**以太坊核心存储结构剖析**

**以太坊核心存储结构：Merkle-Patricia-Tree（前缀树与默克尔树的结合体）**

**以太坊中的交易树、交易收据树、账户树以及合约存储树均使用该树索引**

**该树分为三种类型节点：branch（分支，17个元素的元组）、extension（扩展，2个元素的元组<k, v>）、leaf（叶子节点，2个元素的元组<k, v>），因此为了区分 extension 与 leaf 节点，使用 key 的第一个16进制字符，其中0000与0001均代表扩展节点，0010与0011均代表叶子节点，也就是说使用倒数第二位来区分 extension 与 leaf 节点。最后一位的0、1分别表示了该 key**

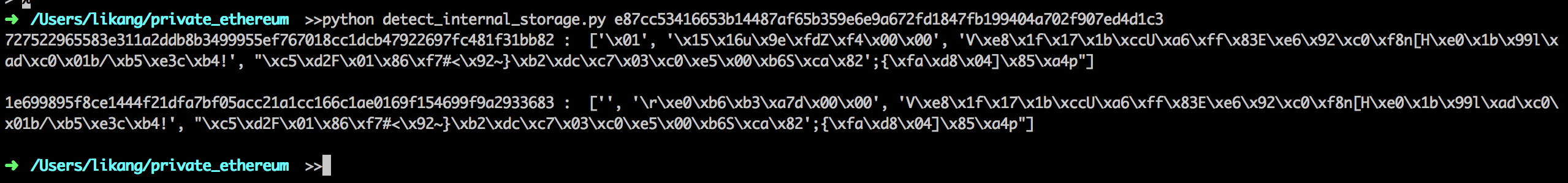
**原先为偶数个16进制字符与奇数个16进制字符，也就意味着为 0 时，需要填充另外的0000**

