Practical 4 Create a blockchain, a genesis block and execute it

In [1]:
```
import datetime
import collections
import binascii
import Crypto  #If not installed run pip install crypto and pip install pycryptodome
import Crypto.Random
from Crypto.Hash import SHA
from Crypto.PublicKey import RSA
from Crypto.Signature import PKCS1_v1_5
```

In [2]:
```
class Block:
    def __init__(self):
        self.verified_transactions=[]
        self.previous_block_hash=""
        self.Nounce=""
```

In [3]:
```
last_block_hash=""
```

In [4]:
```
class Client:
    def __init__(self):
        random = Crypto.Random.new().read
        self._private_key = RSA.generate(1024, random)
        self._public_key = self._private_key.publickey()
        self._signer = PKCS1_v1_5.new(self._private_key)

    @property
    def identity(self):
        return binascii.hexlify(self._public_key.exportKey(format='DER')).decode("as
```

In [5]:
```
Alice=Client()
```

In [6]:
```
class Transaction:
    def __init__(self,sender,recipient, value):
        self.sender=sender
        self.recipient=recipient
        self.value=value
        self.time=datetime.datetime.now()
    def to_dict(self):
        if self.sender == "Genesis":
            identity = "Genesis"
        else:
            identity = self.sender.identity
        #Dictionary objects
        return collections.OrderedDict({
            'sender':identity,
            'recipient':self.recipient,
            'value':self.value,
            'time':self.time
    })
    def sign_transaction(self):
        private_key = self.sender._private_key
```

```
        signer = PKCS1_v1_5.new(private_key)
        h = SHA.new(str(self.to_dict()).encode('utf8'))
        return binascii.hexlify(signer.sign(h)).decode('ascii')
```

In [7]:
```
t0=Transaction("Genesis",Alice.identity,400.0)
```

In [8]:
```
block0=Block()
```

In [9]:
```
block0.previous_block_hash=None
Nounce = None
```

In [10]:
```
block0.verified_transactions.append(t0)
```

In [11]:
```
digest=hash(block0)
last_block_hash=digest
```

In [12]:
```
digest
```

Out[12]:
```
173982780429
```

In [13]:
```
last_block_hash
```

Out[13]:
```
173982780429
```

In [ ]:
```

```