2024-2025学年《区块链基础及应用》

tips：suming老师特别喜欢改题，所以千万不要以往年的考试来复习。只起到一个参考的作用，切记。好好做实验，好好看PPT，好好做CS251，其他就没啥了。

1　一、判断题(2分一个)

1.我们需要控制区块链挖矿难度不变，维持区块链的稳定性

2.不存在一个区块链的交易，其输入和输出都只有一个

3.给了一个区块链交易的具体代码，判断矿工挖出这个区块，所得的比特币是不是对应的数值

4.接上一题，判断脚本中第一行的hash是不是所有交易的hash

5.接上一题，这个交易已经有8个确认了，所以基本上不会被双花了

6.为了保证个人数据的安全性，可以将可证明安全写入到智能合约中

7.区块链中，交易费越高，就越容易得到共识

8.以太坊是图灵完备的，所以其编程语言支持所有类型的运算

9.蒙特币运用的技术是zn-shark

10.区块链的扫码支付和微信支付宝的原理不相同

2　二、填空题(5分一个)

1.比特币的市值为1万美金一个，求所有比特币的总价值

2.填写OP指令，填OP_EQUALVERIFY即可

```
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
```

3 三、解答题

3.1 1.求解线性方程组谜题

跟实验是一样的代码：

```
ex3a_txout_scriptPubKey = [
    OP_2DUP,
    OP_ADD,
    2211,
    OP_EQUALVERIFY,
    OP_SUB,
    43,
    OP_EQUAL
]
```

3.2 2.scrypt挖矿伪代码,bitcoin挖矿伪代码，scrypt相比bitcoin的优势

scrypt:

```
//ASIC mining code
def scrypt(N,seed):
    V = [0] * N
    V[0] = seed
    for(i = 1 to N){
        V[i] = SHA256(V[i-1])//遍历求解
    }
    X=SHA256(V[N-1])//随机选的，就是为了进行检测
    for(i = 1 to N){
        j = X % N
        X = SHA256(X xor V[j])//计算hash值，然后再进行循环
    }
    return X
```

CPU:

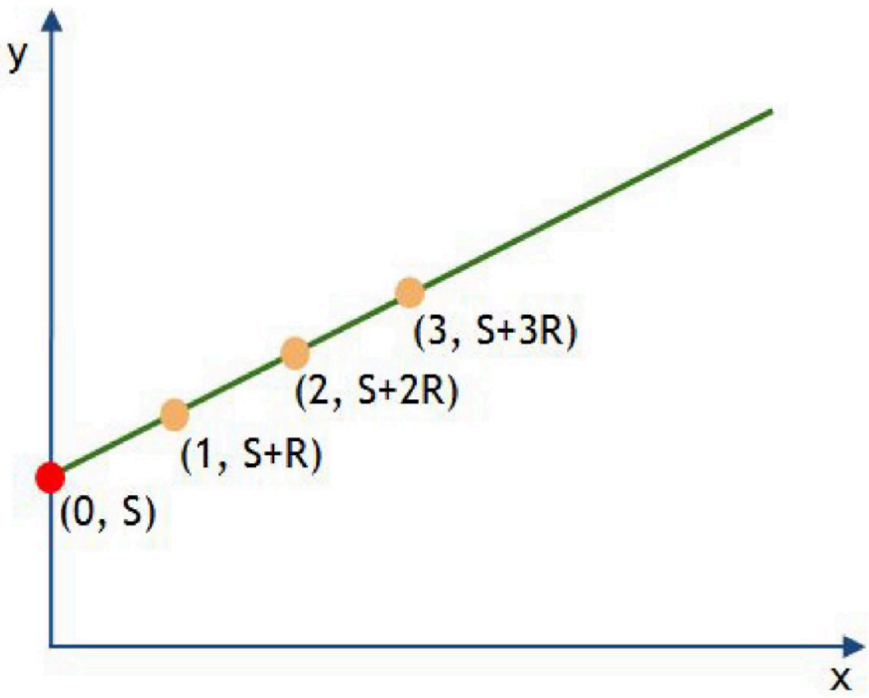
```
//CPU mining code
TARGET = (65535<<208)/DIFFICULTY;
coinbase_nonce = 0;
while(1){
    header = makeBlockHeader(transactions,coinbase_nonce);
    for(header_nonce=0;header_nonce<(1<<32);header_nonce++){
        if(SHA256(SHA256(makeBlock(header,header_nonce)))<TARGET){
            break;
        }
    }
    coinbase_nonce++;
}
```

优势：将算力集中转换到了内存集中，一定程度上保证了比特币的去中心化，让个人也可以挖出比特币

3.3 3. 2/3密钥分存设计

很简单，就是设计一个一次函数就行了

2 out of 3的设计



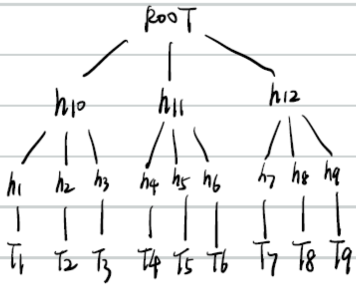
Geometric illustration of 2-out-of-N secret sharing

3.4 4.merkle树证明设计

第一问是作业原题

$S=(T_1,T_2,\cdots T_9)$，共9个节点

$\because k=3$，为三叉树，构建以下Merkle树：



假设哈希函数为Hash(x)，令 $h_i=Hash(T_i)$，则：

Alice需要计算：
$$\begin{cases} h_{10}=Hash(h_1\|h_2\|h_3) \\ h_{11}=Hash(h_4\|h_5\|h_6) \\ h_{12}=Hash(h_7\|h_8\|h_9) \end{cases}$$

