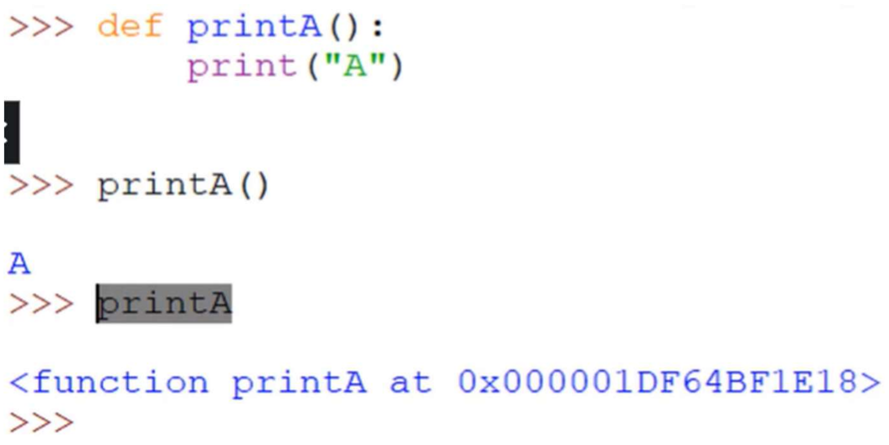
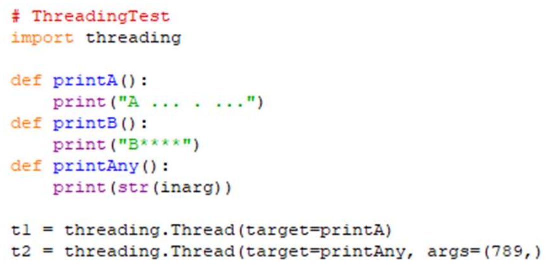
Vid 57 – Basic Threading



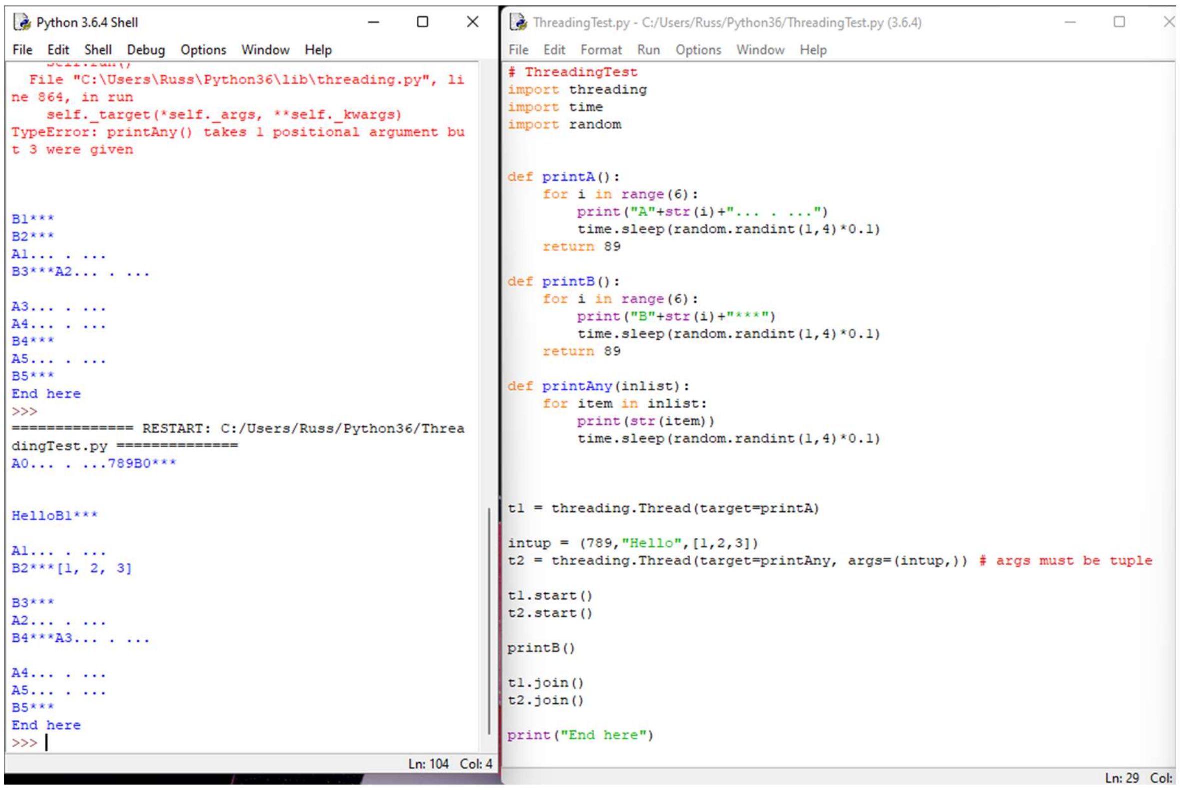
```
>>> def printA():
        print("A")

>>> printA()

A
>>> printA

<function printA at 0x000001DF64BF1E18>
>>>
```



```
# ThreadingTest
import threading

def printA():
    print("A ... . ...")
def printB():
    print("B****")
def printAny():
    print(str(inarg))

t1 = threading.Thread(target=printA)
t2 = threading.Thread(target=printAny, args=(789,)
```

