pragma solidity ^0.6.0;

contract Bank

{

mapping(address=> uint ) private \_balances;

address public owner;

event LogDepositeMade(address accountHoder, uint amount );

constructor () public

{

owner=msg.sender;

emit LogDepositeMade(msg.sender, 1000);

}

function deposite() public payable returns (uint)

{

require ((\_balances[msg.sender] + msg.value) > \_balances[msg.sender] && msg.sender!=address(0));

\_balances[msg.sender] += msg.value;

emit LogDepositeMade(msg.sender , msg.value);

return \_balances[msg.sender];

}

function withdraw (uint withdrawAmount) public returns (uint)

{

require (\_balances[msg.sender] >= withdrawAmount);

require(msg.sender!=address(0));

require (\_balances[msg.sender] > 0);

\_balances[msg.sender]-= withdrawAmount;

msg.sender.transfer(withdrawAmount);

emit LogDepositeMade(msg.sender , withdrawAmount);

return \_balances[msg.sender];

}

function viewBalance() public view returns (uint)

{

return \_balances[msg.sender];

}

}