**下面来看一下以太坊底层存储中是如何实现账户存储的：**

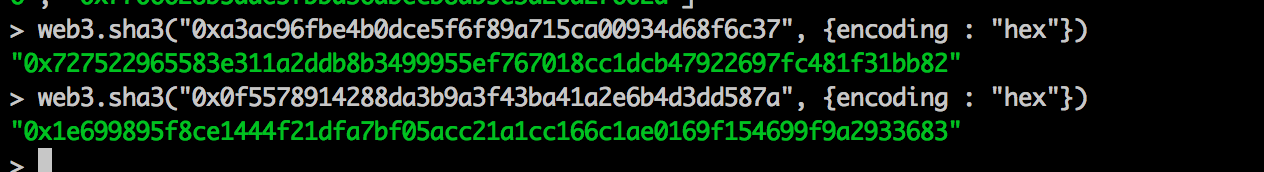
****

**目前以太坊中存在两个账户：0xa3ac96fbe4b0dce5f6f89a715ca00934d68f6c37、0x0f5578914288da3b9a3f43ba41a2e6b4d3dd587a**

**通过使用编写的拉取以太坊底层存储的 python 脚本，拉取目前底层存储的账户数据：**

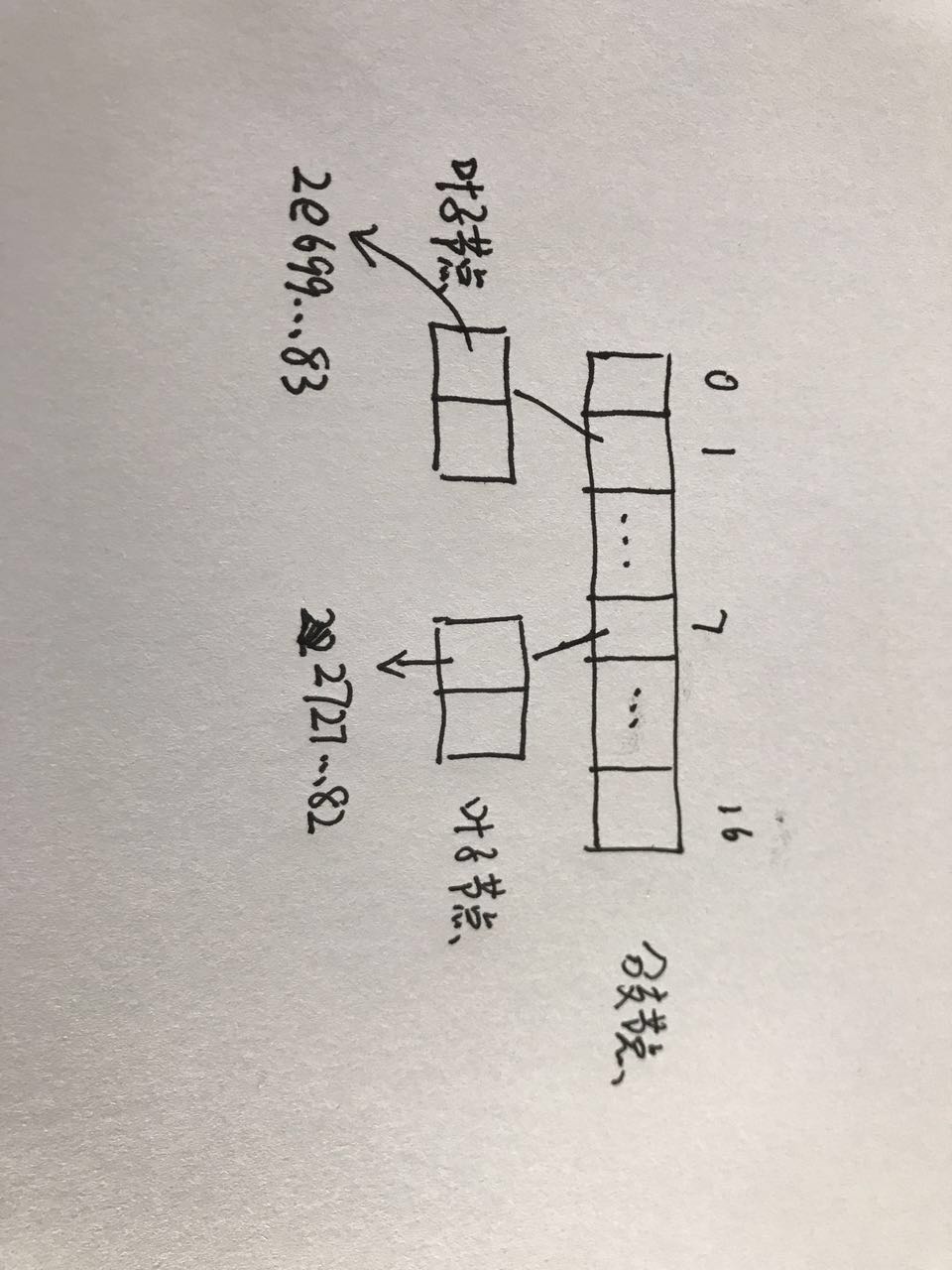
****

**其中前一项是以太坊在底层中真正存储的 key，与账户地址的对应关系如下：**

****

**也就是说，以太坊存储账户数据的时候，会将账户地址进行一次keccak256 哈希计算**

**此时，账户树的形状如下：**

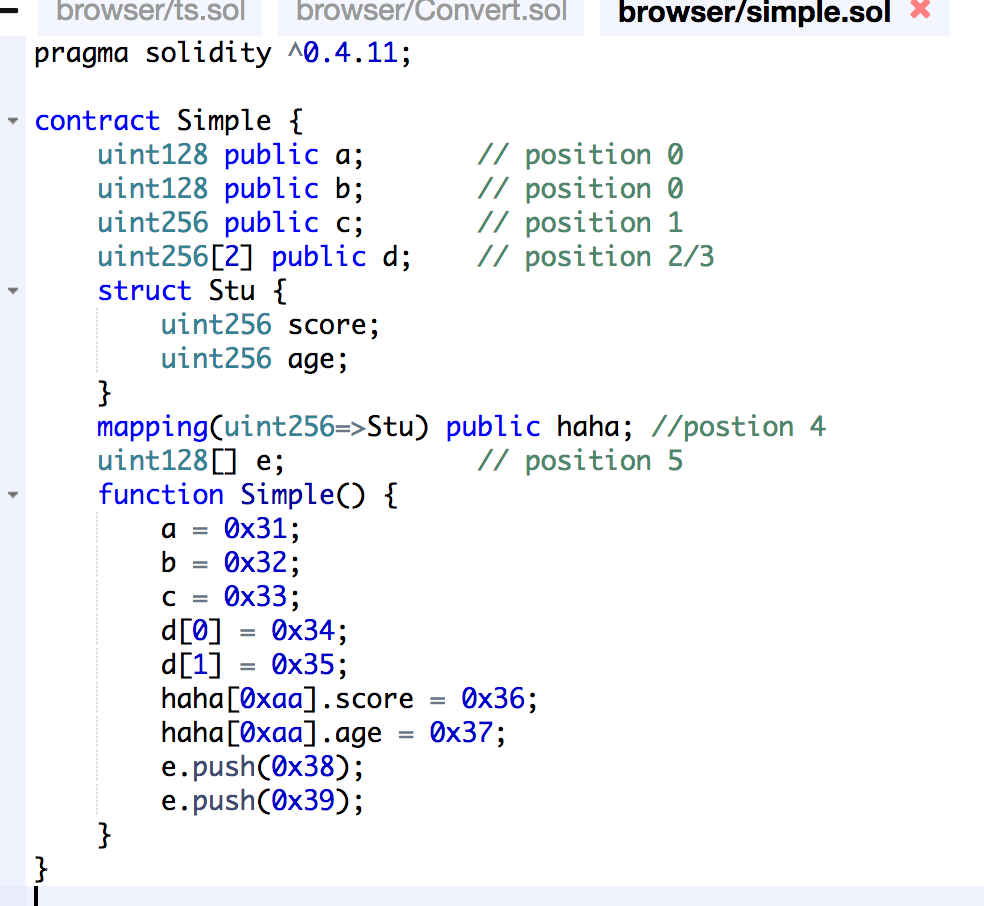