T1 thread prepared when invoked to launch printA

T2 thread prepared to launch printAny w/ the argument 789

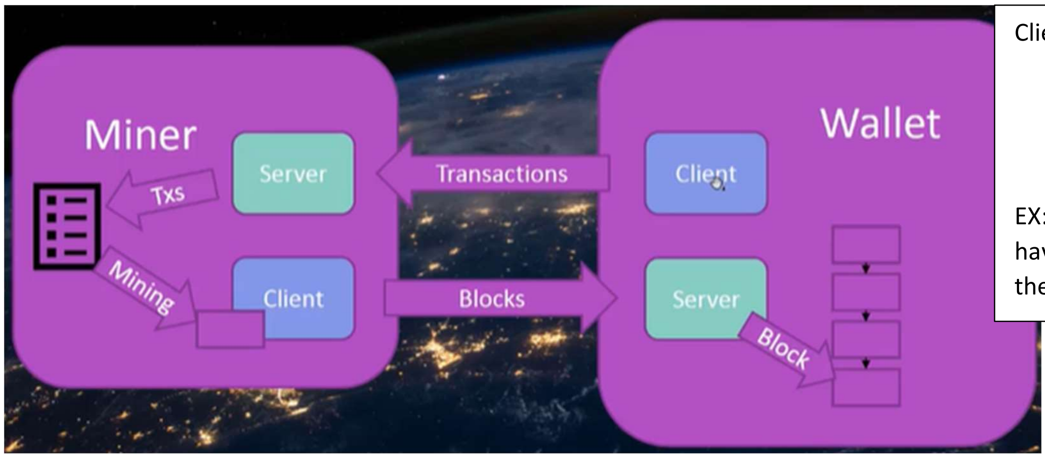


```
  File "C:\Users\Russ\Python36\lib\threading.py", li
ne 864, in run
    self._target(*self._args, **self._kwargs)
TypeError: printAny() takes 1 positional argument bu
t 3 were given



B1***
B2***
A1... . ...
B3***A2... . ...

A3... . ...
A4... . ...
B4***
A5... . ...
B5***
End here
>>>
============== RESTART: C:/Users/Russ/Python36/Threa
dingTest.py ==============
A0... . ...789B0***


HelloB1***

A1... . ...
B2***[1, 2, 3]

B3***
A2... . ...
B4***A3... . ...

A4... . ...
A5... . ...
B5***
End here
>>>
```

```
# ThreadingTest
import threading
import time
import random


def printA():
    for i in range(6):
        print("A"+str(i)+"... . ...")
        time.sleep(random.randint(1,4)*0.1)
    return 89

def printB():
    for i in range(6):
        print("B"+str(i)+"***")
        time.sleep(random.randint(1,4)*0.1)
    return 89

def printAny(inlist):
    for item in inlist:
        print(str(item))
        time.sleep(random.randint(1,4)*0.1)



t1 = threading.Thread(target=printA)

intup = (789,"Hello",[1,2,3])
t2 = threading.Thread(target=printAny, args=(intup,)) # args must be tuple

t1.start()
t2.start()

printB()

t1.join()
t2.join()

print("End here")
```

Vid 59 – Assignment 1: Test Miner Using Threads

- We want to create a wallet that contains a client that creates transactions at the request of probably a user
- Those transactions get received by a server which forms a list of transactions
- Then those are built into a block, and we mine it for a nonce that satisfies the requirements
- Next, we have client that sends those new blocks, any block that now has a valid nonce gets sent to a server on the wallet side. That server on the wallet side receives these blocks and puts them in a blockchain.



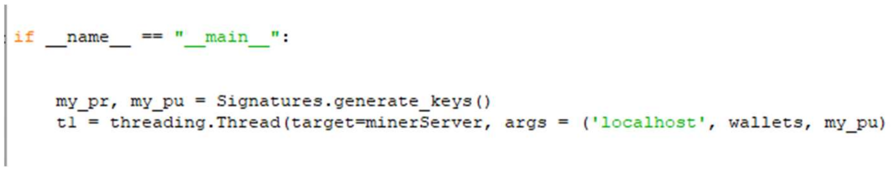
Client → 2 Jobs

1. Create and send transactions
2. Query the balances in the blockchain

EX: if want to know # of coins you have & client will tell you by reading the block

Looking back at what happened last time and how we can improve our testing with threading

Thread that going to be the miner server

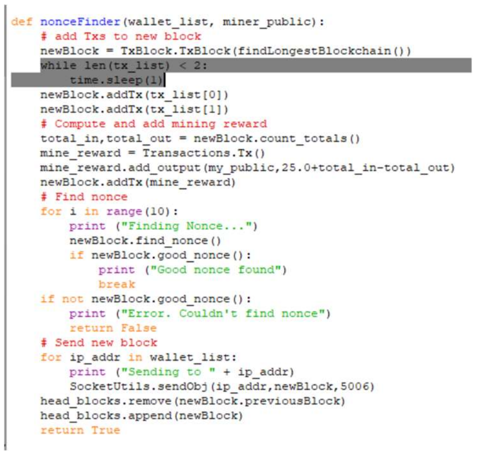


```
if __name__ == "__main__":


    my_pr, my_pu = Signatures.generate_keys()
    t1 = threading.Thread(target=minerServer, args = ('localhost', wallets, my_pu)
```

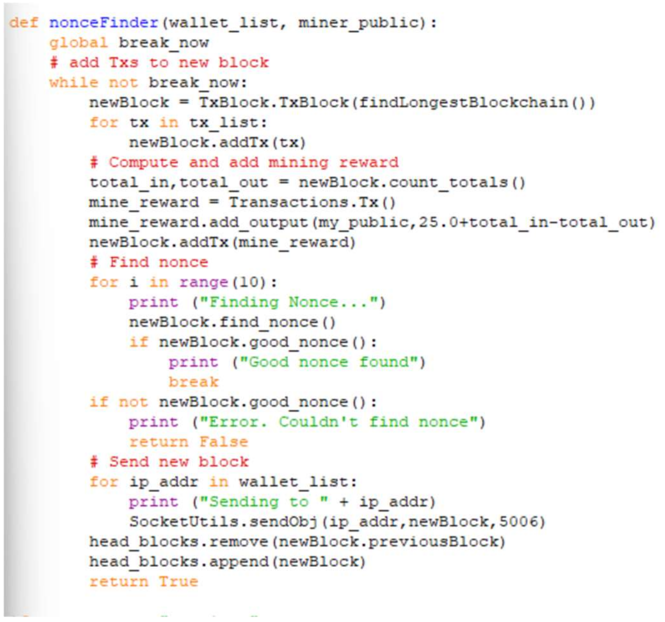
Vid 60 – Assignment 1 Solution

Trying to get the minerServer and nonceFinder to run at the same time

Can do something like a loop here (1:23)

