03_Blockchain - create a smart contract

Code:

```
// Hadcoins ICO

// Version of the compiler
pragma solidity >=0.4.22 <0.7.0;

contract hadcoin_ico {

    // Introducing the max number of Hadcoins available for sale
    uint public max_hadcoins=1000000;

    // Introducing the USD to Hadcoins conversion relocatable
    uint public usd_to_hadcoins=1000;

    // Introducing the total number of Hadcoins that have been bought by the investors
    uint public total_hadcoins_bought = 0;

    // Mapping from the investor address to its equity in Hadcoins and usd_to_hadcoins
    mapping(address => uint) equity_hadcoins;
    mapping(address => uint) equity_usd;

    // Checking if an investor can buy Hadcoins
    modifier can_buy_hadcoins(uint usd_invested){
        require(usd_invested * usd_to_hadcoins + total_hadcoins_bought <= max_hadcoins);
        _;
    }

    // Getting the equity in Hadcoins of an investor
    function equity_in_hadcoins(address investor) external constant returns (uint){
        return equity_hadcoins[investor];
    }

    // Getting the equity in USD of an investor
    function equity_in_usd(address investor) external constant returns (uint){
        return equity_usd[investor];
    }

    // Buying Hadcoins
    function buy_hadcoins(address investor, uint usd_invested) external can_buy_hadcoins(usd_invested){
        uint hadcoins_bought = usd_invested * usd_to_hadcoins;
        equity_hadcoins[investor] += hadcoins_bought;
        equity_usd[investor] = equity_hadcoins[investor] / 1000;
        total_hadcoins_bought += hadcoins_bought;
    }

    // Selling Hadcoins
    function sell_hadcoins(address investor, uint hadcoins_sold) external {
        equity_hadcoins[investor] -= hadcoins_sold;
        equity_usd[investor] = equity_hadcoins[investor] / 1000;
        total_hadcoins_bought -= hadcoins_sold;
    }
}
```