pragma solidity >=0.5.0;

contract Owned{

address owner;

constructor() public {

owner = msg.sender;

}

modifier onlyOwner{

require(msg.sender == owner, "You must be owner");

\_;

}

}

contract Certificate is Owned{

struct Holder{

string name;

uint level;

}

mapping (address => Holder) holders;

address[] public holderAccts;

event HolderInfo(string name, uint level);

modifier checkLevel(uint \_level){

require(\_level <= 3, "You input wrong level");

\_;

}

function setHolder(address \_address, string memory \_name, uint \_level) onlyOwner checkLevel(\_level) public{

holders[\_address].name = \_name;

holders[\_address].level = \_level;

holderAccts.push(\_address);

emit HolderInfo(\_name, \_level);

}

function getHolders() view public returns(address[] memory){

return holderAccts;

}

function getHolder(address \_address) view public returns(string memory,uint){

return(holders[\_address].name, holders[\_address].level);

}

function countHolders() view public returns (uint){

return holderAccts.length;

}

}