****

**以太坊中会对节点使用 RLP 编码来存储在底层数据 leveldb 或 rocksdb 中，存储形式为 <sha3(rlp(node)), rlp(node)>。所以在上图的槽中，slot 1 实际存储的是 sha3(rlp(leaf1))**

**下面来看一下智能合约中的数据是如何存储在以太坊中的：**

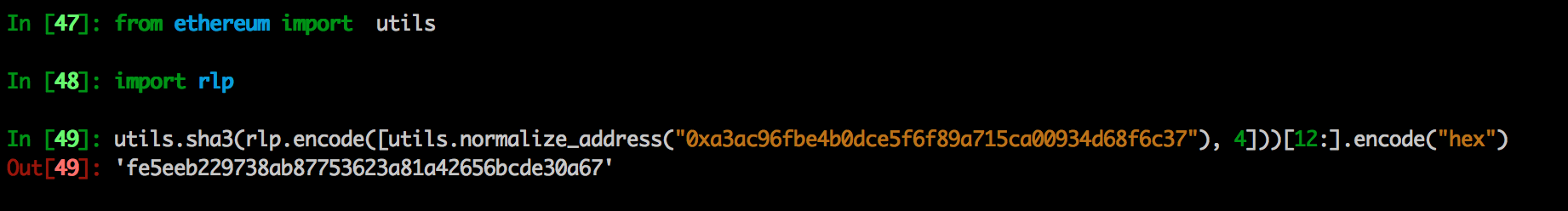
**智能合约中的每一个状态变量都有一个 position，以太坊中每一个**

**storage slot 为 32 个字节，即 256 位，solidity 编译器会尽量将变量装到同一个 storage slot 中去，对于装不下的，会重新分配 storage slot，遇到 mapping、struct 等类型的变量时，编译器会自动重新分配 storage slot。**

