《区块链基础及应用》实验报告

Ex1

学院：网络空间安全学院　专业：信息安全　姓名：陆皓喆　学号：2211044

一、项目构成

github地址：https://github.com/Luhaozhhhe/Blockchain_Fundamentals_and_Applications

Ex1

- codes
  - **config.py：修改了私钥**
  - **ex1.py：完成了发币的操作**
  - keygen.py：生成私钥和地址
  - requirements.txt：环境要求
  - **split_test_coins .py：完成了分币的操作**
  - utils.py：没有做修改
- documents
  - Ex1：解压后的文件夹
  - Ex1.rar：老师下发的压缩包
- reports
  - **BlockChain-Ex1-2211044-陆皓喆.pdf：撰写的实验报告**
  - **output_by_ex1.txt：发币步骤输出的文本**
  - **split_output.txt：分币步骤输出的文本**
  - **分币操作.png：运行split_test_coins .py脚本的结果**
  - **账户信息.png：领取完bitcoin后的账户信息**
  - **发币操作.png：运行ex1.py脚本的结果**
  - **bitcoin领取截图.png：预备实验中的领取记录**

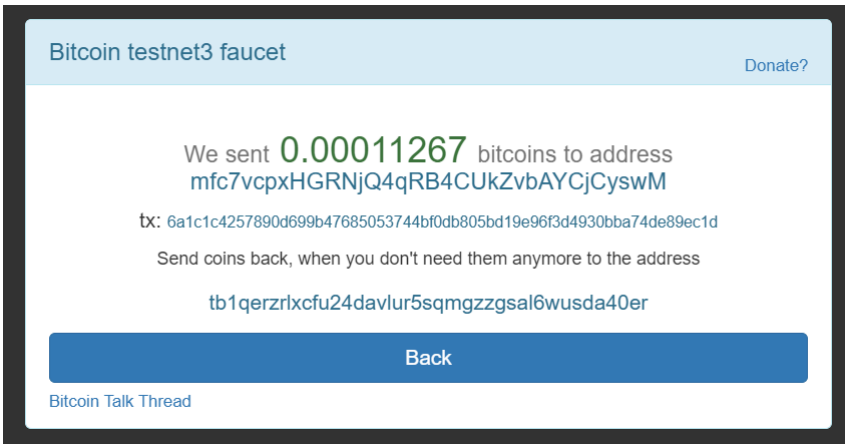
二、实验内容

步骤1：领取bitcoins

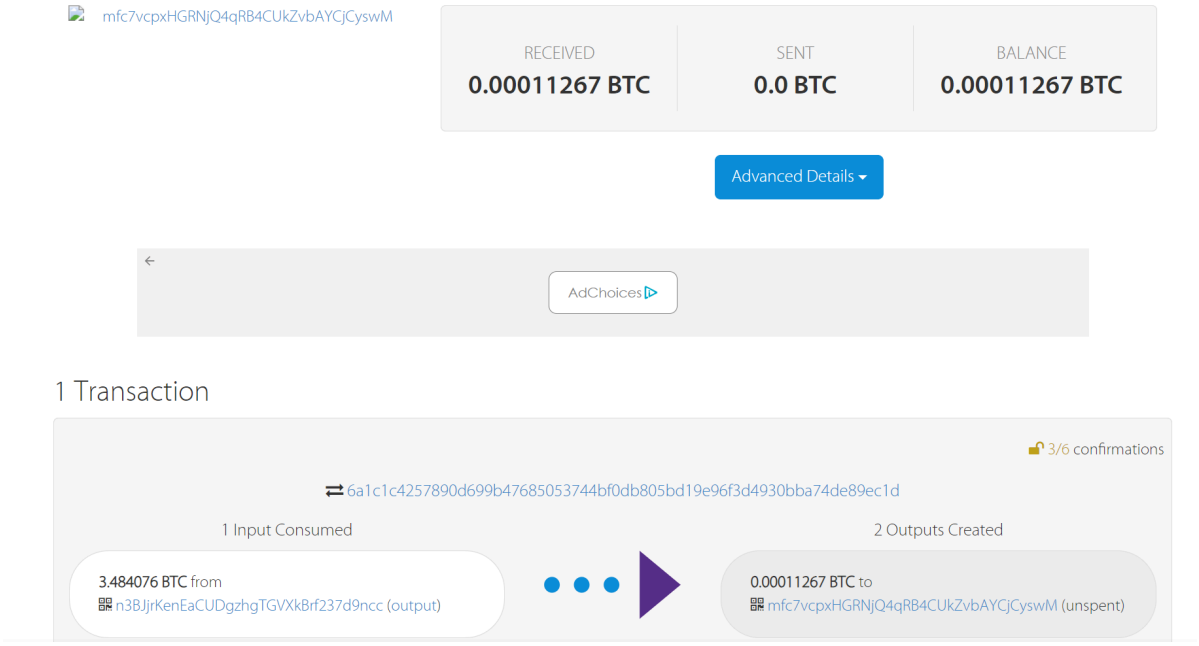
首先，我们根据预备实验，在网址上领取了我们的bitcoins。

bitcoins的相关信息如下所示：

