```
pragma solidity ^0.5.0;

// ----------------------------------------------------------------------------
// BEP Token Standard #20 Interface
//
// ----------------------------------------------------------------------------
contract BEP20Interface {
    function totalSupply() public view returns (uint);
    function balanceOf(address tokenOwner) public view returns (uint balance);
    function allowance(address tokenOwner, address spender) public view returns (uint
remaining);
    function transfer(address to, uint tokens) public returns (bool success);
    function approve(address spender, uint tokens) public returns (bool success);
    function transferFrom(address from, address to, uint tokens) public returns (bool success);

    event Transfer(address indexed from, address indexed to, uint tokens);
    event Approval(address indexed tokenOwner, address indexed spender, uint tokens);
}

// ----------------------------------------------------------------------------
// Safe Math Library
// ----------------------------------------------------------------------------
contract SafeMath {
    function safeAdd(uint a, uint b) public pure returns (uint c) {
        c = a + b;
        require(c >= a);
    }
    function safeSub(uint a, uint b) public pure returns (uint c) {
        require(b <= a); c = a - b; } function safeMul(uint a, uint b) public pure returns (uint c) { c =
a * b; require(a == 0 || c / a == b); } function safeDiv(uint a, uint b) public pure returns (uint c) {
require(b > 0);
        c = a / b;
    }
}


contract SFC is BEP20Interface, SafeMath {
    string public name;
    string public symbol;
    uint8 public decimals; // 18 decimals is the strongly suggested default, avoid changing it

    uint256 public _totalSupply;

    mapping(address => uint) balances;
```

```
    mapping(address => mapping(address => uint)) allowed;

    /**
     * Constrctor function
     *
     * Initializes contract with initial supply tokens to the creator of the contract
     */
    constructor() public {
        name = "SafeCap Token";
        symbol = "SFC";
        decimals = 18;
        _totalSupply = 100000000000000000000000000;

        balances[msg.sender] = _totalSupply;
        emit Transfer(address(0), msg.sender, _totalSupply);
    }

    function totalSupply() public view returns (uint) {
        return _totalSupply  - balances[address(0)];
    }

    function balanceOf(address tokenOwner) public view returns (uint balance) {
        return balances[tokenOwner];
    }

    function allowance(address tokenOwner, address spender) public view returns (uint
remaining) {
        return allowed[tokenOwner][spender];
    }

    function approve(address spender, uint tokens) public returns (bool success) {
        allowed[msg.sender][spender] = tokens;
        emit Approval(msg.sender, spender, tokens);
        return true;
    }

    function transfer(address to, uint tokens) public returns (bool success) {
        balances[msg.sender] = safeSub(balances[msg.sender], tokens);
        balances[to] = safeAdd(balances[to], tokens);
        emit Transfer(msg.sender, to, tokens);
        return true;
    }

    function transferFrom(address from, address to, uint tokens) public returns (bool success) {
```

```
        balances[from] = safeSub(balances[from], tokens);
        allowed[from][msg.sender] = safeSub(allowed[from][msg.sender], tokens);
        balances[to] = safeAdd(balances[to], tokens);
        emit Transfer(from, to, tokens);
        return true;
    }
}
```