然后，Alice需要计算ROOT节点： $ROOT=Hash(h_{10}\|h_{11}\|h_{12})$

Alice通过将ROOT提交给Bob来承诺集合 $S=(T_1,T_2\cdots T_9)$

证明T4在S中：

Alice需要提供 h_5,h_6,h_{10},h_{12} 四个节点的值

为了验证T4在S中，我们只需要计算：

$Hash\left(h_{10}\|Hash\left(Hash(T_4)\|h_5\|h_6\right)\|h_{12}\right)$ 的值，然后和提交的ROOT的值作比较即可

$\begin{cases} \text{若两者相等，则证明T4在S中} \\ \text{若两者不相等，则T4不在S中} \end{cases}$

第二问，修改hash协议为hash(v,r)，r为随机值，为了保证v不被泄露，设计一个新的承诺和证明方案

我认为因为r是随机值，我们只需要修改hash的计算为$hash(v\ xor\ r)$即可

这样就保证了安全性

3.5 5.跨链交易设计

和实验一模一样，简单设计一下就行了

```
    # 步骤 1: 验证收款人签名，无论任何情况都需要收款人的签名正确
    public_key_recipient,
    OP_CHECKSIG,  # 检查签名是否有效

    # 如果收款人签名正确
    OP_IF,
        # 步骤 2: 检查收款人是否提供了 secret 来进行赎回
        OP_IF,
            OP_HASH160,  # 对提供的 secret 进行哈希计算
            hash_of_secret,
            OP_EQUAL,  # 判断是否匹配
            OP_IF,
                OP_1,  # 匹配成功
            OP_ENDIF,

        # 步骤 3: 如果没有提供 secret，则判断发送方是否签名
        OP_ELSE,
            # 将发送方的公钥压入堆栈，用于验证发送方的签名
            public_key_sender,
            OP_CHECKSIG,  # 判断发送方的签名是否有效
            OP_IF,
                OP_1,  # 有效则赎回
            OP_ENDIF,

        OP_ENDIF,

    OP_ENDIF
]
```

3.6 6.难度题：2023年CS251 homework4第一题

Problem 1. In Lecture 15, starting on slide 15, we defined the concept of a polynomial commitment scheme (PCS). In this exercise we will develop an important application for a PCS. First, let us briefly review what is a PCS. A PCS is a tuple of four algorithms: *setup*, *commit*, *prove*, and *verify*. The PCS is initialized by running $setup(d)$ to obtain some public parameters pp. Carol (the committer) has a univariate polynomial $f \in \mathbb{F}_p[X]$ of degree at most d. Carol can commit to f by sending to Roger (the recipient) a commitment string com_f obtained by running $commit(pp, f)$. Later, Roger can choose some $u \in \mathbb{F}_p$ and ask Carol to send him $v := f(u) \in \mathbb{F}_p$ along with a proof $\pi_{u,v}$ that v is indeed the evaluation of the committed polynomial at u. Carol constructs the proof by running $Prove(pp, (u, v), f)$. Roger can verify the proof by running $verify(pp, (com_f, u, v), \pi_{u,v})$ which outputs accept or reject. If *verify* outputs accept then Roger is convinced that the committed polynomial f satisfies $f(u) = v$ and that f is a univariate polynomial of degree at most d. There are PCS constructions where com_f and $\pi_{u,v}$ are as short as 200 bytes each, no matter what d is.

Next, suppose Carol has a set $S = \{s_1, \ldots, s_n\} \subseteq \mathbb{F}_p$. Carol wants to commit to S so that later, given some $s \in \mathbb{F}_p$, if s is in S then she can convince Roger of that fact (an inclusion proof), and if s is not in S then she can convince Roger of that fact (an exclusion proof). One solution is to commit to S using a Merkle tree, where the Merkle root is the commitment to S. Then, for $s \in S$ she can send Roger a Merkle proof of size $O(\log n)$ to convince Roger that s is in S.

Let's see how we can do better using a PCS.

a. Show how Carol can use a PCS to commit to the set S so that later, when Roger sends an $s \in \mathbb{F}_p$, Carol can provide a *constant size* inclusion or an exclusion proof for s that convinces Roger. Explain how Carol commits to S, and how she constructs the exclusion or inclusion proof for a given $s \in \mathbb{F}_p$.
Hint: consider having Carol use the polynomial $f_S(X) := (X - s_1) \cdots (X - s_n) \in \mathbb{F}_p[X]$.

b. For a large n, the inclusion/exclusion proofs in part (a) are already shorter than a Merkle proof. Let's do even better. Suppose Roger sends to Carol $u_1, \ldots, u_k \in \mathbb{F}_p$ and all of them happen to be in S. Carol wants to convince Roger of that fact. Using a Merkle tree, Carol would need to send over a proof of size $O(k \log n)$ – one Merkle inclusion proof for each u_i. Show that using the commitment scheme from part (a), Carol can convince Roger using a *constant size proof* (independent of n and k). You may assume that n/p is negligible.
Hint: Both Carol and Roger can construct the polynomial $g(X) := (X - u_1) \cdots (X - u_k)$. Carol will then prove to Roger that $g(X)$ divides $f_S(X)$. Try doing so using the quotient polynomial technique used in Lecture 15 slide 32. Explain why your short inclusion proof convinces Roger.

还甚至去掉了hint，不是人能写的只能说，94分起扣(bushi)

后记：整张卷子做下来，就是感觉和去年考的完全不一样，只能说后来人加油吧