به نام خدا

تمرین دوم عملی بلاکچین

محمد مهدی صباغی(۹۵۱۰۹۱۲۳)

سوال اول:

در این سوال همان فرم استاندارد P2PKH را پیاده کردیم به آدرس یک نفر پول میریزد. ScriptPubkey باید چک کند که آیا Pubkey مربوط به همان آدرس است و امضای آن را بررسی کند. در ScriptSig هم کافی است که Sig, Pubkey را بگذاریم.

```
return [
    OP_DUP, OP_HASH160, address, OP_EQUALVERIFY, OP_CHECKSIG
]
```

```
return [
    signature, public_key
]
```

سوال دوم:

شماره دانشجویی من 95109123 است و برای این که x, y زوج باشند 9124 , 9510را در نظر گرفتم. در ScriptSig دو عدد x,y حل معادله را داده و در ScriptPubkey چک میکنیم که جمع و تفریق آن ها درست باشد. برای دوبار چک کردن از OP_2DUP استفاده کردیم که دو عدد آخر را تکرار میکند.

```
Q2a_txout_scriptPubKey = [
        OP_2DUP, OP_ADD, 9510, OP_EQUALVERIFY, OP_SUB, 9124, OP_EQUALVERIFY
    ]
```

```
txin_scriptSig = [
    True, 9317, 193
]
```

سوال سوم:

۱. در قسمت اول برای بفهمد از کدام پروتکل میخواهند استفاده کنند از OP_Depth استفاده کردیم تا تعداد امضاها را بررسی کند(چون Multisig یک متغیر اضافه میخواهد باید تعداد عملیات ها یکی بیشتر از امضاها). سپس برای فراز و عطا از یک Multisig استفاده کرده و برای یکی از آنها و بقیه سهامداران از دو Multisig پشت سرهم استفاده کردم، تنها باید حواسمان باشد که در ScriptSig امضای عطا یا فراز باید آخر باشد که اول Multisig آن ها بررسی میشود. نهایتا در Multisig امضاها را علاوه بر dummy variables فرستادیم.

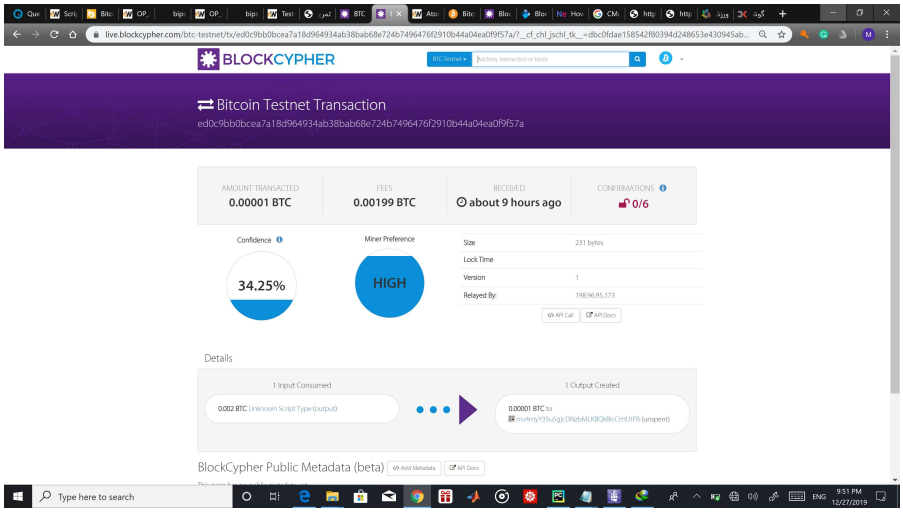
```
Q31a_txout_scriptPubKey = [
    OP_DEPTH, 3, OP_EQUAL, OP_IF, 2, Ata_Pubkey, Faraz_Pubkey, 2, OP_CHECKMULTISIG, OP_ELSE, 1, Ata_Pubkey, Faraz_Pubkey,
    2, OP_CHECKMULTISIGVERIFY, 3, share1_Pubkey, share2_Pubkey, share3_Pubkey, share4_Pubkey, share5_Pubkey, 5,
    OP_CHECKMULTISIG, OP_ENDIF
    ]
```

```
txin_scriptSig = [
    1, Ata_sig, Faraz_sig
]
```

