周亚男 2020131062

完善合约代码

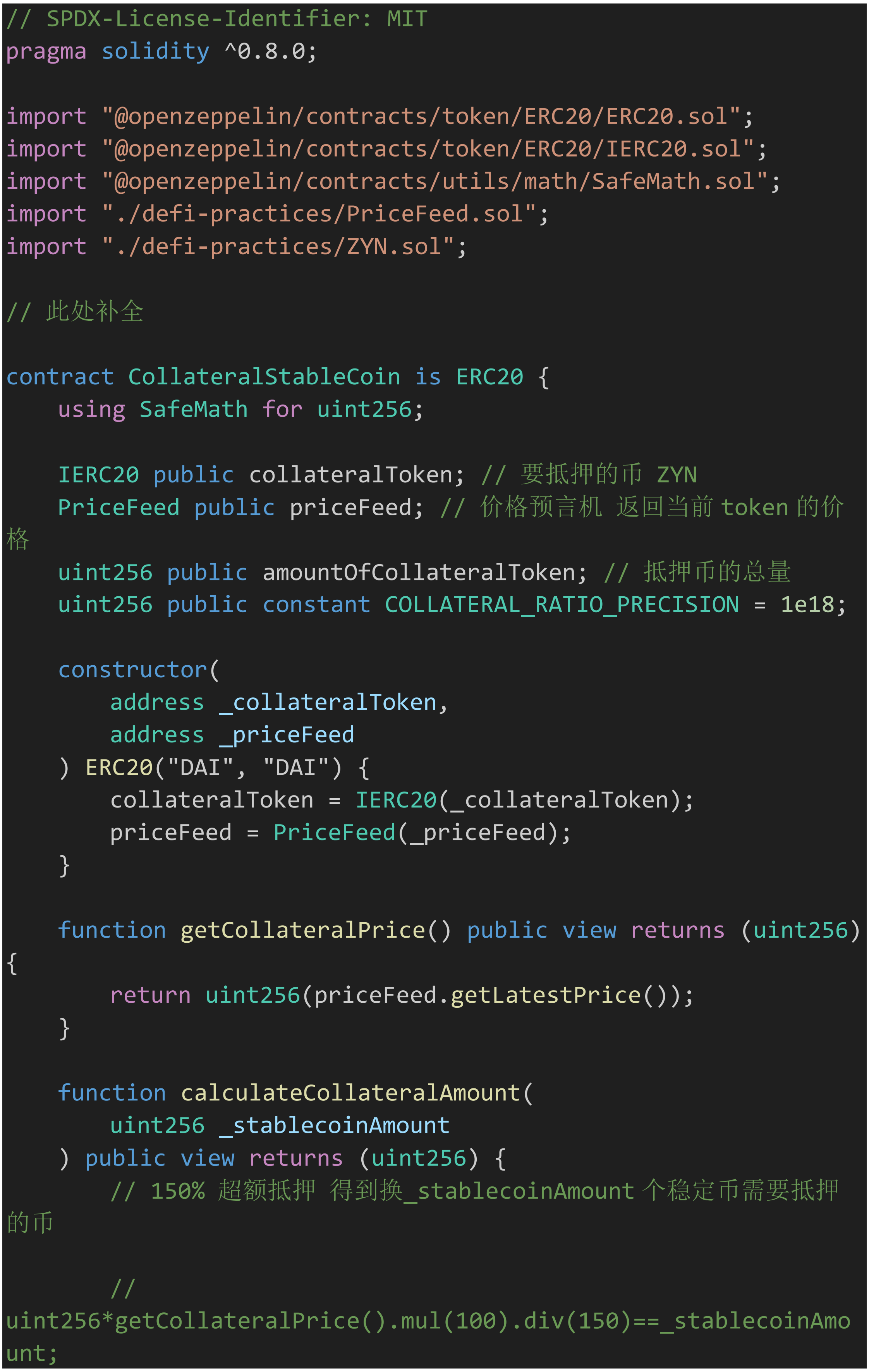
ZYN.sol

```
// SPDX-License-Identifier: SEE LICENSE IN LICENSE
pragma solidity ^0.8.12;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/access/Ownable.sol";
import
"@openzeppelin/contracts/token/ERC20/extensions/ERC20Burnable.
sol";

contract ZYN is ERC20, Ownable, ERC20Burnable {
    constructor() ERC20("ZYN", "ZYN") {
    }
    function mint(address reciever, uint256 amount) public
onlyOwner {
        _mint(reciever, amount);
    }
    function _burn(uint256 amount) public onlyOwner {
        burn(amount);
    }
}
```

