pragma solidity ^0.4.19;

contract Auction {

address public auctioneer;

address public seller;

uint public latestBid;

address public latestBidder;

uint public previousBid;

uint public count;

constructor() public {

auctioneer = msg.sender;

count = 0;

}

function auction(uint bid) public {

latestBid = bid \* 1 ether; //1000000000000000000;

seller = msg.sender;

}

function bid() public payable {

require(msg.value > latestBid);

if (latestBidder != 0x0) {

latestBidder.transfer(latestBid);

}

previousBid = latestBid;

latestBidder = msg.sender;

latestBid = msg.value;

count = count + 1;

//latestBid-previousBid == returnAmount;

}

function finishAuction() restricted public {

if(count == 1){

seller.transfer(previousBid);

}

else{

seller.transfer(previousBid);

latestBidder.transfer(address(this).balance);

}

}

modifier restricted() {

require(msg.sender == auctioneer);

\_;

}

}