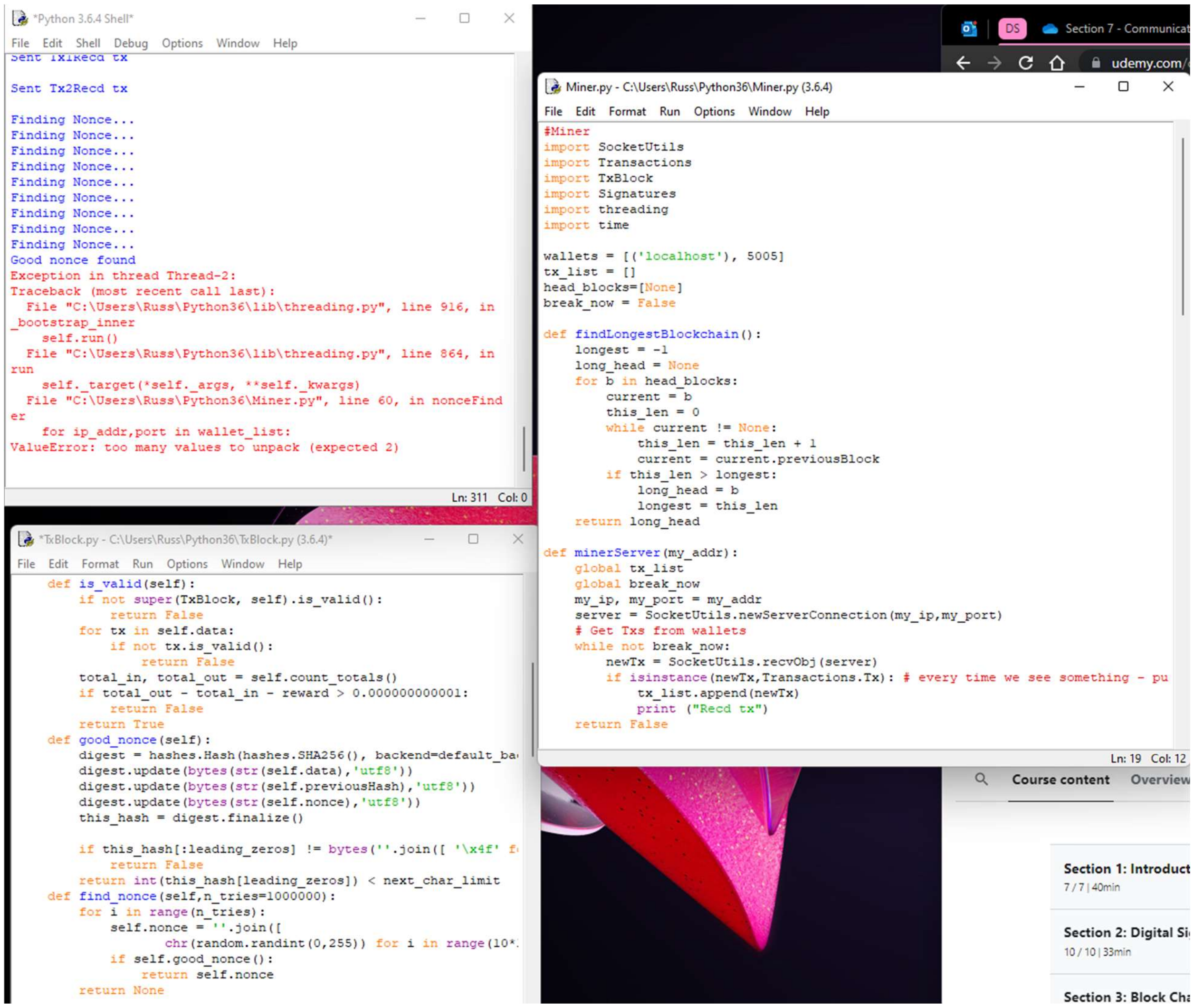


```
def nonceFinder(wallet_list, miner_public):
    # add Txs to new block
    newBlock = TxBlock.TxBlock(findLongestBlockchain())
    while len(tx_list) < 2:
        time.sleep(1)
    newBlock.addTx(tx_list[0])
    newBlock.addTx(tx_list[1])
    # Compute and add mining reward
    total_in,total_out = newBlock.count_totals()
    mine_reward = Transactions.Tx()
    mine_reward.add_output(my_public,25.0+total_in-total_out)
    newBlock.addTx(mine_reward)
    # Find nonce
    for i in range(10):
        print ("Finding Nonce...")
        newBlock.find_nonce()
        if newBlock.good_nonce():
            print ("Good nonce found")
            break
    if not newBlock.good_nonce():
        print ("Error. Couldn't find nonce")
        return False
    # Send new block
    for ip_addr in wallet_list:
        print ("Sending to " + ip_addr)
        SocketUtils.sendObj(ip_addr,newBlock,5006)
    head_blocks.remove(newBlock.previousBlock)
    head_blocks.append(newBlock)
    return True
```



```python
def nonceFinder(wallet_list, miner_public):
    global break_now
    # add Txs to new block
    while not break_now:
        newBlock = TxBlock.TxBlock(findLongestBlockchain())
        for tx in tx_list:
            newBlock.addTx(tx)
        # Compute and add mining reward
        total_in,total_out = newBlock.count_totals()
        mine_reward = Transactions.Tx()
        mine_reward.add_output(my_public,25.0+total_in-total_out)
        newBlock.addTx(mine_reward)
        # Find nonce
        for i in range(10):
            print ("Finding Nonce...")
            newBlock.find_nonce()
            if newBlock.good_nonce():
                print ("Good nonce found")
                break
        if not newBlock.good_nonce():
            print ("Error. Couldn't find nonce")
            return False
        # Send new block
        for ip_addr in wallet_list:
            print ("Sending to " + ip_addr)
            SocketUtils.sendObj(ip_addr,newBlock,5006)
        head_blocks.remove(newBlock.previousBlock)
        head_blocks.append(newBlock)
        return True
```

One issue were gonna have is that this nonce finding is going to take a long time


