Bitcoin, Cryptocurrency, Blockchain and Python Bitcoin apps

Imran Shaik

November 10, 2017

Bitcoin

Bitcoin is a digital currency meaning it exists only as an electronic record and unlike physical currency, you cannot hold it in hand. You can send and receive bitcoin using Bitcoin address. You can pay for sevices using Bitcoin. Bitcoin is also interchangeable with flat currency. Bitcoin is the first and the most popular digital currency. It uses peer to peer technology to upgrade with no central authority or banks intervention. It is open source. It is digital and decentralized in turn greater control funds with low fees.

Bitcoin Uses

Fast and Easy payments Bitcoin is available 24/7. You can send payments internationally and there is no wait period unlike banks. Privacy Share bitcoin without using personal info like card information or signing up Negligable transaction fees Secure Secured by military grade caligraphy Multi Signature for Business to control their spending by getting authorization from selected people Great for Developers

Blockchain

Blockchain is a digital ledger. Its a public record of bitcoin transactions in chronological order. It is a permission less distributed database based on bitcoin protocol. Each user has the copy of whole block chain. It prevents double spending and it is immutable. It cannot be tampered. Each blockchain record is secured cryptographically. Blockchain is made of several blocks, which in turn is a record that contains and confirm many waiting transaction. For every 10 mintues, a new block of transactions is append to blockchain using mining. File named bitcoin-blockchain is available for all the computers at the same time containing data about all bitcoin transactions, which is also called ledger
It is a public database where new data are stored in a container called a block and are added to an immutable chain called blockchain with data added in the past. In case of bitcoin and other cryptocurrencies, these data are group of transactions. But, that data can be any type Blockchain has revolutionaized distributed computing in the form of technologies like Ethereum, which has introduced smart contracts

Blockchain vs Bitcoin

Blockchain is technology behing bitcoin working as a database for all bitcoin transactions. Digital currency is not the only use of blockchain. Some other example are digital contracts, permenant public transport ledger system for compiling data on sales etc. A digital nation called BitNation is developed through blockchain

Bitcoin Wallets

You need bitcoin wallet to start using bitcoins. Care must be taken to secure the bitcoin wallet. Bitcoin price is very volatile. Bitcoin wallets is available is so many forms like smartphone app wallet, online web wallet, desktop based wallets, hardware wallets Some examples of online web wallet are Coinbase, BitGo, BTC, Coin.space, Green address, Xapo etc. These wallets are platform independent. Mobile wallets are breadwallet, coinspace, electrum, bitcoin wallet, simple wallet, btc, copay, mycelium etc. Desktop based bitcoin wallets are Bitcoin core, bitcoin knots, bitgo, bither, armory, arcbit etc. Hardware wallets are Trezor giving more security using signature. Bitcoins can be held offline using bitcoin cold storage like paper wallet using bitcoin address from bitcoin.org and store the bitcoins offline

Bitcoin send/receive

I have personally used coinbase.com to buy, send and receving bitcoins. I also traded the bitcoins with other coins at coinexchange.com. This gave me good exposure on how to send/receive bitcoins. We can also buy bitcoin using bitcoin atm at coinatmradar.com We can also sell and buy bitcoin from other users who are interested to sell their held bitcoins using LocalBitcoins.com We can also spend bitcoin on products and services like Spendabit.co List of companies who accepts bitcoins as payments can be found at 99bitcoins.com Few examples are Subway, Microsoft, Reddit, OkCupid, Expedia, Dell, Wikipedia etc. Coinmap.org proives local business accepting bitcoins near you