- **tx:**6a1c1c4257890d699b47685053744bf0db805bd19e96f3d4930bba74de89ec1d
- **address:**mfc7vcpxHGRNjQ4qRB4CUkZvbAYCjCyswM
- **private key:**cW3fxHAhp5RRfDcFkbctRPymyYJZe3AEF7UngN8qfnKhkPnccKNj



目前我们账户下所拥有的bitcoins如下图所示：



步骤2：分块bitcoins

在领取完bitcoins之后，我们通过编写程序来完成对bitcoins的分块。

在分块之前，首先我们要填写一些相关的地址信息。我们打开`config.py`文件，将私钥替换成我们自己的私钥

```
#将预备实验中的创建的私钥(private key)写入
my_private_key = CBitcoinSecret(
    'cVqivA5FLCHRkvvNpWCZ6XvaoWSnf2jHLCUQLad6dtRMBcuNfrQw')
```

然后我们打开`split_test_coins .py`文件，开始正式的分币工作。`split_coins`函数已经帮我们写好，因此我们只需要完成main函数的部分即可。

我们给出填写的值的含义：

- **amount_to_send**：就是我们一共需要花费的bitcoins的数量，由于我们只领取到了0.00011个bitcoins，所以在此处我们选择花费0.0001个bitcoins
- **txid_to_spend**：就是我们之前领取bitcoins时的tx值
- **utxo_index**：由于之前我们都没有进行过bitcoins的交易，所以交易对应的索引为0

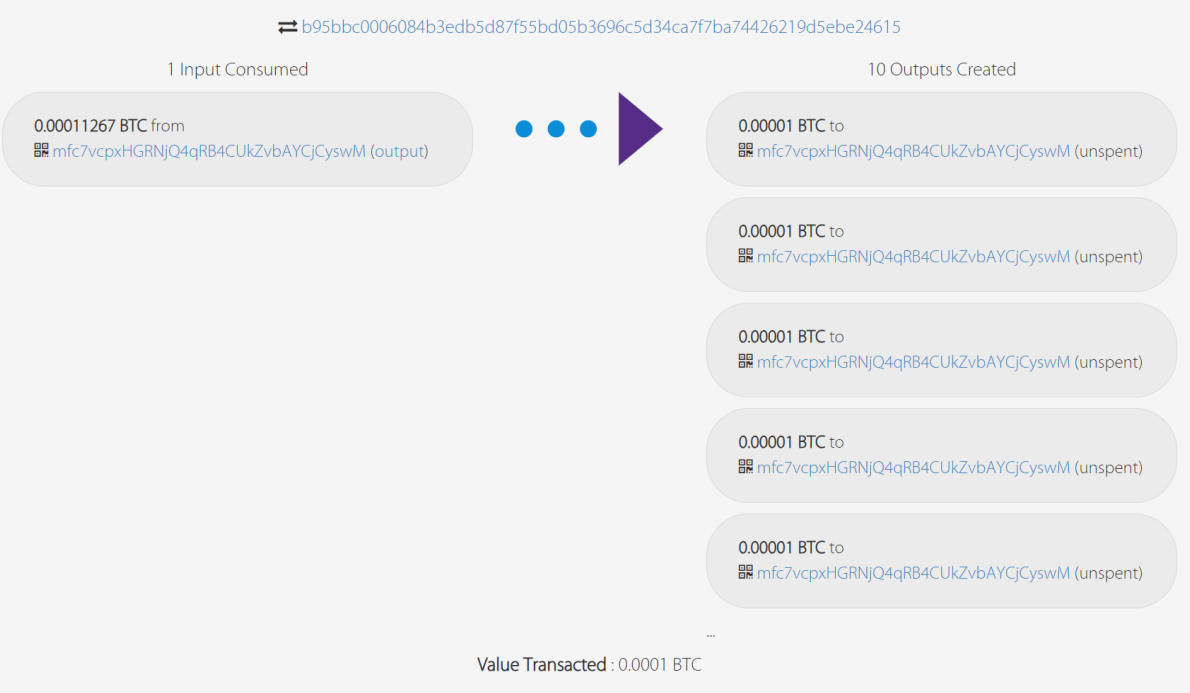
- **n**：就是我们规定的分块的数量，按照课上所说，我们要将其拆分成10份，因此n=10