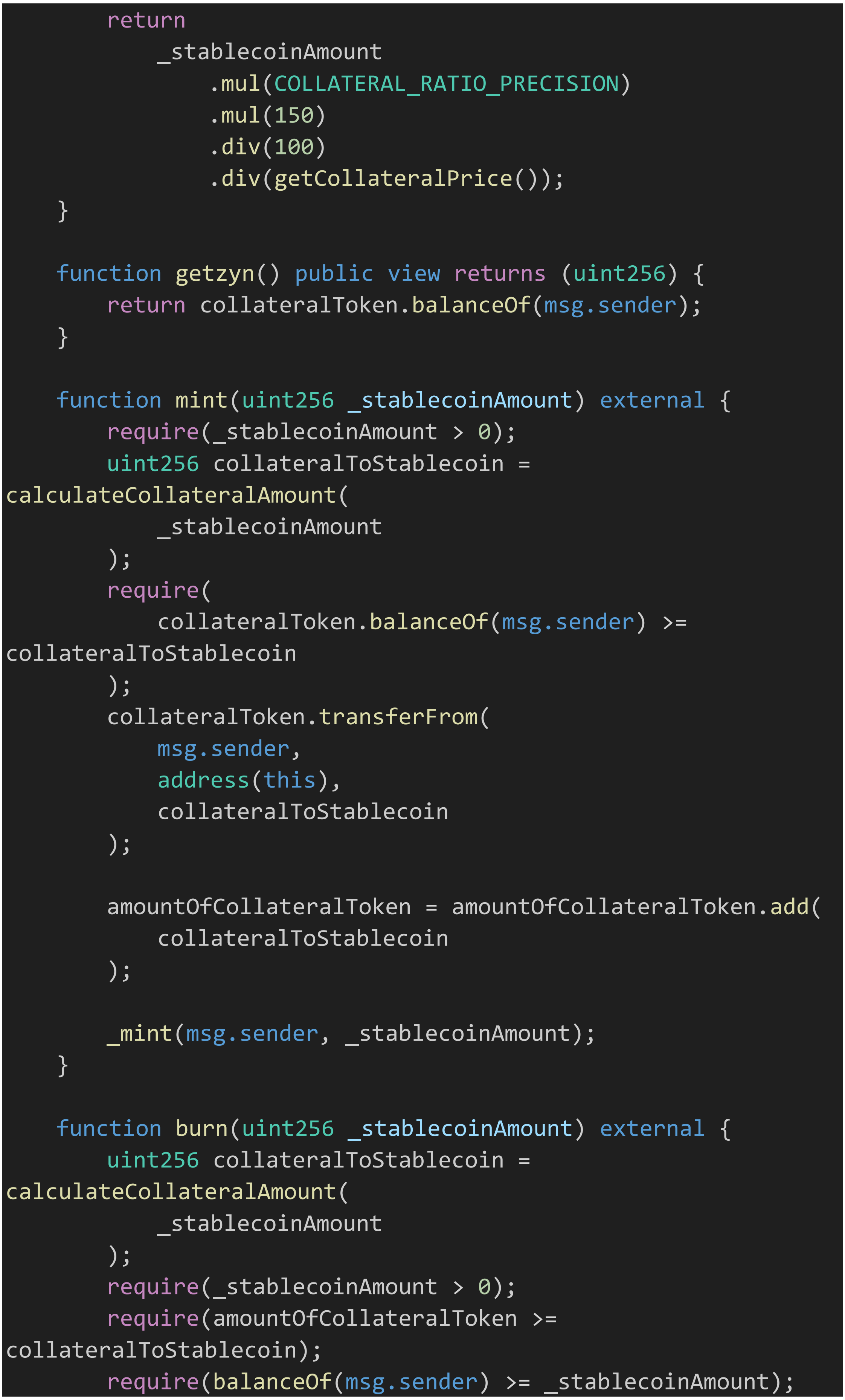
Pricefeed.sol

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract PriceFeed {
    constructor() {}