Programming Bitcoin and Blockchain with Python

Install bitcoin using 'pip install bitcoin' This package doesnt need any bitcoin node on computer. It connects to blockchain.info and gets the data

```python
import bitcoin
from bitcoin import *

#generate a private key
private_key = random_key()
print("Private Key:%s\n" %private_key)

#generate public key
public_key = privtopub(private_key)
print("Public Key:%s\n" %public_key)

#create bitcoin address
bitcoin_address = pubtoaddr(public_key)
print("Bitcoin addresss:%s\n" %bitcoin_address)
#Generally we use unique bitcoin address for each transaction
```

```
## Private Key:b5714fee43e2ba7930a7148b0d5f33b0f92e6bdd57098139f59b40a4f6694a26
##
## Public Key:045fc3c79da262553c1b390600399e7f7f1a2d32c4e610938218cbf756b125a519df937fbb36fa82d9182ca3e
##
## Bitcoin addresss:1J1FJ1qzEuDQT1L4qswanvRboJk66r9vtF
```

Multi signature bitcoin address

Multi signature address is associated with more than one private key They are used in organizations where authorization from few group of people is needed for single transaction

```python
import bitcoin
from bitcoin import *
#generate 3 private keys
private_key_1 = random_key()
private_key_2 = random_key()
private_key_3 = random_key()
```

```python
#generate 3 public keys
public_key_1 = privtopub(private_key_1)
public_key_2 = privtopub(private_key_2)
public_key_3 = privtopub(private_key_3)

#create multi signature address
multisig = mk_multisig_script(public_key_1,public_key_2,public_key_3,2,3)
multisig_address = scriptaddr(multisig)
print("Multisignature addresss:%s\n" %multisig_address)
```

```
## Multisignature addresss:3KFPzqAKtCCY9HoLbH73JssZqQS6jom2jK
```

Getting transactional history from a bitcoin address

```python
import bitcoin
from bitcoin import *
bitcoin_address = '171mn8VcaZWp9dHeGNcuwiDqWoxB21zq5'
print(history(bitcoin_address))
```

```
## [{'block_height': 490808, 'address': '171mn8VcaZWp9dHeGNcuwiDqWoxB21zq5', 'value': 44790000, 'output
```

Blockchain.info is a popular blockchain and bitcoin network and a wallet provider. We can view detailed information of each bitcoin transactions. We can also get bitcoin stats and also market data. We can get this data programatically using Blockchain API python library Get this library from 'pip install blockchain'

```python
import blockchain
from blockchain import *

#Get bitcoin exchange rates of latest 15 minutes
ticker = exchangerates.get_ticker()

print("Bitcoin Prices")
for i in ticker:
  print(i,ticker[i].p15min)
```

```
## Bitcoin Prices
## THB 322729.5
## JPY 1131088.58
## RUB 578660.95
## NZD 14314.66
## ISK 1027616.64
## CLP 6357566.51
## PLN 35074.44
## AUD 13033.74
## CNY 68302.79
## TWD 297311.65
## DKK 62109.86
## EUR 8349.56
## CHF 9749.86
## SGD 13330.21
## KRW 10718495.4
## CAD 12702.31
## HKD 77320.25
## GBP 7489.04
## INR 638355.67
## BRL 31817.93
## SEK 82664.94
## USD 9910.47
```

Get Bitcoin equivalent for particular amount of currency

```python
import blockchain
from blockchain import *

#Get bitcoin value for a particular amount
btc = exchangerates.to_btc('USD',1000)
print("\n1000 USD in Bitcoin: %s" %btc)
```

```
##
## 1000 USD in Bitcoin: 0.10091428
```

Get Bitcoin statistics

```python
import blockchain
from blockchain import *

stats = statistics.get()

#Bitcoin trade volume
print("\nBitcoin Trade Volume: %s" %stats.trade_volume_btc)

#Total bitcoins mined so far
print("\nTotal bitcoins mined so far: %s" %stats.btc_mined)

#Bitcoin market price
print("\nBitcoin market price in USD: %s" %stats.market_price_usd)
```

```
##
## Bitcoin Trade Volume: 82703.16
##
## Total bitcoins mined so far: 196250000000
##
## Bitcoin market price in USD: 9902.28
```