Python 3.6.4 Shell
File Edit Shell Debug Options Window Help
Ln: 311 Col: 0
Miner.py - C:\Users\Russ\Python36\Miner.py (3.6.4)
File Edit Format Run Options Window Help
Ln: 19 Col: 12
TxBlock.py - C:\Users\Russ\Python36\TxBlock.py (3.6.4)
File Edit Format Run Options Window Help
DS
Section 7 - Communicat
udemy.com/
Course content
Overview
Section 1: Introduct
7 / 7 | 40min
Section 2: Digital Si
10 / 10 | 33min
Section 3: Block Cha

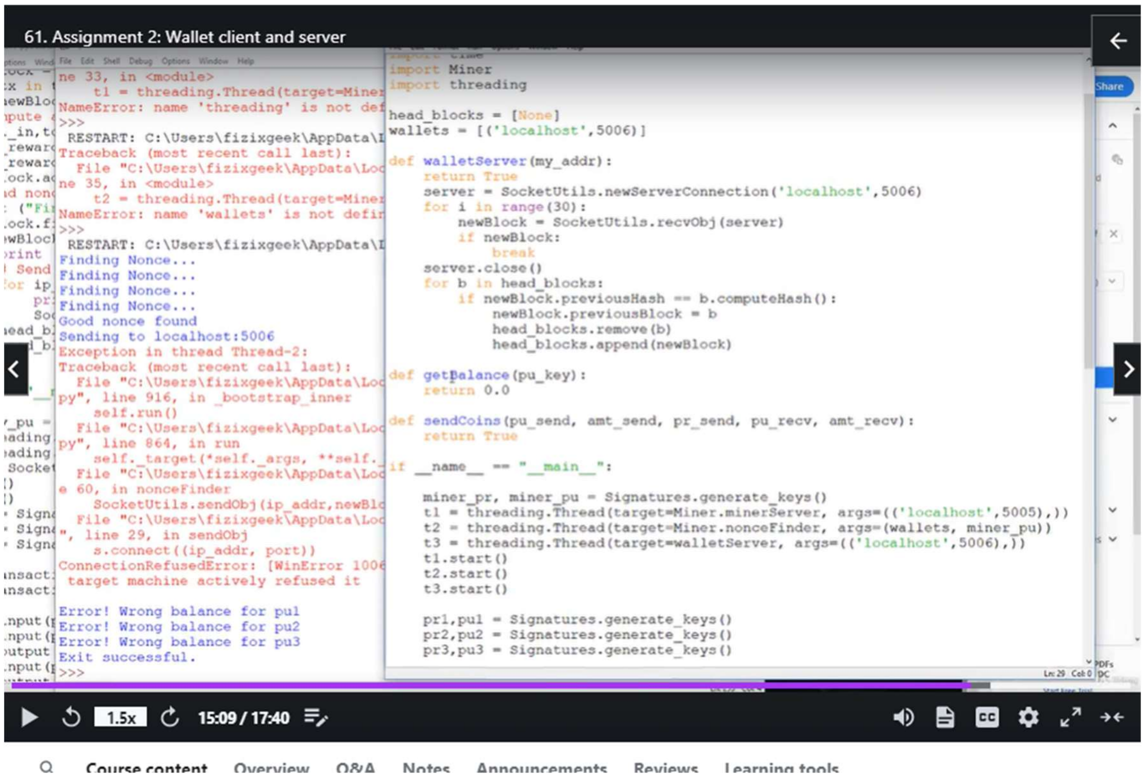
```
Sent Tx1Recd tx

Sent Tx2Recd tx

Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Good nonce found
Exception in thread Thread-2:
Traceback (most recent call last):
  File "C:\Users\Russ\Python36\lib\threading.py", line 916, in
_bootstrap_inner
    self.run()
  File "C:\Users\Russ\Python36\lib\threading.py", line 864, in
run
    self._target(*self._args, **self._kwargs)
  File "C:\Users\Russ\Python36\Miner.py", line 60, in nonceFind
er
    for ip_addr,port in wallet_list:
ValueError: too many values to unpack (expected 2)
```

```python
#Miner
import SocketUtils
import Transactions
import TxBlock
import Signatures
import threading
import time

wallets = [('localhost'), 5005]
tx_list = []
head_blocks=[None]
break_now = False

def findLongestBlockchain():
    longest = -1
    long_head = None
    for b in head_blocks:
        current = b
        this_len = 0
        while current != None:
            this_len = this_len + 1
            current = current.previousBlock
        if this_len > longest:
            long_head = b
            longest = this_len
    return long_head

def minerServer(my_addr):
    global tx_list
    global break_now
    my_ip, my_port = my_addr
    server = SocketUtils.newServerConnection(my_ip,my_port)
    # Get Txs from wallets
    while not break_now:
        newTx = SocketUtils.recvObj(server)
        if isinstance(newTx,Transactions.Tx): # every time we see something - pu
            tx_list.append(newTx)
            print ("Recd tx")
    return False
```

```python
    def is_valid(self):
        if not super(TxBlock, self).is_valid():
            return False
        for tx in self.data:
            if not tx.is_valid():
                return False
        total_in, total_out = self.count_totals()
        if total_out - total_in - reward > 0.000000000001:
            return False
        return True
    def good_nonce(self):
        digest = hashes.Hash(hashes.SHA256(), backend=default_ba
        digest.update(bytes(str(self.data),'utf8'))
        digest.update(bytes(str(self.previousHash),'utf8'))
        digest.update(bytes(str(self.nonce),'utf8'))
        this_hash = digest.finalize()

        if this_hash[:leading_zeros] != bytes(''.join([ '\x4f' f
            return False
        return int(this_hash[leading_zeros]) < next_char_limit
    def find_nonce(self,n_tries=1000000):
        for i in range(n_tries):
            self.nonce = ''.join([
                    chr(random.randint(0,255)) for i in range(10*
            if self.good_nonce():
                return self.nonce
        return None
```