****

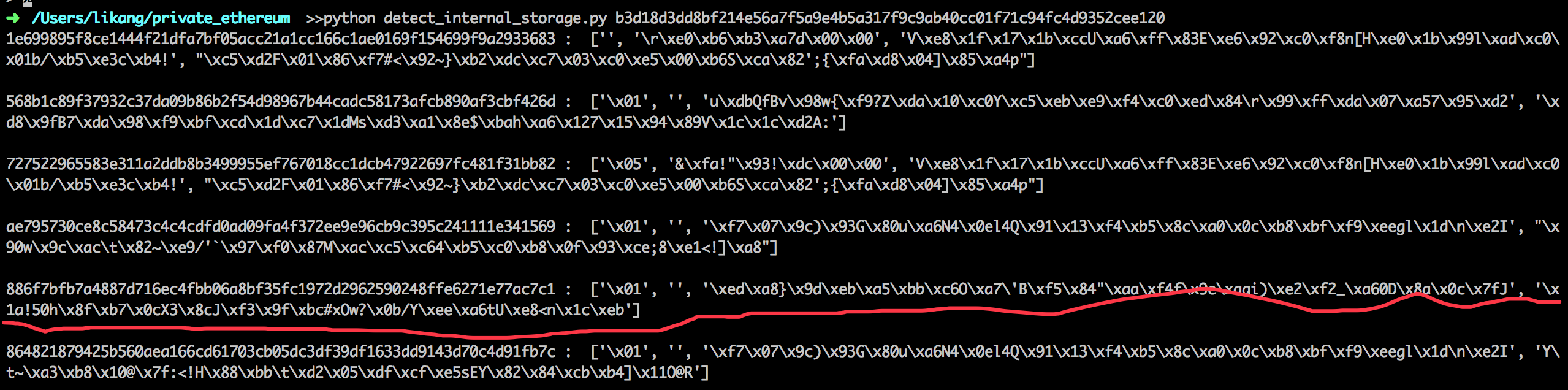
**该代码存在于账户下，该合约的地址为** 0xfe5eeb229738ab87753623a81a42656bcde30a67

**contract address = sha3(rlp.encode([creator address, nonce]))**

****

**该账户在底层数据库中存储的 key 为 > web3.sha3("**0xfe5eeb229738ab87753623a81a42656bcde30a67**", {encoding : "hex"})**

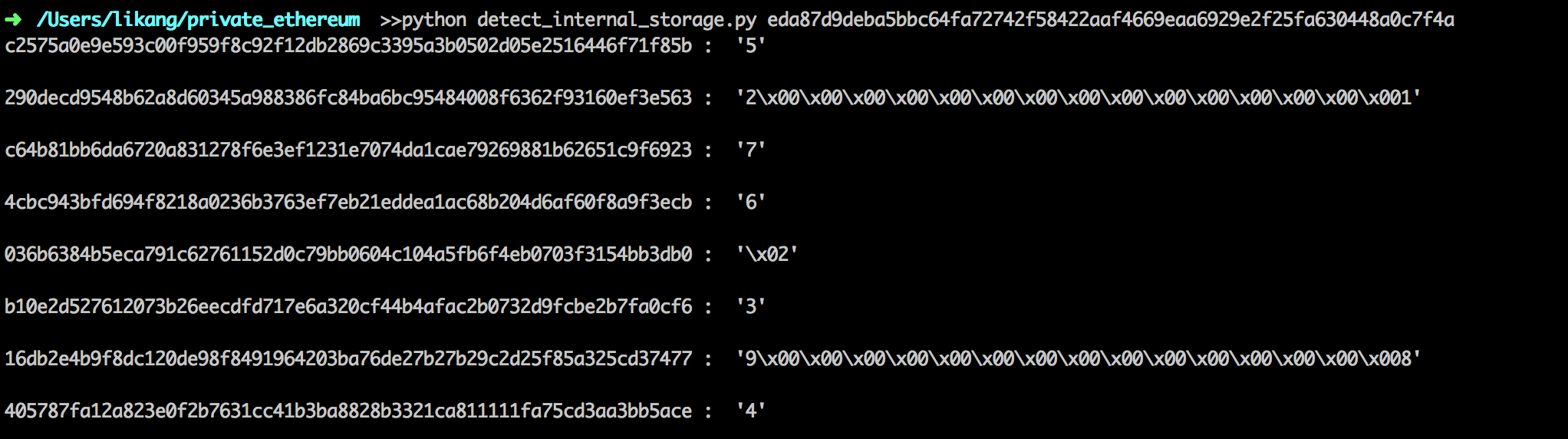
**0x886f7bfb7a4887d716ec4fbb06a8bf35fc1972d2962590248ffe6271e77ac7c1**