    /**
     * Returns the latest price.
     */
    // 抵押品的价格，比如 1 ZYN = 2 USD
    function getLatestPrice() public pure returns (int price) {
        return 2 * 1e18;
    }
}
```

02_CollateralStableCoin_start.sol



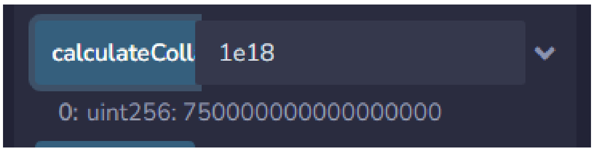
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

import "@openzeppelin/contracts/token/ERC20/ERC20.sol";
import "@openzeppelin/contracts/token/ERC20/IERC20.sol";
import "@openzeppelin/contracts/utils/math/SafeMath.sol";
import "./defi-practices/PriceFeed.sol";
import "./defi-practices/ZYN.sol";

// 此处补全

contract CollateralStableCoin is ERC20 {
    using SafeMath for uint256;

    IERC20 public collateralToken; // 要抵押的币 ZYN
    PriceFeed public priceFeed; // 价格预言机 返回当前 token 的价格
    uint256 public amountOfCollateralToken; // 抵押币的总量
    uint256 public constant COLLATERAL_RATIO_PRECISION = 1e18;

    constructor(
        address _collateralToken,
        address _priceFeed
    ) ERC20("DAI", "DAI") {
        collateralToken = IERC20(_collateralToken);
        priceFeed = PriceFeed(_priceFeed);
    }

    function getCollateralPrice() public view returns (uint256) {
        return uint256(priceFeed.getLatestPrice());
    }

    function calculateCollateralAmount(
        uint256 _stablecoinAmount
    ) public view returns (uint256) {
        // 150% 超额抵押 得到换_stablecoinAmount 个稳定币需要抵押的币

        // uint256*getCollateralPrice().mul(100).div(150)==_stablecoinAmount;
```



```
        return
            _stablecoinAmount
                .mul(COLLATERAL_RATIO_PRECISION)
                .mul(150)
                .div(100)
                .div(getCollateralPrice());
    }

    function getzyn() public view returns (uint256) {
        return collateralToken.balanceOf(msg.sender);
    }

    function mint(uint256 _stablecoinAmount) external {
        require(_stablecoinAmount > 0);
        uint256 collateralToStablecoin =
calculateCollateralAmount(
            _stablecoinAmount
        );
        require(
            collateralToken.balanceOf(msg.sender) >=
collateralToStablecoin
        );
        collateralToken.transferFrom(
            msg.sender,
            address(this),
            collateralToStablecoin
        );

        amountOfCollateralToken = amountOfCollateralToken.add(
            collateralToStablecoin
        );

        _mint(msg.sender, _stablecoinAmount);
    }

    function burn(uint256 _stablecoinAmount) external {
        uint256 collateralToStablecoin =
calculateCollateralAmount(
            _stablecoinAmount
        );
        require(_stablecoinAmount > 0);
        require(amountOfCollateralToken >=
collateralToStablecoin);
        require(balanceOf(msg.sender) >= _stablecoinAmount);
        collateralToken.transfer(msg.sender,
collateralToStablecoin);
        amountOfCollateralToken = amountOfCollateralToken.sub(
            collateralToStablecoin
        );
        _burn(msg.sender, _stablecoinAmount);
    }
}
```

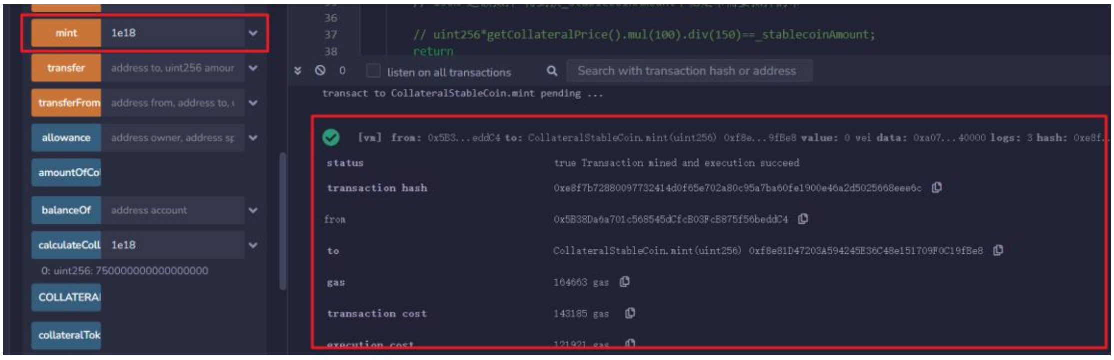
实验过程

1. 获取一个 dai 计算抵押数量



```
0: uint256: 750000000000000000
```

2. 给自己账户 mint 一个 dai

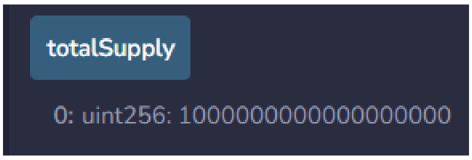

mint
1e18
transfer
address to, uint256 amour
transferFrom
address from, address to, u
allowance
address owner, address sp
amountOfCol
balanceOf
address account
calculateColl
1e18
COLLATERA
collateralTok
listen on all transactions
Search with transaction hash or address

```
0: uint256: 750000000000000000
```