Vid 61 – Assignment 2: Wallet Client and Server

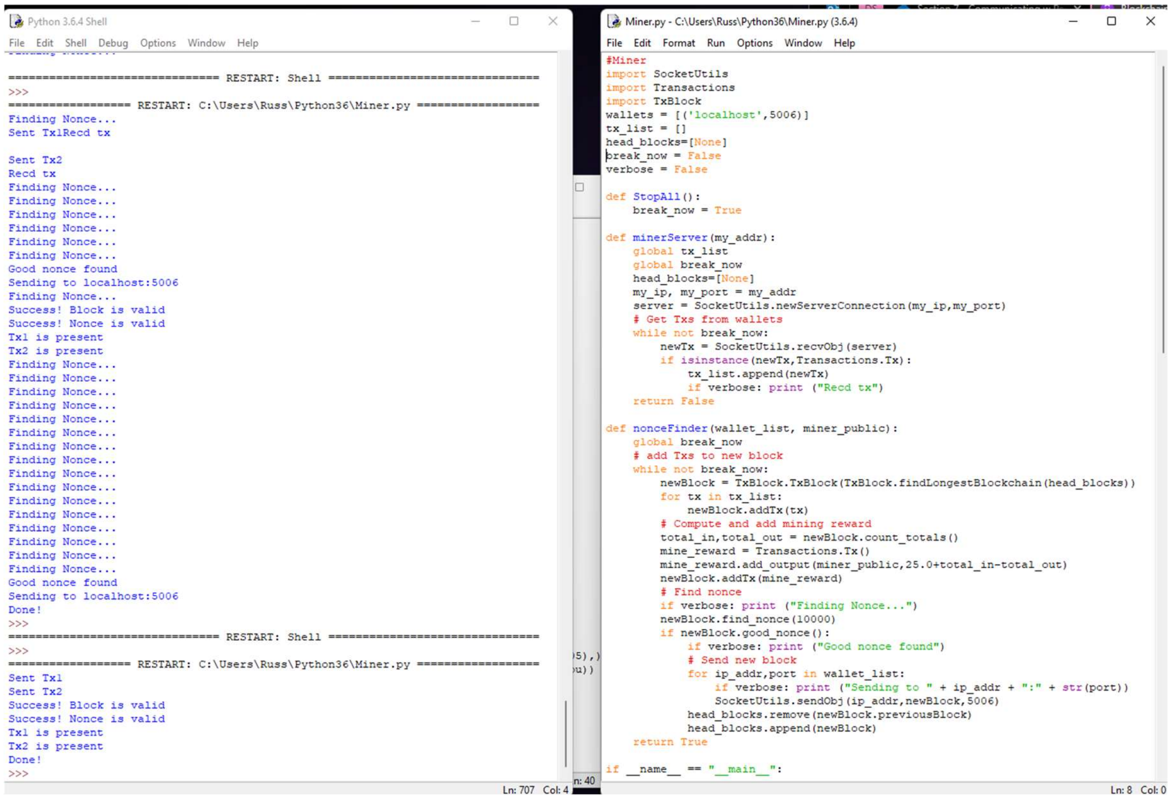
13:54 → good spot for comparing professor's miner and wallet code


61. Assignment 2: Wallet client and server

```
ne 33, in <module>
    t1 = threading.Thread(target=Miner
NameError: name 'threading' is not def
>>>
 RESTART: C:\Users\fizixgeek\AppData\L
Traceback (most recent call last):
  File "C:\Users\fizixgeek\AppData\Loc
ne 35, in <module>
    t2 = threading.Thread(target=Miner
NameError: name 'wallets' is not defin
>>>
 RESTART: C:\Users\fizixgeek\AppData\L
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Good nonce found
Sending to localhost:5006
Exception in thread Thread-2:
Traceback (most recent call last):
  File "C:\Users\fizixgeek\AppData\Loc
py", line 916, in _bootstrap_inner
    self.run()
  File "C:\Users\fizixgeek\AppData\Loc
py", line 864, in run
    self._target(*self._args, **self._
  File "C:\Users\fizixgeek\AppData\Loc
e 60, in nonceFinder
    SocketUtils.sendObj(ip_addr,newBlo
  File "C:\Users\fizixgeek\AppData\Loc
", line 29, in sendObj
    s.connect((ip_addr, port))
ConnectionRefusedError: [WinError 1006
 target machine actively refused it

Error! Wrong balance for pu1
Error! Wrong balance for pu2
Error! Wrong balance for pu3
Exit successful.
>>>
```

```python
import Miner
import threading

head_blocks = [None]
wallets = [('localhost',5006)]

def walletServer(my_addr):
    return True
    server = SocketUtils.newServerConnection('localhost',5006)
    for i in range(30):
        newBlock = SocketUtils.recvObj(server)
        if newBlock:
            break
    server.close()
    for b in head_blocks:
        if newBlock.previousHash == b.computeHash():
            newBlock.previousBlock = b
            head_blocks.remove(b)
            head_blocks.append(newBlock)

def getBalance(pu_key):
    return 0.0

def sendCoins(pu_send, amt_send, pr_send, pu_recv, amt_recv):
    return True

if __name__ == "__main__":

    miner_pr, miner_pu = Signatures.generate_keys()
    t1 = threading.Thread(target=Miner.minerServer, args=(('localhost',5005),))
    t2 = threading.Thread(target=Miner.nonceFinder, args=(wallets, miner_pu))
    t3 = threading.Thread(target=walletServer, args=(('localhost',5006),))
    t1.start()
    t2.start()
    t3.start()

    pr1,pu1 = Signatures.generate_keys()
    pr2,pu2 = Signatures.generate_keys()
    pr3,pu3 = Signatures.generate_keys()
```

