Day18

pragma solidity ^0.4.25;

library SafeMath {

function mul(uint256 a, uint256 b) internal pure returns (uint256) {

uint256 c = a \* b;

require(c / a == b);

return c;

}

function div(uint256 a, uint256 b) internal pure returns (uint256) {

require(b > 0); // Solidity only automatically asserts when dividing by 0

uint256 c = a / b;

return c;

}

function sub(uint256 a, uint256 b) internal pure returns (uint256) {

require(b <= a); // underflow

uint256 c = a - b;

return c;

}

function add(uint256 a, uint256 b) internal pure returns (uint256) {

uint256 c = a + b;

require(c >= a); // overflow

return c;

}

function mod(uint256 a, uint256 b) internal pure returns (uint256) {

require(b != 0);

return a % b;

}

}

contract Main {

function test() public pure returns (uint256) {

uint256 a = 100;

uint256 b = 10;

return SafeMath.add(a, b);

}

}