pragma solidity ^0.5.0;

// lvl 1: equal split

contract AssociateProfitSplitter {

// @TODO: Create three payable addresses representing `employee\_one`, `employee\_two` and `employee\_three`.

address payable employee\_one;

address payable employee\_two;

address payable employee\_three;

constructor(address payable \_one, address payable \_two, address payable \_three) public {

employee\_one = \_one;

employee\_two = \_two;

employee\_three = \_three;

}

function balance() public view returns(uint) {

return address(this).balance;

}

function deposit() public payable {

// @TODO: Split `msg.value` into three

uint amount = msg.value/3; // Your code here!

// @TODO: Transfer the amount to each employee

// Your code here!

employee\_one.transfer(amount);

employee\_two.transfer(amount);

employee\_three.transfer(amount);

// @TODO: take care of a potential remainder by sending back to HR (`msg.sender`)

// Your code here!

msg.sender.transfer(msg.value-amount\*3);

}

function() external payable {deposit();

// @TODO: Enforce that the `deposit` function is called in the fallback function!

// Your code here!

}

}