pragma solidity ^0.5.0;

// lvl 3: equity plan

contract DeferredEquityPlan {

address human\_resources;

address payable employee; // bob

bool active = true; // this employee is active at the start of the contract

// @TODO: Set the total shares and annual distribution

// Your code here!

uint total\_shares = 1000;

uint annual\_distribution = 250;

uint start\_time = now; // permanently store the time this contract was initialized

// @TODO: Set the `unlock\_time` to be 365 days from now

// Your code here!

uint unlock\_time = now + 365 days;

uint public distributed\_shares; // starts at 0

constructor(address payable \_employee) public {

human\_resources = msg.sender;

employee = \_employee;

}

function distribute() public {

require(msg.sender == human\_resources || msg.sender == employee, "You are not authorized to execute this contract.");

require(active == true, "Contract not active.");

// @TODO: Add "require" statements to enforce that:

// 1: `unlock\_time` is less than or equal to `now`

// 2: `distributed\_shares` is less than the `total\_shares`

// Your code here!

require (unlock\_time <= now);

require (distributed\_shares < total\_shares);

// @TODO: Add 365 days to the `unlock\_time`

// Your code here!

unlock\_time += 365 days;

// @TODO: Calculate the shares distributed by using the function (now - start\_time) / 365 days \* the annual distribution

// Make sure to include the parenthesis around (now - start\_time) to get accurate results!

// Your code here!

distributed\_shares = (now - start\_time) / 365 days \* annual\_distribution;

// double check in case the employee does not cash out until after 5+ years

if(distributed\_shares > 1000) {

distributed\_shares = 1000;

}

}

// human\_resources and the employee can deactivate this contract at-will

function deactivate() public {

require(msg.sender == human\_resources || msg.sender == employee, "You are not authorized to deactivate this contract.");

active = false;

}

// Since we do not need to handle Ether in this contract, revert any Ether sent to the contract directly

function() external payable {

revert("Do not send Ether to this contract!");

}

}