Block explorer method

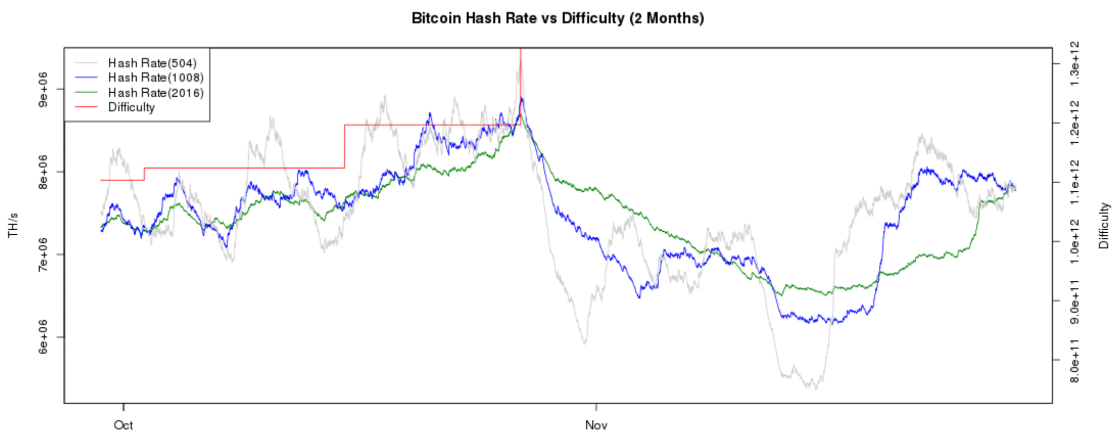
```python
import blockchain
from blockchain import blockexplorer

#Particular bitcoin block
block = blockexplorer.get_block("0000000000000000007c3160e3f097a33e51b0ebf3f60f03c732fdc1c5404c47")

print("\nBlock fee: %s" %block.fee)
print("\nBlock size: %s" %block.size)
print("\nBlock transactions: %s" %block.transactions)
```

```
##
## Block fee: 0
##
## Block size: 286
##
## Block transactions: [<blockchain.blockexplorer.Transaction object at 0x000002188173EE80>]
```

Mining Bitcoin

Process of adding bitcoin transaction data to bitcoin ledger of past transactions. Each bitcoin miner is competing with all the other miners in the network to correctly assemble the outstanding transactions into a block by solving specialized math problems. Miner are rewarded for all the transactions for evaluating transactions and solving math problems. The received fees attached for all the transactions that they succesufully evaluated are included in a block. In addition to tranasctoins fees, each miner are also rewarded for each block they mine. Specialized bitcoin mining hardware exists that can process transactions and build block quickly. Bitcoin.com provides bitcoin mining in cloud. 21.co is a specialized hardward to mine bitcoin. Bitcoin mining gets harder and difficult to solve day by day due to competition. Super computers are competing to mine next bitcoin and block

Bicoin hashrate vs difficulty



Earn bitcoin programmatically

Accept bitcoin on websites

Bitpay.com provides payments using bitcoin. This websites is integrated with several ecommerce websites, shoppint carts and also helps create bitcoin buttions to our websites by providing html code specific to merchants.

Building and Releasing bitcoin enabled APIs

Through 21.co, we can sell microservices for bitcoins. It currently supports Ubuntu and MacOS. First we need to install 21.co SDK on AWS Ubuntu instance, then login into 21.co account.

Building Bitcoin trading bot

We can trigger buy/sell alerts by keeping a threshold price for buy/sell. We will first get the coin price using Bitfinex API and trigger functions depending upon its value.