Different Outputs when using if verbose

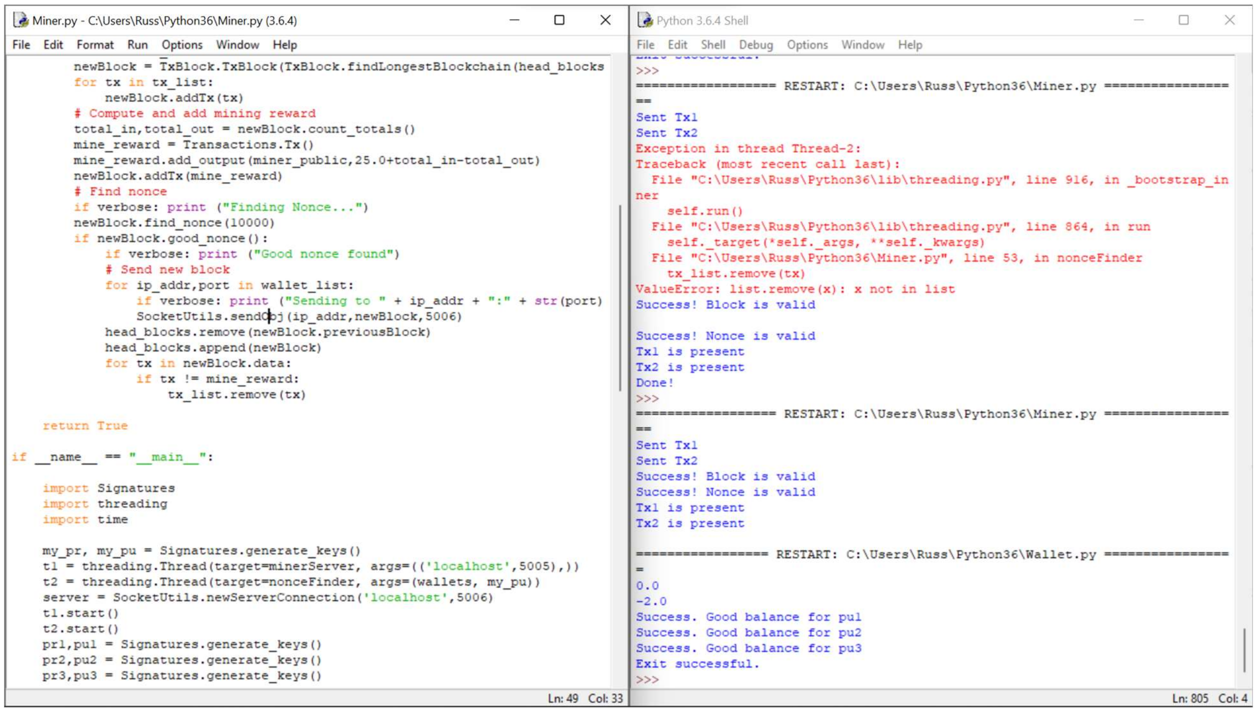

Python 3.6.4 Shell

```
=============================== RESTART: Shell ================================
>>>
================== RESTART: C:\Users\Russ\Python36\Miner.py ==================
Finding Nonce...
Sent Tx1Recd tx

Sent Tx2
Recd tx
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Good nonce found
Sending to localhost:5006
Finding Nonce...
Success! Block is valid
Success! Nonce is valid
Tx1 is present
Tx2 is present
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Finding Nonce...
Good nonce found
Sending to localhost:5006
Done!
>>>
=============================== RESTART: Shell ================================
>>>
================== RESTART: C:\Users\Russ\Python36\Miner.py ==================
Sent Tx1
Sent Tx2
Success! Block is valid
Success! Nonce is valid
Tx1 is present
Tx2 is present
Done!
>>>
```

Miner.py - C:\Users\Russ\Python36\Miner.py (3.6.4)

```python
#Miner
import SocketUtils
import Transactions
import TxBlock
wallets = [('localhost',5006)]
tx_list = []
head_blocks=[None]
break_now = False
verbose = False

def StopAll():
    break_now = True

def minerServer(my_addr):
    global tx_list
    global break_now
    head_blocks=[None]
    my_ip, my_port = my_addr
    server = SocketUtils.newServerConnection(my_ip,my_port)
    # Get Txs from wallets
    while not break_now:
        newTx = SocketUtils.recvObj(server)
        if isinstance(newTx,Transactions.Tx):
            tx_list.append(newTx)
            if verbose: print ("Recd tx")
    return False

def nonceFinder(wallet_list, miner_public):
    global break_now
    # add Txs to new block
    while not break_now:
        newBlock = TxBlock.TxBlock(TxBlock.findLongestBlockchain(head_blocks))
        for tx in tx_list:
            newBlock.addTx(tx)
        # Compute and add mining reward
        total_in,total_out = newBlock.count_totals()
        mine_reward = Transactions.Tx()
        mine_reward.add_output(miner_public,25.0+total_in-total_out)
        newBlock.addTx(mine_reward)
        # Find nonce
        if verbose: print ("Finding Nonce...")
        newBlock.find_nonce(10000)
        if newBlock.good_nonce():
            if verbose: print ("Good nonce found")
            # Send new block
            for ip_addr,port in wallet_list:
                if verbose: print ("Sending to " + ip_addr + ":" + str(port))
                SocketUtils.sendObj(ip_addr,newBlock,5006)
            head_blocks.remove(newBlock.previousBlock)
            head_blocks.append(newBlock)
    return True

if __name__ == "__main__":
```

File Edit Shell Debug Options Window Help
Ln: 707 Col: 4
File Edit Format Run Options Window Help
Ln: 8 Col: 0


Miner.py - C:\Users\Russ\Python36\Miner.py (3.6.4)
File Edit Format Run Options Window Help

