Sobre o Bribe

Uma coisa que eu percebi do bribe é que só de colocar algumas variáveis como `private` no contrato do Schelling já torna minha vida mais difícil no contrato de suborno. Por exemplo, eu queria checar se o endereço que está requisitando o endereço do suborno era um participante do "Schelling" e qual o valor que ele votou. Mudei algumas variáveis do "Schelling" para public por causa disso, além de ter adicionado uma função que permite checar o valor do commit de um usuário após ele ter sido revelado. Isso foi suficiente para fazer o "Bribe" funcionar.

Abaixo, o código do contrato de suborno. Endereços aceitam um suborno na função `acceptBribe()` e um valor que será distribuído igualmente entre subornados é alocado no deploy do contrato. Também é escolhido no deploy qual será o comportamento escolhido para subornar, por exemplo, se eu quero subornar que votem "yes" então tem que colocar o valor 0 na variável `_toBribe`.

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

import "./Schelling.sol";
import "hardhat/console.sol";

contract Bribe {

    Schelling private sch;
    uint256 public toReceiveBribe;
    uint256 public bribe;
    mapping(address => bool) private participants;
    uint256 public numParticipants;

    enum RevealingState {
        waiting,
        canReveal,
        finished
    }

    constructor(address _schellingAddress, uint256 _toBribe) payable {
        sch = Schelling(_schellingAddress);
        toReceiveBribe = _toBribe;
        bribe = msg.value;
    }

    function acceptBribe() public {
        participants[msg.sender] = true;
        numParticipants += 1;
    }

    function claimBribe() public {
        require(uint256(sch.showCurrentState()) == 2, "not revealing state");
        uint256 value = sch.geCommitValue(msg.sender);
        if (value == toReceiveBribe) {
            uint256 sendValue = bribe / numParticipants;
```

```
            payable(msg.sender).transfer(sendValue);
        } else {
            revert("you did not act accordingly");
        }
    }
}
```