****

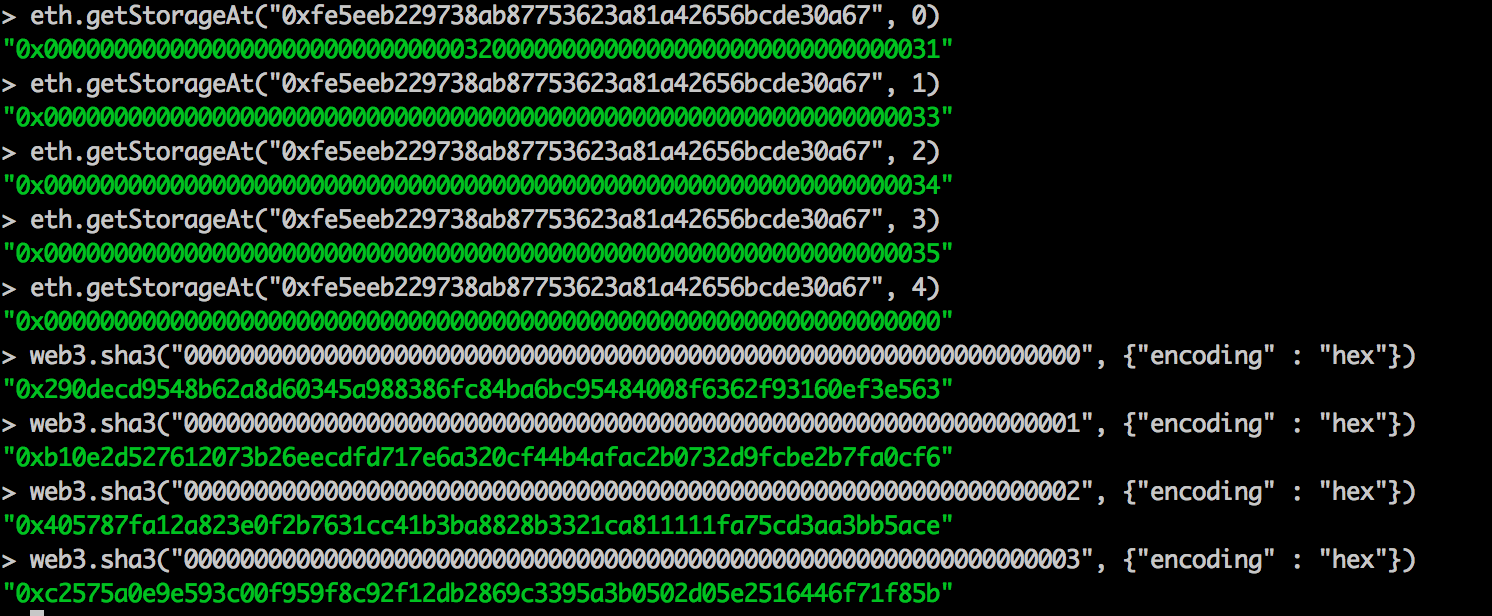
**然后我们根据该账户对应的 value 中的第三个元素（存储树的根哈希值），再次使用上述提到的解析存储脚本：**

**In [50]: '\xed\xa8}\x9d\xeb\xa5\xbb\xc6O\xa7\'B\xf5\x84"\xaa\xf4f\x9e\xaai)\xe2\xf2\_\xa60D\x8a\x0c\x7fJ'.encode("hex")**

**Out[50]: 'eda87d9deba5bbc64fa72742f58422aaf4669eaa6929e2f25fa630448a0c7f4a'**

****

**我们来一一分析：**

****

**我们可以看到在相应的位置上分别存储了相应的值，**

**对于 mapping 来说，其中元素存储的position如下：**

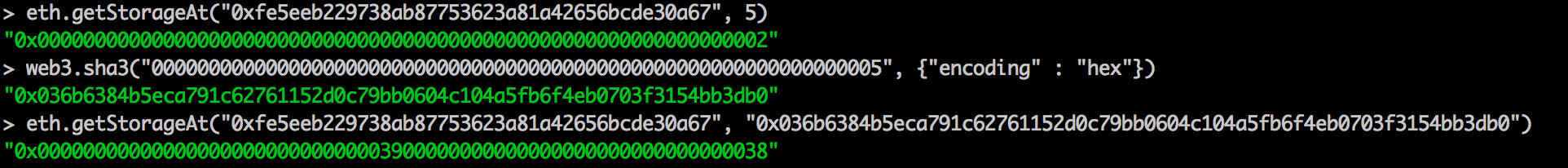
**sha3(LeftPad32(key, 0), LeftPad32(map position, 0))**

**所以有：**

****

**由于 mapping 中 value 为一个 struct，size 大于 256 位，因此，在存储的位置按顺位加 1，如上图所示。**

**来看一下，动态数组在以太坊底层存储形式：**

****

**其中，在位置 5 存储动态数组的长度，然后以位置 sha3(5) 开始顺序存放数组元素。**