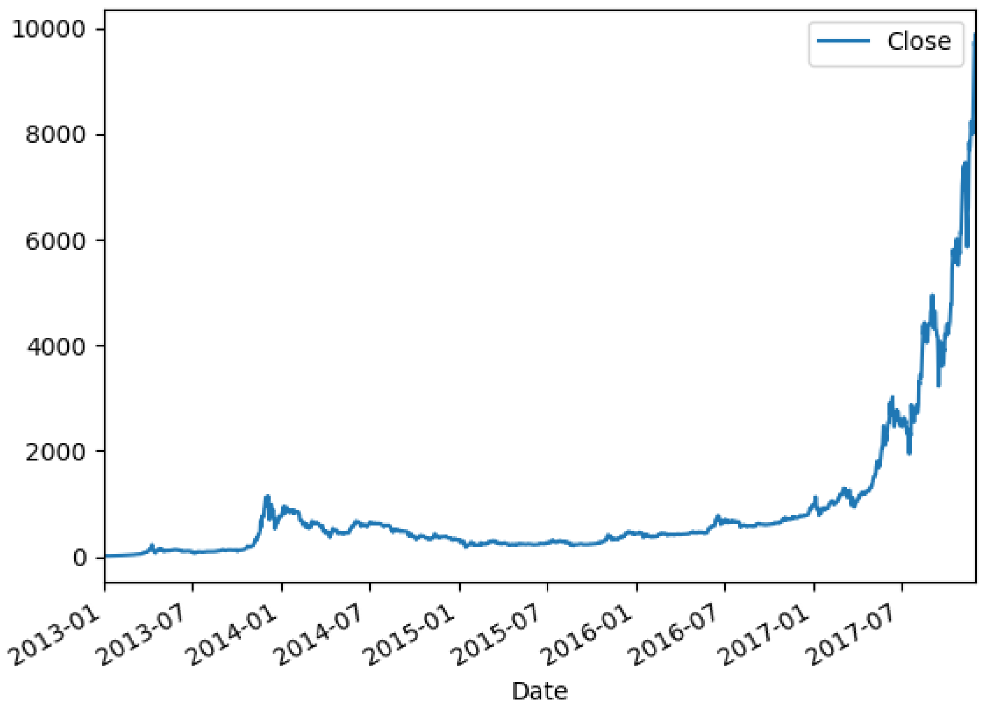
Bitcoin data anlaysis

Exploring, mainpulating and Visualizing bitcoin data

```
import pandas as pd
import matplotlib.pyplot as plt

price = pd.read_csv("coindesk-bpi-USD-ohlc_data-2012-12-31_2017-11-28.csv")
print("Bitcoin Price first six rows")
print(price.head())
print("\nBitcoin price info")
print(price.info())
print("\nBitcoin Price last six rows")
print(price.head())
price = price.dropna()
price['Date']=pd.to_datetime(price['Date'],format="%Y-%m-%d")
price.index =   price['Date']
del price['Date']
del price['Open']
del price['High']
del price['Low']
print("\nMinimum price of bitcoin")
print(price.min())

print("\nMaximum price of bitcoin")
print(price.max())

price.plot()

plt.savefig('bitcoin.png')
```

```
## tput: terminal attributes: No such device or address
##
```

```
## tput: terminal attributes: No such device or address
##
## tput: terminal attributes: No such device or address
##
## tput: terminal attributes: No such device or address
##
## Bitcoin Price first six rows
##                   Date   Open   High    Low  Close
## 0  2012-12-31 00:00:00  13.45  13.56  13.37  13.51
## 1  2013-01-01 00:00:00  13.51  13.56  13.16  13.30
## 2  2013-01-02 00:00:00  13.30  13.40  13.16  13.28
## 3  2013-01-03 00:00:00  13.28  13.46  13.25  13.40
## 4  2013-01-04 00:00:00  13.40  13.52  13.27  13.50
##
## Bitcoin price info
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 1819 entries, 0 to 1818
## Data columns (total 5 columns):
## Date     1819 non-null object
## Open     1817 non-null float64
## High     1817 non-null float64
## Low      1817 non-null float64
## Close    1817 non-null float64
## dtypes: float64(4), object(1)
## memory usage: 71.1+ KB
## None
##
## Bitcoin Price last six rows
##                   Date   Open   High    Low  Close
## 0  2012-12-31 00:00:00  13.45  13.56  13.37  13.51
## 1  2013-01-01 00:00:00  13.51  13.56  13.16  13.30
## 2  2013-01-02 00:00:00  13.30  13.40  13.16  13.28
## 3  2013-01-03 00:00:00  13.28  13.46  13.25  13.40
## 4  2013-01-04 00:00:00  13.40  13.52  13.27  13.50
##
## Minimum price of bitcoin
## Close    13.28
## dtype: float64
##
## Maximum price of bitcoin
## Close    9877.02
## dtype: float64
```



