Merkle Tree 的简单实现

Merkle Tree，通常也被称作Hash Tree。是用于存储hash值的一棵树。 Merkle树一个很重要的用处是检查块中是否包含指定的交易，Merkle树是通过递归哈希节点对来构造的，直到只有一个哈希。

构造原理

Merkle树的叶子节点上的value，可由我们根据设计需要来指定。 3非叶子节点的value是根据它下面所有的叶子节点值，然后按照一定的算法计算而得出的。

我们使用一个二叉树来模拟Merkle Tree的操作 过程分为：

1.准备交易的数据

2.计算出每个数据的哈希值，从左到右逐步组成树的左右节点

3.执行循环直到最后一个节点 这里核心的算法就是构造树，并且要计算每个节点的哈希值

实现过程

我们构造一个11个叶子节点的二叉树，然后测试执行返回各节点的哈希值.

```
time1 = time.time()
data = ['a','b','c','d','e','f','g','h','i','j','k']
Hdata = data_hash(data)
Merkle(Hdata)
time2 = time.time()
time = time2 - time1
print(time)
```

执行结果：

```
========== RESTART: C:\Users\dell\Desktop\mt2.py ==========
['0cc175b9c0f1b6a831c399e269772661', '92eb5ffee6ae2fec3ad71c777531578f', '4a8a08f09d37b73795649038408b5f33', '8277e0910d750195b448797616e091ad', 'e1671797c52e15f763380b45e841ec32', '8fa14cdd754f91cc6554c9e71929cce7', 'b2f5ff47436671b6e533d8dc3614845d', '2510c39011c5be704182423e3a695e91', '865c0c0b4ab0e063e5caa3387c1a8741', '363b122c528f54df4a0446b6bab05515', '8ce4b16b22b58894aa86c421e8759df3']
['3bc22fb7aaebe9c8c5d7de312b876bb8', '1ee715fcc373a83611d163d9d4ea02a3', '18b32bf9b1d0a31db2b1ba789eedecf4', '5ff16050d33f486f1381641b2866d0b2', 'bfec547b3e7f1dcfad66118def6c12ec', '91f0918cf58f4e83396c0522fcbd469e']
['dfbd34a5b6634d7ccd0aa26a4dec2ecc', 'b5e76f350f3e9373c4aedaf63c33067b', '4fc9baa84db4e5626a22e939859e41ba']
['cdb375e6b952328a2056dbb10bed9a49', '3f22fa23e202fb1f695773e90816ea51']
['da0be7b2b07453d343aa0a0a8b845798']
>>>
```