

---

SCHOOL OF COMPUTER ENGINEERING AND SCIENCE SHU  
BLOCKCHAIN TECHNOLOGY AND DATA ECONOMICS  
(08696017 1000)

---

## PRACTICAL REPORT

### PRACTISE 1

#### PROGRAMMING BITCOIN WITH PYTHON

Name: Lin Yijun      ID: 18120189

Date: 8 Apr. 2020

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>Materials and Methods</b>	<b>3</b>
2.1	Pi Bitcoin tools library . . . . .	3
2.2	How to generate private keys and public keys . . . . .	4
2.3	Creating a multi-signature bitcoin address . . . . .	4
<b>3</b>	<b>Results</b>	<b>5</b>
<b>4</b>	<b>Conclusion</b>	<b>7</b>
4.1	Functions Used . . . . .	7
4.2	Practise Conclusion . . . . .	8
<b>5</b>	<b>References</b>	<b>8</b>
<b>6</b>	<b>Appendix</b>	<b>8</b>

# 1 Introduction

In this practise, we are going to introduce the following topics:

- The Raspberry Pi Bitcoin tools library and how to start using it
- How to generate private keys and public keys
- How to create a simple bitcoin address from the private keys and public keys you generated

*To get started with bitcoin using Python, you must install Python 3.x and the bitcoin Python library called Pi Bitcoin tools in the system.*

## 2 Materials and Methods

### 2.1 Pi Bitcoin tools library

To install the Pi Bitcoin tools library, open the command-line program and execute the following command:

```
pip install bitcoin
```

*The best thing about this library is that it does not need to have a bitcoin node on your computer in order for you to start using it.*

This library connects to the bitcoin network and pulls data from places such as Blockchain.info. We shall start by writing the equivalent of a *Hello World* program for bitcoin in Python. In the *hello\_bitcoin.py* script, the demonstration of a new bitcoin address is created using Python. Go through the following steps to run the program:

#### 1. Import the bitcoin library:

```
1  #!/usr/bin/env python3
2  # -*- coding: utf-8 -*-
3  """
4  Title - Hello Bitcoin
5  This program demonstrates the creation of
6  - private key,
7  - public key,
8  - and a bitcoin address
9  Created on Wed Apr  8 10:30:32 2020
10 @author: reneelin
11 """
12
13 #import bitcoin
14 from bitcoin import *
```

#### 2. Generate a private key using the random key function:

```
#Generate Private Key
my_private_key = random_key()
```

#### 3. Display the private key on the screen:

```
print ("Private Key: %s\n" % my_private_key)
```

## 2.2 How to generate private keys and public keys

With the private key, a public key is generated. Perform this step by passing the private key that was generated to the *privtopub* function, as shown here:

```
1 #Generate Public Key
2 my_public_key = privtopub(my_private_key)
3 print ("Public Key: %s\n" % my_public_key)
```

Now, with the public key, generate a bitcoin address. Do this by passing the public key that is generated to the *pubtoaddr* function:

```
1 #Create a bitcoin address
2 my_bitcoin_address = pubtoaddr(my_public_key)
3 print ("Bitcoin Address: %s\n" % my_bitcoin_address)
```

## 2.3 Creating a multi-signature bitcoin address

A multi-signature address is an address that is associated with more than one private key; therefore, we need to create three private keys. Go through the following steps to create a multi-signature bitcoin address:

### 1. Create three private keys:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Title - Create multi-signature address
5 This program demonstrates the creation of
6 Multi-signature bitcoin address.
7 Created on Wed Apr 8 11:05:32 2020
8 @author: reneelin
9 """
10
11 #import bitcoin
12 from bitcoin import *
13 #Create Private Keys
14 my_private_key1 = random_key()
15 my_private_key2 = random_key()
16 my_private_key3 = random_key()
17 print ("Private Key1: %s\n" % my_private_key1)
18 print ("Private Key2: %s\n" % my_private_key2)
19 print ("Private Key3: %s\n" % my_private_key3)
20 print ('\n')
```

### 2. Create three public keys from those private keys using the *privtopub* function:

```
1 #Create Public Keys
2 my_public_key1 = privtopub(my_private_key1)
3 my_public_key2 = privtopub(my_private_key2)
4 my_public_key3 = privtopub(my_private_key3)
5 print ("Public Key1: %s\n" % my_public_key1)
6 print ("Public Key2: %s\n" % my_public_key2)
7 print ("Public Key3: %s\n" % my_public_key3)
8 print ('\n')
```

3. After generating the public keys, create the *multisig* by passing the three private keys to the *mk\_multi\_sig\_script* function. The resulting *multisig* is passed to the *addr* script function to create the *multisignature* bitcoin address.