Build a blockchain and populate it

In blockchain, each block is stored with a timestamp or index. Each block have a self identifying hash to ensure integrity. Each block's hash will be a cryptographic hash of block's index, timestamp, data and the hash of previous block's hash. Data can be anything.

Each block requires information from previous block. The first block also called special block or genesis block, is added manually or has unique logic allowing it to be added.

```
import hashlib

#This will be our block structure
class Block:
  def __init__(self, index, timestamp, data, previous_hash):
    self.index = index
    self.timestamp = timestamp
    self.data = data
    self.previous_hash = previous_hash
    self.hash = self.hash_block()

  def hash_block(self):
    sha = hasher.sha256()
    sha.update(str(self.index) +
               str(self.timestamp) +
               str(self.data) +
                str(self.previous_hash))
    return sha.hexdigest()
```

```
#Manually construct genisis block having index zero, arbitary data and arbitary previous hash
import datetime as date

def create_genesis_block():
  return Block(0, date.datetime.now(), "Genesis Block", "0")
```

Now we will add blocks by taking previous block as parameter. The integrity of new block increases with each new block as block hash parameter comes from previous block

Once a block is added to blockchain, it cannot be removed or replaced.

```
def next_block(last_block):
  this_index = last_block.index + 1
  this_timestamp = date.datetime.now()
  this_data = "Hey! I'm block " + str(this_index)
  this_hash = last_block.hash
  return Block(this_index, this_timestamp, this_data, this_hash)
```

