carol signed by Carol
2	Txin: 1[1] Txout: 2.0 → Bob, 4.0 → Alice signed by Bob
3	Txin: 1[2] Txout: 3.0 → Bob, 7.0 → Carol, 3.0 → Alice signed by Carol

Solution:
Transactions are not valid. At Tx2 the Txin 1[1] is owned by Alice. Therefore Bob cannot use this Txout for his transaction. He would have to use Txin 1[0].

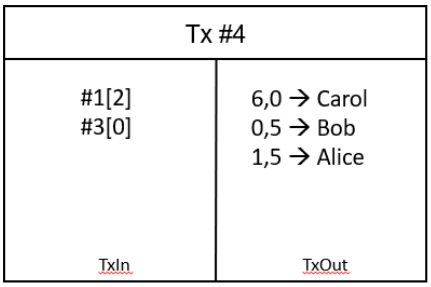
2. Below is the representation of four transactions in the Bitcoin network where Alice receives Bitcoins from two different miners. Transaction fees are ignored.



Alice now wants to make two payments. She wants to transfer Carol 6,0 BTC and Bob 0,5 BTC. Draw the necessary transactions for Alice using the notation of diagram above.



Solution:



Different combinations of transactions are also possible, such as sending first to Carol and then to Bob. In the context of our exercise, it is okay to create two or three transactions. Please note: It could be possible that we ask about the "solution with the minimum amount of transactions", which would result in the above presented solution as the only correct one.

3. Bitcoin clients and exchanges provide "block explorers" that let users search transactions, blocks, addresses and other relevant blockchain network information. One of the well-known Bitcoin block explorer is `https://www.blockchain.com/explorer`.

   Visit the block explorer and find the following information for the Bitcon blockchain:

   (a) What is the current hash rate?

   Solution:
   Current hash rate is around 51,000,000 TH/s (22nd of May 2019)

(b) What was the all time peak value of unconfirmed transactions and when has it occured?

Solution:
The peak value is 185,258 transactions unconfirmed at 08.12.2017. Please note that there is no "objectively correct" number to the highest amount of unconfirmed transactions in the network. Different nodes have a different perception of the network. E.g., other statistics report a highest amount of unconfirmed transactions in the mempool on December 22, 2017 with a peak value of roughly 261,000 transactions.

(c) Find the transaction a1075db55d416d3ca199f55b6084e2115b9345e16c5cf302fc80e9d5fbf5d48d. Fill the following information:

(a) Block of the transaction

Solution:
Block number 57043.

(b) Sender and the receiver

Solution:
Sender: 1XPTgDRhN8RFnzniWCddobD9iKZatrvH4
Receiver: 17SkEw2md5avVNyYgj6RiXuQKNwkXaxFyQ

(c) The value of the transaction

Solution:
10,000 Bitcoins (plus 0.99 BTC fee)

(d) What is particular about this transaction?

Solution:
This transaction was sent from a user in 2010 for a pizza. At that time it was worth around 25$. Please note that Blockchain.com website displays the US$value based on the current price of Bitcoin.

B. Bitcoin Script

4. Take a look at Bitcoin script's opcodes in the slides. Which fundamental commands are missing? What could be the reason they are not added?

Solution:

The Bitcoin script does not contain loops or jumps. The main reason is to avoid infinite loops and therefore avoid the halting problem (just for your information, the halting problem is not part of the lecture.). The Bitcoin network requires complete determinism as any difference between nodes could lead to forks.
Bitcoin is specifically designed as a digital currency. However, blockchains with loops and jumps (Turing complete) do exist. E.g. Ethereum. We will see more about Ethereum and its script in following lectures.

5. Alice wants to protect her Bitcoins and therefore her unspent transaction outputs. She decides to protect it with a password. She hashes the password and writes a script: The script requires the person (which intends to spend the output) to provide the password as an input. This input is hashed and compared to the predefined hash, proving that Alice spends the transaction. What are the possible flaws with this Bitcoin Script?

Solution:

- Every node that sees the transaction can change the Txout (as long as it is not mined). There is no signature to ensure the will of the user contained in the transaction, therefore arbitrary transactions can be generated. In other words: Alice cannot enforce her defined transaction output, as there is no way to bind the output to her transaction input, in which she enters the password.

6. There is an op_code called locktimeverify and a time lock in the transaction itself. What is the difference? What are examples in which these are useful?

Solution:

The locktimeverify op_code basically locks a Txout (and therefore the associated Bitcoins) for that certain amount of time. However, the output can be mined and included in the Blockchain. An application would be: I pay you, but in a year from now without being able to interfere with the payment process. After a year, the Txout would be spendable, no matter what.
The time lock in a transaction defines the time until a transaction can be mined. Until then, it is invalid, therefore not included in the Blockchain. This is an optional insurance for micropayment channels. If the referencing txout is already spent, then the transaction is invalid.

7. Following transaction output is provided:

```
OP_DUP OP_HASH160 8a014218a5a42e2c6fc5d573ab54a91ff555d1de OP_EQUALVERIFY OP_CHECKSIG
```

(a) Can you tell which entity has created this transaction output?

Solution: No. As the we only see the transaction output, we only know the receiver of the transaction.

(b) Can you tell if this transaction output is spent?

Solution: No. We do not know if transactions exist which spend the output.

(c) Can you tell which entity is allowed to spend this transaction output?

Solution: The owner of the private key which corresponds to the public key which corresponds to the hash 8a014...1de.

(d) What specific data is required to spend the transaction output?

Solution: The public key of the hash 8a014...1de and the corresponding signature (therefore the private key).