具体代码如下所示：

```
if __name__ == '__main__':

    amount_to_send = 0.0001 # 由于我们一共只有0.00011个bitcoins，因此我们只需要拆分
0.0001个bitcoins
    txid_to_spend = (
        '6a1c1c4257890d699b47685053744bf0db805bd19e96f3d4930bba74de89ec1d')
    utxo_index = 0
    n=10 # 根据课上所讲，拆成10份

    split_coins(amount_to_send, txid_to_spend, utxo_index, n)
```

运行程序，就会输出我们的分块结果，由于输出内容过长，此处不再详细展示，我将其放到了**reports文件夹下的split_output.txt中**

下面是faucet上的截图，可以说明我们已经成功完成了对bitcoin的分块操作。



步骤3：发放bitcoins

分完bitcoin之后，我们就要进行对应的发放工作，此处我们主要填写ex1.py文件中的代码。

我们需要补全的有：

- P2PKH_scriptPubKey函数
- P2PKH_scriptSig函数
- main函数

P2PKH_scriptPubKey函数

我们上网查询可知，该函数用于生成一个标准的交易输出脚本，根据课本第三章的内容可以知道，该部分由以下部分构成：

> - **OP_DUP**：复制堆栈顶端数据
> - **OP_HASH160**：计算hash函数两次，第一次用SHA-256，第二次用RIPEMD-160
> - **temp_hash_value**：前面计算出来的公钥hash值
> - **OP_EQUALVERIFY**：检查栈顶两个元素是否相等，是一个bool值
> - **OP_CHECKSIG**：检查栈顶元素是否是有效签名

我们根据以上信息，编写出对应的函数，如下所示：

```
def P2PKH_scriptPubKey(address):

    script_address=address.to_scriptPubKey()   #script_address基于给定的bitcoin地址生成了一个脚本

    temp_hash_value=script_address[3:-2]   #提取出公钥哈希的值

    Script_PubKey = [
        OP_DUP,                     # 复制堆栈顶端数据
        OP_HASH160,                 # 计算hash函数两次，第一次用SHA-256，第二次用RIPEMD-160
        temp_hash_value,            # 前面计算出来的公钥hash值
        OP_EQUALVERIFY,             # 检查栈顶两个元素是否相等，是一个bool值
        OP_CHECKSIG                 # 检查栈顶元素是否是有效签名
    ]

    return Script_PubKey
```

可以看到，函数首先输入一个地址，然后利用to_scriptPubKey函数将其转化为脚本地址，接着提取出公钥hash的值。我们输出的Script_PubKey主要由五部分构成：堆栈顶端的数据、计算出的hash函数、前面算出的公钥hash值、比较栈顶元素是否相等、检查栈顶是否是有效签名。