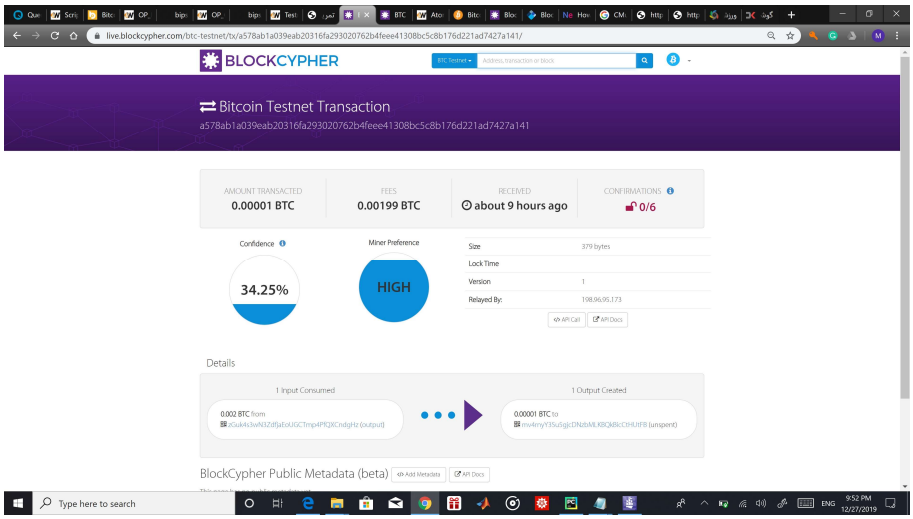
۲. در اینجا همان کد بالا بود صرفا OP_if را برداشتیم چون دیگر عطا و فراز باهم نبودند:

```
Q31b_txout_scriptPubKey = [
    1, Ata_Pubkey, Faraz_Pubkey,
    2, OP_CHECKMULTISIGVERIFY, 3, share1_Pubkey, share2_Pubkey, share3_Pubkey, share4_Pubkey, share5_Pubkey, 5,
    OP_CHECKMULTISIG
    ]
```

فقط یک نکته این که هر چقدر پاداش را بالا بردیم کسی Redeemها را confirmنکرد و من عکس آن ها گرفته ام:




Q31b




Q32b

سوال چهارم:

در قسمت اول کافی بود که از یک تایم لاک استفاده کنیم. زمان را به ثانیه در سیستم یونیکس به دست آورد و هنگامی که گذشت Drop می شود:

```
Q4a_txout_scriptPubKey = [
    1577274600, OP_CHECKLOCKTIMEVERIFY, OP_DROP, OP_DUP, OP_HASH160, address4, OP_EQUALVERIFY, OP_CHECKSIG
    ]
```

در قسمت دوم هم از OP_Return استفاده شد که هیچ وقت نمیتوان آن را Redeem کرد ومیتوان بعد از آن تا ۸۰ بایت پیغام گذاشت.

```
Q4_txout_scriptPubKey = [
        OP_RETURN, "Happy Birthday Hamed".encode('UTF-8')
    ]
```

سوال پنجم:

در این سوال همانطور که گفته شده بود دو ایده را اجرا کردیم. اولی اینکه یک آدرس از رویHash ساختیم. از آنجا Hash فایل پابلیک مجاز نبود. ابتدا Hash160 آن را حساب کرده و از روی آن ساختیم. یک ایده دیگر اینکه مانند قسمت دوم سوال چهار که پیغام مفرستاد اینجاهم از OP-return استفاده کردیم.

اما در قسمت بعد Hash همه فایل ها را باهم گرفته و از روی آن مانند قبلی آدرس را ساختیم.

سوال ششم:

برای ساختن ScriptPubkey باید هر دو حالت که تراکنش انجام شود یا نشود را در نظر میگرفتیم و سپس با استفاده از Op_if آن را میساختیم. در حالت اول کافی بود که در ابندا بررسی کند که PreImage درست است و در حالت دوم کافی بود باید امضای هردو میبود و از Multisig استفاده شد:

```
return [
    OP_DEPTH, 2, OP_EQUAL, OP_IF, OP_HASH160, hash_of_secret, OP_EQUALVERIFY, public_key_recipient, OP_CHECKSIG, OP_ELSE, 2, public_key_sender,
    public_key_recipient, 2, OP_CHECKMULTISIG, OP_ENDIF
]
```

برای Scriptsig هم در حالت اول امضا و Preimage را باید میفرستادیم و در حالت دوم دو امضا به علاوه متغیر اضافی:

```
return [
    sig_recipient, secret
]
```

```
return [
    1, sig_sender, sig_recipient
]
```