```
1 #Create Multi-signature bitcoin address
2 my_multi_sig = mk_multisig_script(my_private_key1, my_private_key2,
    my_private_key3, 2,3)
3 my_multi_address = scriptaddr(my_multi_sig)
4 print ("Multi signature address: %s" % my_multi_address)
```

4. Print the *multisignature* address and execute the script.

You can also look at the preexisting bitcoin addresses' transactional history. We will first get a valid address from Blockchain.info. The following screenshot shows the copied address of a bitcoin block:

**BTC / Address**

Addresses are identifiers which you use to send Bitcoin to another person

Address	12dRugNcdxK39288NjcDV4GX7rMsKCGn6B
Format	BASE58 (P2PKH)
Transactions	2,544
Total Received	31028.94015641 BTC
Total Sent	30136.68307843 BTC
Final Balance	892.25707798 BTC

Payment Request    Donation Button

Pass the copied address to the *history* function, as shown in the following code:

```
1 #!/usr/bin/env python3
2 # -*- coding: utf-8 -*-
3 """
4 Title - Bitcoin Transaction History
5 This program demonstrates listing history of a bitcoin address.
6 Created on Wed Apr 8 11:32:00 2020
7 @author: reneelin
8 """
9
10 #import bitcoin
11 from bitcoin import *
12 #View address transaction history
13 a_valid_bitcoin_address = '12dRugNcdxK39288NjcDV4GX7rMsKCGn6B'
14 print (history(a_valid_bitcoin_address))
```

## 3 Results

Mac OS set python 2.x before it left factory. So the practical paper's command could not be used due to the conflict.

```
pip install bitcoin
```

Instead, user needs to use the following command to announce the version of Python.

```
pip3 install bitcoin
```

The following screenshot shows the installation of Pi Bitcoin tools library in Terminal(Mac OS):

```
reneeelin -- zsh -- 80x25
Last login: Wed Apr  8 01:58:06 on ttys000
reneeelin@Yijuns-MacBook-Pro ~ % pip3 install bitcoin
Collecting bitcoin
  Downloading bitcoin-1.1.42.tar.gz (36 kB)
Building wheels for collected packages: bitcoin
  Building wheel for bitcoin (setup.py) ... done
  Created wheel for bitcoin: filename=bitcoin-1.1.42-py3-none-any.whl size=44411
  sha256=f8d25b1dd59719e75b05395af51e725a12a0af926454ecc8409347f16124ae55
  Stored in directory: /Users/reneeelin/Library/Caches/pip/wheels/6c/66/7d/2b4114
  252aed67e75a79a1235ce8b2701da915acac6fd7061
Successfully built bitcoin
Installing collected packages: bitcoin
Successfully installed bitcoin-1.1.42
reneeelin@Yijuns-MacBook-Pro ~ %
```

Some third-party collections may have problem when using Spyder to import, so user needs to install Pi Bitcoin tools library in Spyder's console using the following command:

```
!pip install bitcoin
```

The following screenshot shows the installation of Pi Bitcoin tools library in Spyder, using Anaconda 5:

```
IPython console
Console 1/A
In [6]: !pip install bitcoin
Collecting bitcoin
  Using cached https://files.pythonhosted.org/packages/12/88/
  c93906385b2713d38ccea46c93e8ec804f052a15a94f9b1d4c66baabd24/
  bitcoin-1.1.42.tar.gz
Building wheels for collected packages: bitcoin
  Building wheel for bitcoin (setup.py) ... [725]done
  reated wheel for bitcoin: filename=bitcoin-1.1.42-cp37-none-any.whl size=44412
  sha256=39bb5485ab06cc4e3695285a7077634bb0b86ec5470ed7a0fa42bad23ac8af5b
  Stored in directory: /Users/reneeelin/Library/Caches/pip/wheels/1c/22/e0/
  a6d5c7103f57cbdb0a10303f537f4670f65a2fce81d9068230
Successfully built bitcoin
Installing collected packages: bitcoin
Successfully installed bitcoin-1.1.42
```

Using the code written in 'Materials and Methods', the running results complete the practise's aim. The following screenshot shows the private key, public key and bitcoin address that is generated:

```
IPython console
Console 1/A
Python 3.7.0 (default, Jun 28 2018, 07:39:16)
Type "copyright", "credits" or "license()" for more information.