P2PKH_scriptSig函数

P2PKH_scriptSig(txin, txout, txin_scriptPubKey)函数主要用于生成一个有效脚本用来解锁输出并发送回faucet。

该函数的参数含义如下所示：

> - **txin**：表示输入的交易数据
> - **txout**：表示输出的交易数据
> - **txin_scriptPubKey**：表示输入交易的脚本公钥

该函数需要完成的任务有：**验证输入txin和输出txout的有效性，确保交易数据是有效的；解析txin_scriptPubKey，提取出公钥hash；使用private key对输入txin进行签名，生成一个脚本签名；最后，将脚本签名和公钥作为输入的脚本签名（scriptSig）返回。**

返回部分主要由脚本签名和公钥构成，也就是说我们需要return两个变量：**signature**和**Public_Key**。

我们根据以上信息，编写出对应的函数，如下所示：

```
def P2PKH_scriptSig(txin, txout, txin_scriptPubKey):
    #创建新的签名
    signature = create_OP_CHECKSIG_signature(txin, txout, txin_scriptPubKey,
                                             my_private_key)
    Public_Key=my_public_key    #获取我们的公钥

    return [signature,Public_Key]    #返回我们的新的签名和公钥
```

main函数

main函数主要完成对一些参数的配置工作，我们只需要按照个人的信息进行填写即可。

部分参数含义如下所示：

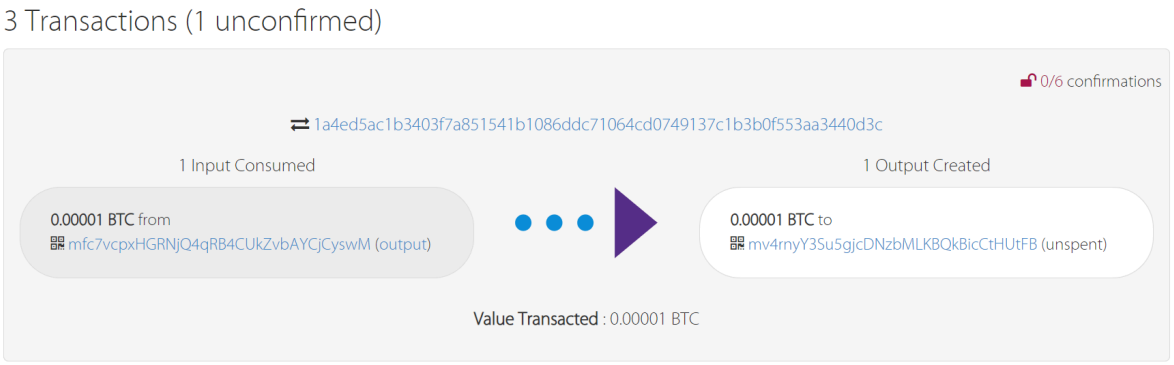
> - **amount_to_send**：交易的费用
> - **txid_to_spend**：之前交易的hash id
> - **utxo_index**：索引值，还是0，因为没有产生过交易

代码如下所示：

```
if __name__ == '__main__':

    #交易的费用
    amount_to_send = 0.00001

    #交易的hash id
    txid_to_spend = (
        'b95bbc0006084b3edb5d87f55bd05b3696c5d34ca7f7ba74426219d5ebe24615')

    #utxo索引，依旧是0，因为没有产生过交易
    utxo_index = 0

    txout_scriptPubKey = P2PKH_scriptPubKey(faucet_address)
    response = send_from_P2PKH_transaction(
        amount_to_send, txid_to_spend, utxo_index, txout_scriptPubKey)
    print(response.status_code, response.reason)
    print(response.text)
```

运行程序后，我们发现已经完成了交易，输出的部分我们存放在**reports中的output_by_ex1.txt中**

faucet截图如下所示：



可以看到，我们成功完成了对分币后的某一笔交易的发币操作，并将其发给了老师的bitcoin账户上！