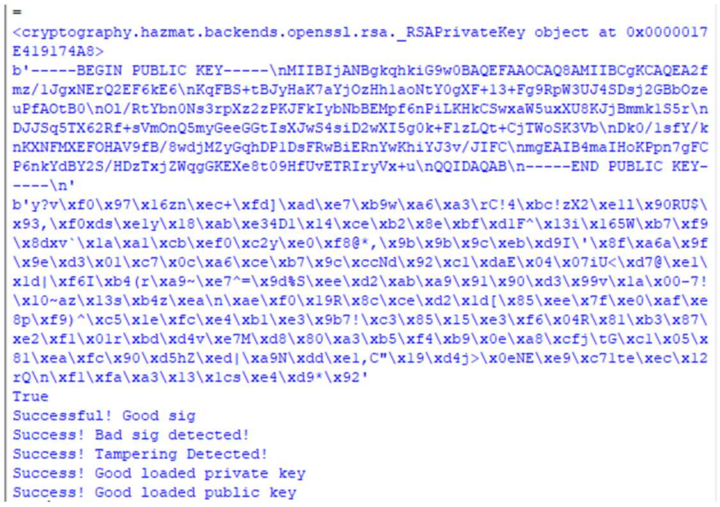
```python
        newBlock = TxBlock.TxBlock(TxBlock.findLongestBlockchain(head_blocks
        for tx in tx_list:
            newBlock.addTx(tx)
        # Compute and add mining reward
        total_in,total_out = newBlock.count_totals()
        mine_reward = Transactions.Tx()
        mine_reward.add_output(miner_public,25.0+total_in-total_out)
        newBlock.addTx(mine_reward)
        # Find nonce
        if verbose: print ("Finding Nonce...")
        newBlock.find_nonce(10000)
        if newBlock.good_nonce():
            if verbose: print ("Good nonce found")
            # Send new block
            for ip_addr,port in wallet_list:
                if verbose: print ("Sending to " + ip_addr + ":" + str(port)
                SocketUtils.sendObj(ip_addr,newBlock,5006)
            head_blocks.remove(newBlock.previousBlock)
            head_blocks.append(newBlock)
            for tx in newBlock.data:
                if tx != mine_reward:
                    tx_list.remove(tx)

    return True

if __name__ == "__main__":

    import Signatures
    import threading
    import time

    my_pr, my_pu = Signatures.generate_keys()
    t1 = threading.Thread(target=minerServer, args=(('localhost',5005),))
    t2 = threading.Thread(target=nonceFinder, args=(wallets, my_pu))
    server = SocketUtils.newServerConnection('localhost',5006)
    t1.start()
    t2.start()
    pr1,pu1 = Signatures.generate_keys()
    pr2,pu2 = Signatures.generate_keys()
    pr3,pu3 = Signatures.generate_keys()
```

```
>>>
================== RESTART: C:\Users\Russ\Python36\Miner.py =================
==
Sent Tx1
Sent Tx2
Exception in thread Thread-2:
Traceback (most recent call last):
  File "C:\Users\Russ\Python36\lib\threading.py", line 916, in _bootstrap_in
ner
    self.run()
  File "C:\Users\Russ\Python36\lib\threading.py", line 864, in run
    self._target(*self._args, **self._kwargs)
  File "C:\Users\Russ\Python36\Miner.py", line 53, in nonceFinder
    tx_list.remove(tx)
ValueError: list.remove(x): x not in list
Success! Block is valid

Success! Nonce is valid
Tx1 is present
Tx2 is present
Done!
>>>
================== RESTART: C:\Users\Russ\Python36\Miner.py =================
==
Sent Tx1
Sent Tx2
Success! Block is valid
Success! Nonce is valid
Tx1 is present
Tx2 is present

================= RESTART: C:\Users\Russ\Python36\Wallet.py =================
=
0.0
-2.0
Success. Good balance for pu1
Success. Good balance for pu2
Success. Good balance for pu3
Exit successful.
>>>
```

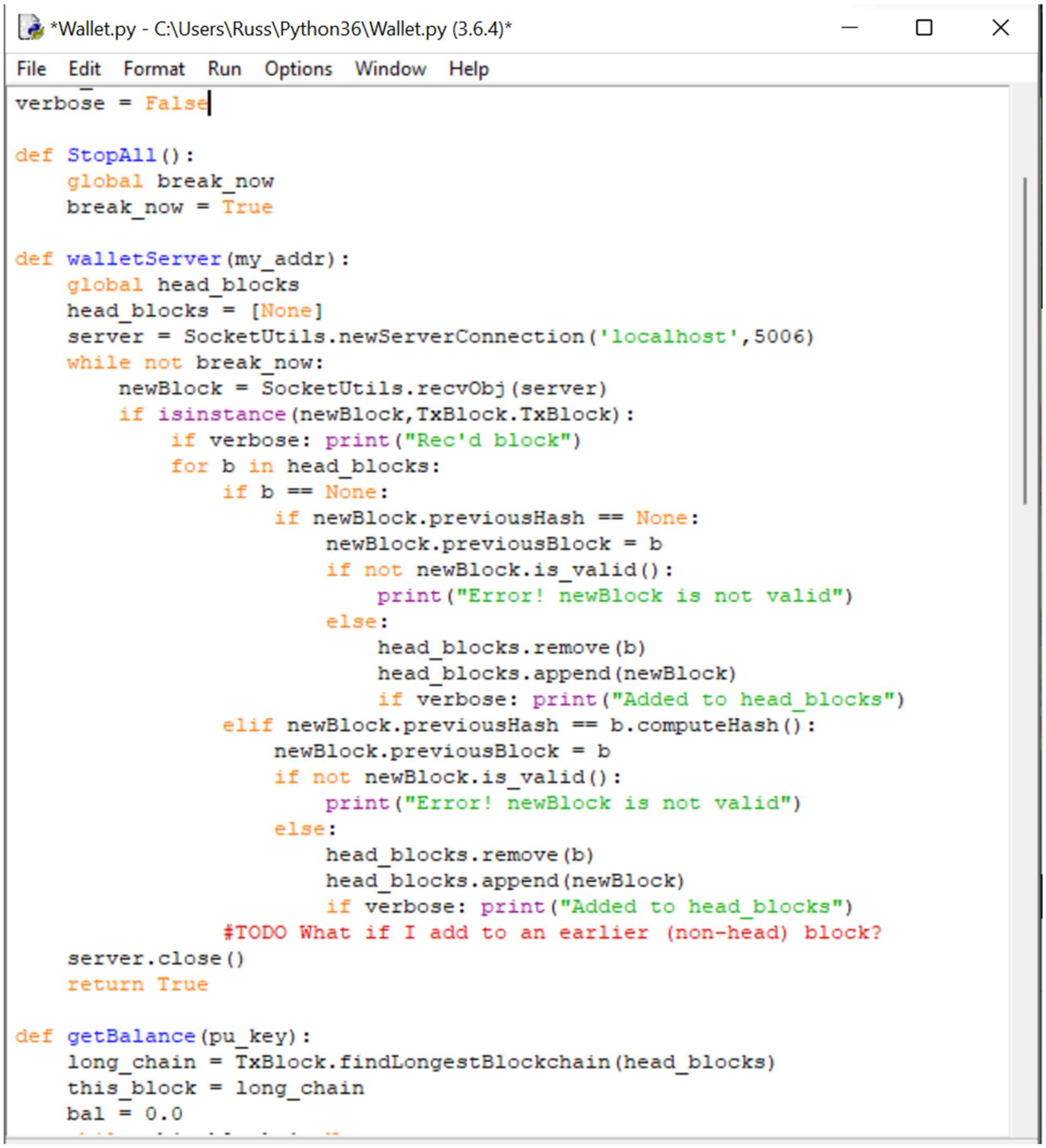
Vid 64 – Assignment 64: Saving and Restoring Keys