IPython 7.8.0 -- An enhanced Interactive Python.

In [1]: runfile('/Users/reneeelin/Desktop/Student/[19:S]/区块链技术与数据经济学/
  Practical Section/Week1/Hello_Bitcoin.py', wdir='/Users/reneeelin/Desktop/
  Student/[19:S]/区块链技术与数据经济学/Practical Section/Week1')
Private Key: 8c7c5fe0755c4a723b762926a99af80d9e78d4911cf9e6ed9d09cf2784d7d643
Public Key:
0434fed45f4bd30b32ca0b4cd4b8cd7d906be2ca6f16af0c48114b1f7b4d211b7908bfc323da8b7
2c2296a9e7c49dea2edc209478306c7ee75b3d1766c16c18e1
Bitcoin Address: 1JRAgqbYfXaY96UnyCKKPGJCtkbWZY9gm
```

The following screenshot shows the output for the multisig bitcoin address:

```
IPython console
Console 1/A
In [2]: runfile('/Users/reneeelin/Desktop/Student/[19:S]/区块链技术与数据经济学/
  Practical Section/Week1/Create_multi-signature_address.py', wdir='/Users/
  reneeelin/Desktop/Student/[19:S]/区块链技术与数据经济学/Practical Section/Week1')
Private Key1: 08b9eb3ca172ab5a724a7fb7609068b4e8686b0560908e95b35743ec894f5c1
Private Key2: a50ba1d2bbf23f8ed67b30afce0ad82094b1d018426e8de2fa00eab935d2e541
Private Key3: 7cfb433d15d1adb11145e9e2b80294196099492d40f9d3eab747747ad0fb7ca6

Public Key1:
04aa06ed81edac66d731b644ee363818372d0f03ba85e9a5e57e205e23d3a92b19aa80756072d7a7
faa20bd02c1cfaa22b8b45e1a9926b25b83543854129b4dd
Public Key2:
0426eb32a4b3aa2f92c7dec7839ae3c5dde4925998e8f19ce3bf0d4c0b24e58d64a023f7bd252d9a
b7eeee038406c1c57b58c0a2bdfbe534034c2708ae8e2ab7ab
Public Key3:
040c838259b70ed41040497bb727acf2da632525f29725723c3d687a5c2799639f8954ffccfda5b3
35f2d1d588f1f1f9f984fcccfa8104c4a07f8dc2bb25e1dd4

Multi signature address: 3DHCBPX4ADrs7QcEVjDR4MlFtKaxkveDRh
```

The following screenshot shows the output to get the history of the bitcoin address, including the transactional information:

```

IPython console
Console 1/A
Fetching more transactions... 1050
Fetching more transactions... 1100
Fetching more transactions... 1150
Fetching more transactions... 1200
Fetching more transactions... 1250
Fetching more transactions... 1300
Fetching more transactions... 1350
Fetching more transactions... 1400
Fetching more transactions... 1450
Fetching more transactions... 1500
Fetching more transactions... 1550
Fetching more transactions... 1600
Fetching more transactions... 1650
Fetching more transactions... 1700
Fetching more transactions... 1750
Fetching more transactions... 1800
Fetching more transactions... 1850
Fetching more transactions... 1900
Fetching more transactions... 1950
Fetching more transactions... 2000
Fetching more transactions... 2050
Fetching more transactions... 2100
Fetching more transactions... 2150
Fetching more transactions... 2200
Fetching more transactions... 2250
Fetching more transactions... 2300
Fetching more transactions... 2350
Fetching more transactions... 2400
Fetching more transactions... 2450
Fetching more transactions... 2500
[{'address': '12dRugNcdxK39288NjcDV4GX7rMsKCGn6B', 'value': 1319630805,
'output': 'deed9d54c4e0b477c48669a924e1d142179598d24772924be1c5c95352a9198e:0',
'block_height': 602400, 'spend':
'6dad492a25646e4168ba71058d88ccd1ce272539f48096515a8b3b3be052d554:0'},
{'address': '12dRugNcdxK39288NjcDV4GX7rMsKCGn6B', 'value': 299416444, 'output':
'56f26c44062414564077fe1cb508c9ba11051965ef53a9153c62d4527a24b207:2',
'block_height': 605413, 'spend':
'b51f4f831c672452244707b423cb3503bd86d25026fb45530db5ba0bde6b670e:8'},
{'address': '12dRugNcdxK39288NjcDV4GX7rMsKCGn6B', 'value': 250552735, 'output':
'338befb02dea68b10adb20d709344a6012a2b5cd6cf409911a38c5781e322cb0:4',
'block_height': 603388, 'spend':
'809b92632aaa950285795e513814942d779c732154bc65d99aa3d1aa57116240:10'},
{'address': '12dRugNcdxK39288NjcDV4GX7rMsKCGn6B', 'value': 433168353, 'output':
'b8391128488ba07a7e482f30bf9c8854a399c443cf3e1be46fcbabbb466fff88:5',
'block_height': 603237, 'spend':
'22b4a00850d5c0496481a18208a6141b4d1ab1c9f16e88c911d726c5d3880195:27'},
{'address': '12dRugNcdxK39288NjcDV4GX7rMsKCGn6B', 'value': 620086249, 'output':
'c8dafc428e12c1243692cf4666e62180045b8aadeb474ee7cace51fe145b70a8:3',
'block_height': 602923, 'spend':
'338befb02dea68b10adb20d709344a6012a2b5cd6cf409911a38c5781e322cb0:26'},

```

## 4 Conclusion

### 4.1 Functions Used

**random\_key:** () -> **privkey** used for generating a random 32 bytes private key.

**privtopub:** (privkey) -> **pubkey** used for converting a private key to a public key.

**pubtoaddr:** (pubkey) -> **address** used for converting a public key to an address.

**mk\_multi-sig\_script:** (pubkeys, k, n) -> **k-of-n multisig script from pubkeys** For example, n=3 for three public keys, and k = 2 to choose two from these three. But in this practise, it uses private keys to generate the multi-sig.

**history:** (address1, address2, etc) -> **outputs to those addresses** used for loading transactions of the target address.

## 4.2 Practise Conclusion

In this practise, I have learnt to install The Raspberry Pi Bitcoin tools library with both terminal and Spyder(Anaconda). Meanwhile, I learnt how to start using it.

Using the functions in the library, I successfully generate private keys and converting them to public keys.

Then I tried to create a simple bitcoin address from the private keys and public keys I generated. Mutisignature address was also successfully generated.

At the end of the practise, I search for an address on 'Blockchain.com' and pick an address that belongs to Antpool, using command to view its transactions.

In this practise, I get familiar with bitcoin library. Because I'm a new learner in both Python and Blockchain Technology & Data Economics, I plan to learn more features after class. Hope the next 9 weeks!

## 5 References

PDF document: Practical 1 Guide File.

Website: [Pypi.org/project/bitcoin.](https://pypi.org/project/bitcoin/)

Website: Bitcoin Wiki.

PDF document: How to Write a Practical/Laboratory Report—Learning Guide.

## 6 Appendix

### Bitcoin

Bitcoin is a peer-to-peer currency. Peer-to-peer means that no central authority issues new money or tracks transactions. These tasks are managed collectively by the network.

### Private key

A private key in the context of Bitcoin is a secret number that allows bitcoins to be spent. Every Bitcoin wallet contains one or more private keys, which are saved in the wallet file. The private keys are mathematically related to all Bitcoin addresses generated for the wallet.

### Public key

A public key is a cryptographic code that allows a user to receive cryptocurrencies into his or her account. The public key coupled with the private key are significant tools required to ensure the security of the crypto economy.

### Bitcoin address

A bitcoin address is a single-use token. Just as people use email addresses to send and receive emails, you can use this bitcoin address to send and receive bitcoins. Unlike email addresses, however, people have many different bitcoin addresses, and a unique address should be used for each transaction.



# Address Map