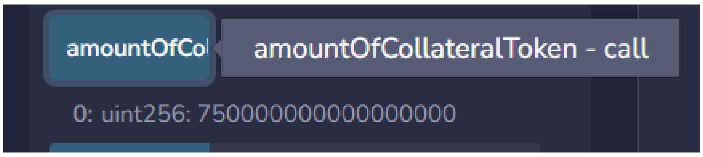
```
// uint256*getCollateralPrice().mul(100).div(150)==_stablecoinAmount;
return
```

```
transact to CollateralStableCoin.mint pending ...

[vm] from: 0x5B3...eddC4 to: CollateralStableCoin.mint(uint256) 0xf8e...9fBe8 value: 0 wei data: 0xa07...40000 logs: 3 hash: 0xe8f...
status              true Transaction mined and execution succeed
transaction hash    0xe8f7b72880097732414d0f65e702a80c95a7ba60fe1900e46a2d5025668eee6c
from                0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                  CollateralStableCoin.mint(uint256) 0xf8e81D47203A594245E36C48e151709F0C19fBe8
gas                 164663 gas
transaction cost    143185 gas
execution cost      121921 gas
```

3.

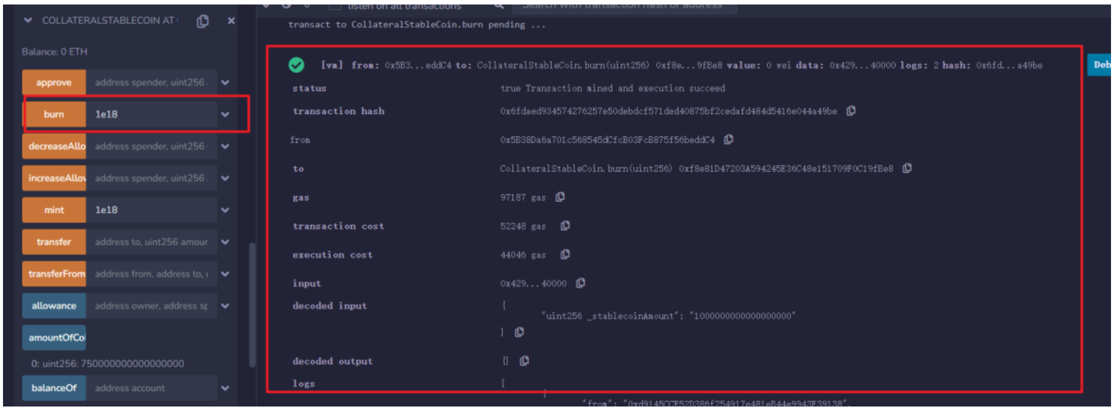


```
0: uint256: 1000000000000000000
```



```
0: uint256: 750000000000000000
```

4. 销毁一个 dai



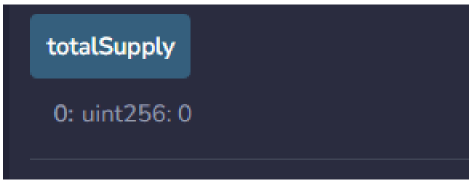
burn 1e18

mint 1e18

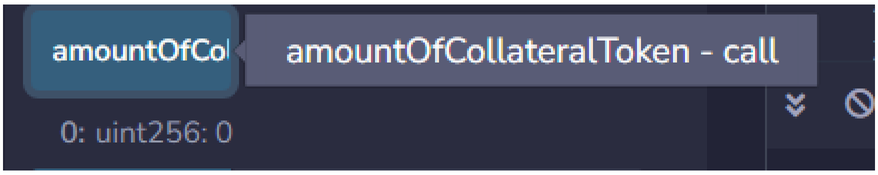
```
0: uint256: 750000000000000000
```

```
transact to CollateralStableCoin.burn pending ...

[vm] from: 0x5B3...eddC4 to: CollateralStableCoin.burn(uint256) 0xf8e...9fBe8 value: 0 wei data: 0x429...40000 logs: 2 hash: 0x6fd...a49be
status                  true Transaction mined and execution succeed
transaction hash        0x6fdaed934574276257e50debdcf571ded40875bf2cedafd484d5416e044a49be
from                    0x5B38Da6a701c568545dCfcB03FcB875f56beddC4
to                      CollateralStableCoin.burn(uint256) 0xf8e81D47203A594245E36C48e151709F0C19fBe8
gas                     97187 gas
transaction cost        52248 gas
execution cost          44046 gas
input                   0x429...40000
decoded input           {
                                "uint256 _stablecoinAmount": "1000000000000000000"
                        }
decoded output          {}
logs                    [
                                {
                                        "from": "0xd9145CCE52D386f254917e481eB44e9943F39138",
```

5.



```
0: uint256: 0
```



```
0: uint256: 0
```