Now we will create blockchain, which is just a python list with first element as genesis block. We will add here 50 blocks to our blockchain.

```
# Create the blockchain and add the genesis block
import hashlib as hasher
import datetime as date
import random

#This will be our block structure
class Block:
  def __init__(self, index, timestamp, data, previous_hash):
    self.index = index
    self.timestamp = timestamp
    self.data = data
    self.previous_hash = previous_hash
    self.hash = self.hash_block()

  def hash_block(self):
    sha = hasher.sha256()
    sha.update((str(self.index) +
                str(self.timestamp) +
                str(self.data) +
                str(self.previous_hash)).encode("UTF-8"))
    return(sha.hexdigest())

def create_genesis_block():
  return(Block(0, date.datetime.now(), "Genesis Block", "0"))


def next_block(last_block):
```

```python
  this_index = last_block.index + 1
  this_timestamp = date.datetime.now()
  this_data = "Hey! I'm block " + str(this_index)
  this_hash = last_block.hash
  return(Block(this_index, this_timestamp, this_data, this_hash))

blockchain = [create_genesis_block()]
previous_block = blockchain[0]

num_of_blocks_to_add = 50

# Add blocks to the chain
for i in range(0, num_of_blocks_to_add):
  block_to_add = next_block(previous_block)
  blockchain.append(block_to_add)
  previous_block = block_to_add
  # Tell everyone about it!
  print ("Block #{} has been added to the blockchain!".format(block_to_add.index))
  print ("Hash: {}\n".format(block_to_add.hash))
```

```
## Block #1 has been added to the blockchain!
## Hash: d2cb19b1753afa216787e0cfce90031f48f27d39d6a702988625041eb9fb4261
##
## Block #2 has been added to the blockchain!
## Hash: 6d4c255198fcabb92e97984110d9858847f9df516ede4d56987e232a9e968b48
##
## Block #3 has been added to the blockchain!
## Hash: f227db423b1d995792afb563a9faedd6d9b042da1aadd969ec5e8e80626d313f
##
## Block #4 has been added to the blockchain!
## Hash: 64187fcbdffc94dc5315782a811363a1a4ce20584780fbbaacb1adb764e90554
##
## Block #5 has been added to the blockchain!
## Hash: fa04eda4c5a31a4268d3b3a88c602de96b27a7066531b231def565bf4c3bb522
##
## Block #6 has been added to the blockchain!
## Hash: 5c61b9682cef942b879cacfb37e185c20b8f66bc1665e4b04a0fd3c18962ac53
##
## Block #7 has been added to the blockchain!
## Hash: e967cc28d9d7e0bba617229b65cf777994f36f90890f0b4922c7e0964867cd60
##
## Block #8 has been added to the blockchain!
## Hash: fe755287deb0358173853c8549b769574bbfd44b253b8b74205036f109e39546
##
## Block #9 has been added to the blockchain!
## Hash: fb7b33be6b9db6d89850f7c103057681a89d22381f15e0601567a7817c32686a
##
## Block #10 has been added to the blockchain!
## Hash: 8384a1e8259acb19a86151b251810cf3b36dde41f471013517e460285a4a2fdf
##
## Block #11 has been added to the blockchain!
## Hash: db14ec86eba1a71392104d95a40e43d642538e27bc1a37d914e88e75d68a72a3
##
## Block #12 has been added to the blockchain!
## Hash: c5c428cd3781fc30fa13e95e2cf2d9d474cae0cbc2f4a3631a05cd21d6b13334
##
## Block #13 has been added to the blockchain!
## Hash: 72ebdb27de7e7682c7e266c5e17e99fe53dfae823d61baf8c505e490b40d8215
##
## Block #14 has been added to the blockchain!
## Hash: a9f41c57db0313e100c664c5e99080ca9c5a2cd393d70e72b0920abbd0fa81ff
##
## Block #15 has been added to the blockchain!
## Hash: 04ed94d4ce96f74ce81bdcc00a7fccf17100b2195113457103ae6281ea265f6f
##
## Block #16 has been added to the blockchain!
## Hash: f10e067cdd1301edd82cff001921f1afb7ad57eae5c62efd895e054eafad7a1f
##
## Block #17 has been added to the blockchain!
## Hash: f375f5b26be584a8560037ee6ab9a588c468107c3ee001555d50b873f073acbb
##
## Block #18 has been added to the blockchain!
## Hash: be5ef60bd9dc5a89b5502617a8e040c7fba783a2835c868be63614137bf118e6
##
## Block #19 has been added to the blockchain!
## Hash: 2294313bf5b47292be1b81985eeb3f0c31ec1ef526da449d2e09032e2493aebe
##
## Block #20 has been added to the blockchain!