https://cryptography.io/en/latest/hazmat/primitives/asymmetric/rsa/#generation

```python
def savePrivate(pr_key, filename):
    pem = pr_key.private_bytes(
        encoding=serialization.Encoding.PEM,
        format=serialization.PrivateFormat.TraditionalOpenSSL,
        encryption_algorithm=serialization.NoEncryption()
    )
    fp = open(filename, "wb")
    fp.write(pem)
    fp.close()
    return
def loadPrivate(filename):
    fin = open(filename, "rb")
    pr_key = serialization.load_pem_private_key(
        fin.read(),
        password=None,
        backend=default_backend()
    )
    fin.close()
    return pr_key
def savePublic(pu_key, filename):
    fp = open(filename, "wb")
    fp.write(pu_key)
    fp.close()
    return True
def loadPublic(filename):
    fin = open(filename, "rb")
    pu_key = fin.read()
    fin.close()
    return pu_key
```



```
=
<cryptography.hazmat.backends.openssl.rsa._RSAPrivateKey object at 0x0000017
E419174A8>
b'-----BEGIN PUBLIC KEY-----\nMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA2f
mz/1JgxNErQ2EF6kE6\nKqFBS+tBJyHaK7aYjOzHhlaoNtY0gXF+13+Fg9RpW3UJ4SDsj2GBbOze
uPfAOtB0\nO1/RtYbn0Ns3rpXz2zPKJFkIybNbBEMpf6nPiLKHkCSwxaW5uxXU8KJjBmmk1S5r\n
DJJSq5TX62Rf+sVmOnQ5myGeeGGtIsXJwS4siD2wXI5g0k+F1zLQt+CjTWoSK3Vb\nDk0/1sfY/k
nKXNFMXEFOHAV9fB/8wdjMZyGqhDP1DsFRwBiERnYwKhiYJ3v/JIFC\nmgEAIB4maIHoKPpn7gFC
P6nkYdBY2S/HDzTxjZWqgGKEXe8t09HfUvETRIryVx+u\nQQIDAQAB\n-----END PUBLIC KEY-
----\n'
b'y?v\xf0\x97\x16zn\xec+\xfd]\xad\xe7\xb9w\xa6\xa3\rC!4\xbc!zX2\xe11\x90RU$\
x93,\xf0xds\xe1y\x18\xab\xe34D1\x14\xce\xb2\x8e\xbf\xd1F^\x13i\x165W\xb7\xf9
\x8dxv`\x1a\xa1\xcb\xef0\xc2y\xe0\xf8@*,\x9b\x9b\x9c\xeb\xd9I\'\x8f\xa6a\x9f
\x9e\xd3\x01\xc7\x0c\xa6\xce\xb7\x9c\xccNd\x92\xc1\xdaE\x04\x07iU<\xd7@\xe1\
x1d|\xf6I\xb4(r\xa9~\xe7^=\x9d%S\xee\xd2\xab\xa9\x91\x90\xd3\x99v\x1a\x00-7!
\x10~az\x13s\xb4z\xea\n\xae\xf0\x19R\x8c\xce\xd2\x1d[\x85\xee\x7f\xe0\xaf\xe
8p\xf9)^\xc5\x1e\xfc\xe4\xb1\xe3\x9b7!\xc3\x85\x15\xe3\xf6\x04R\x81\xb3\x87\
xe2\xf1\x01r\xbd\xd4v\xe7M\xd8\x80\xa3\xb5\xf4\xb9\x0e\xa8\xcfj\tG\xc1\x05\x
81\xea\xfc\x90\xd5hZ\xed|\xa9N\xdd\xe1,C"\x19\xd4j>\x0eNE\xe9\xc71te\xec\x12
rQ\n\xf1\xfa\xa3\x13\x1cs\xe4\xd9*\x92'
True
Successful! Good sig
Success! Bad sig detected!
Success! Tampering Detected!
Success! Good loaded private key
Success! Good loaded public key
```


Wallet.py - C:\Users\Russ\Python36\Wallet.py (3.6.4)
File Edit Format Run Options Window Help

```
verbose = False

def StopAll():
    global break_now
    break_now = True

def walletServer(my_addr):
    global head_blocks
    head_blocks = [None]
    server = SocketUtils.newServerConnection('localhost',5006)
    while not break_now:
        newBlock = SocketUtils.recvObj(server)
        if isinstance(newBlock,TxBlock.TxBlock):
            if verbose: print("Rec'd block")
            for b in head_blocks:
                if b == None:
                    if newBlock.previousHash == None:
                        newBlock.previousBlock = b
                        if not newBlock.is_valid():
                            print("Error! newBlock is not valid")
                        else:
                            head_blocks.remove(b)
                            head_blocks.append(newBlock)
                            if verbose: print("Added to head_blocks")
                elif newBlock.previousHash == b.computeHash():
                    newBlock.previousBlock = b
                    if not newBlock.is_valid():
                        print("Error! newBlock is not valid")
                    else:
                        head_blocks.remove(b)
                        head_blocks.append(newBlock)
                        if verbose: print("Added to head_blocks")
                #TODO What if I add to an earlier (non-head) block?
    server.close()
    return True

def getBalance(pu_key):
    long_chain = TxBlock.findLongestBlockchain(head_blocks)
    this_block = long_chain
    bal = 0.0
```

Vid 66 – Save and Restore Blocks and Transactions