## Hash: 24c1c1fb6291a4da7f5cdba3bbc9229e88b79f307c73452c7a4b610dbb2a81a0
##
## Block #21 has been added to the blockchain!
## Hash: e48183071555ece88277d09ed46a5cd0f9d66d1ea6b8148437ea316eba3867ed
##
## Block #22 has been added to the blockchain!
## Hash: 57b95420bbd664f9f4e0d583a9ba874ed78d4ab3b80df7a79603b7b997fd30f2
##
## Block #23 has been added to the blockchain!
## Hash: da3b3c92843f9fa9583b547e5abe9ae0013377146336714f81aacabc2a0ff585
##
## Block #24 has been added to the blockchain!
## Hash: 58db750dd0f97e37ae54a8be28608cecbd416ea0f8fed2ada3d50ca3aa6f76d6
##
## Block #25 has been added to the blockchain!
## Hash: c4ddd62e0c90ed60b694d7d3516e9fdfcbb11bc569b4f04e9a49ff102282c0f6
##
## Block #26 has been added to the blockchain!
## Hash: 333490370f0540b2a2a9b87531e4f7141182e749b98ccdb8642c563786e841d5
##
## Block #27 has been added to the blockchain!
## Hash: 724aae7f41f2b60669cb4b1298097cb0f0149784f8f73ec77ffb5d7e2add55de
##
## Block #28 has been added to the blockchain!
## Hash: d5f20364bf383bc904279bc28e3ab9929ec9c5b58c86b5699cee1a99e3edb24c
##
## Block #29 has been added to the blockchain!
## Hash: c2e14bb1be8125dc8e231247131ab7edd9219b245039c0dcab60fca6aa09df30
##
## Block #30 has been added to the blockchain!
## Hash: 1f8dd0804cc256b8fb4e69fee83b14f9112e2d4000e5d5e7aeb7d357deb3940c
##
## Block #31 has been added to the blockchain!
## Hash: 40f31520c61db943501d24ebb89a29e99373eb3b139bac6cc9de29da4507dfb1
##
## Block #32 has been added to the blockchain!
## Hash: c43555f508863a10225f42ea5ea91a646161337612e6e331867e283985c53ec0
##
## Block #33 has been added to the blockchain!
## Hash: 179ed347d2145f816a953fea2fa81f62a5c71d94d167c252dc9e40591eafbdba
##
## Block #34 has been added to the blockchain!
## Hash: 4719d1d9c8006f894a9804bce5b15440ae6c2e7dfbd64b04fda060b08cf1c6bf
##
## Block #35 has been added to the blockchain!
## Hash: f1420030d6cb7cc62385d1baee4e52d7947dc547ee2644c8afb7d656a7ad9455
##
## Block #36 has been added to the blockchain!
## Hash: 00ea5fa8519de837cbfd5cabb321378e85765dc1f4c708bed672723681f993b4
##
## Block #37 has been added to the blockchain!
## Hash: 21df90f34a30e95ba03bda22ed520d9c8c5581df28ba4384e059cf44947b406e
##
## Block #38 has been added to the blockchain!
## Hash: 1029f44fb592d1ea63e6922f64533168dd85f05fea044cdff2bf361a082e9fa1
##
## Block #39 has been added to the blockchain!
## Hash: 7e526dff49cda425462ce942b6a69d8a35e51f668330694b84dae6aaf20aaa09
##
## Block #40 has been added to the blockchain!
## Hash: 87cc6df08966f55f868c63db7f77e53a4842591284ce1ed0a4ac69c5bc8cf311
##
## Block #41 has been added to the blockchain!
## Hash: 2e8ff2bbc7c49a99a1a808452f2517a11f61c1acd8e884226baf4a6072210dba
##
## Block #42 has been added to the blockchain!
## Hash: 9005fd1c2147a370c16d659c0d11a7ac46e0f0574adc632510b76290c6d3fc2b
##
## Block #43 has been added to the blockchain!
## Hash: bf554cf7744ef327a0497e5eb0a190b5c3c5ad7e202d68f53e53bccdc789b39d
##
## Block #44 has been added to the blockchain!
## Hash: 27d3aaf16604397b2b9456c4441741ca9c4cfc381e6ef72a789cf4ebaa222e94
##
## Block #45 has been added to the blockchain!
## Hash: 5e0df89154e4ad394972b68cae9e4cc54a4772f7d9501b3ac0fd24b25b0fcbd6
##
## Block #46 has been added to the blockchain!
## Hash: cd272496d6db6b477824ddd21cfbcf2c3b592ebb687f085779fa7004d74336ce
##
## Block #47 has been added to the blockchain!
## Hash: 6b6e655fa344763ab5a4fac0b0af1abc09a7c3bc246cf0d02a6658b393f0a307
##
## Block #48 has been added to the blockchain!
## Hash: 2598133130545eb6095ff0d3686e7f4646da2af7c78c1f409efd688f0519330f
##
## Block #49 has been added to the blockchain!
## Hash: 66d32ced0184626cc1044e41d198666afd7b310832e9f301c152cce07558024c
##
## Block #50 has been added to the blockchain!
## Hash: 118ee7915ffef7048b7922f1cdff5f534fff9118e3ac0cc6cb91b3431bfbbfc6
```

References:

https://Udemy.com https://medium.com/crypto-currently