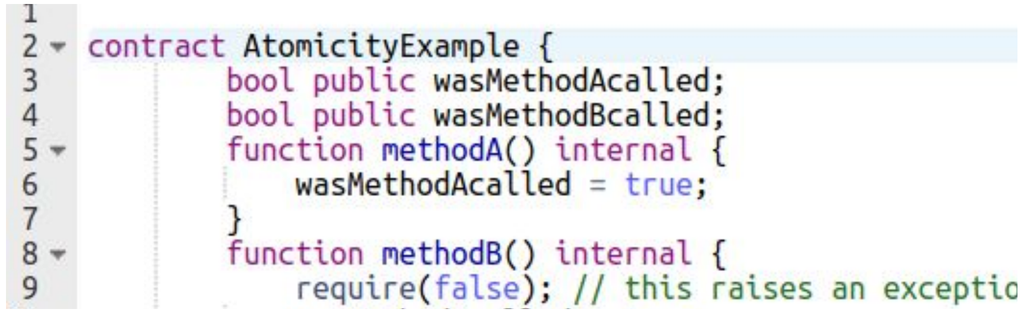
Lecture Follow-up: Smart Contracts

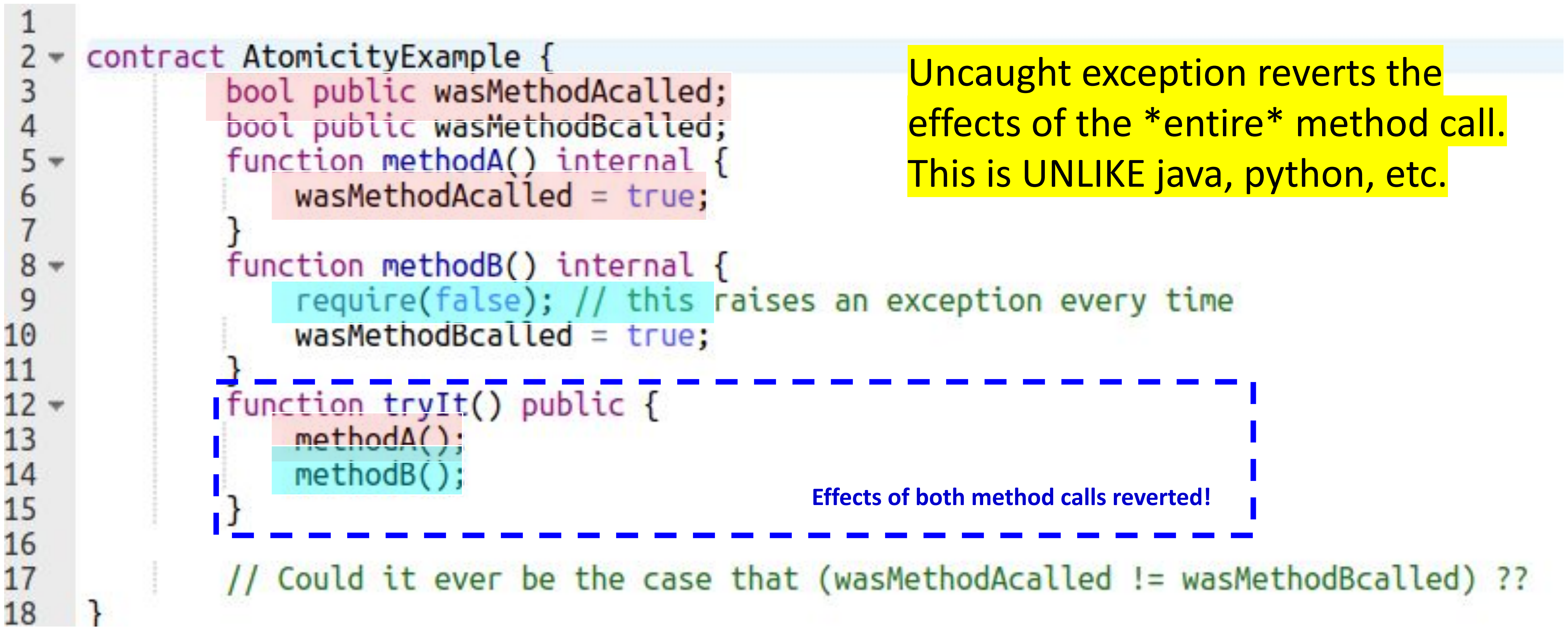
1. Atomicity in smart contracts



```

contract AtomicityExample {
        bool public wasMethodAcalled;
        bool public wasMethodBcalled;
        function methodA() internal {
            wasMethodAcalled = true;
        }
        function methodB() internal {
            require(false); // this raises an exceptio
```

2. Dutch auction contracts as a legal contract

Atomicity in Smart Contracts

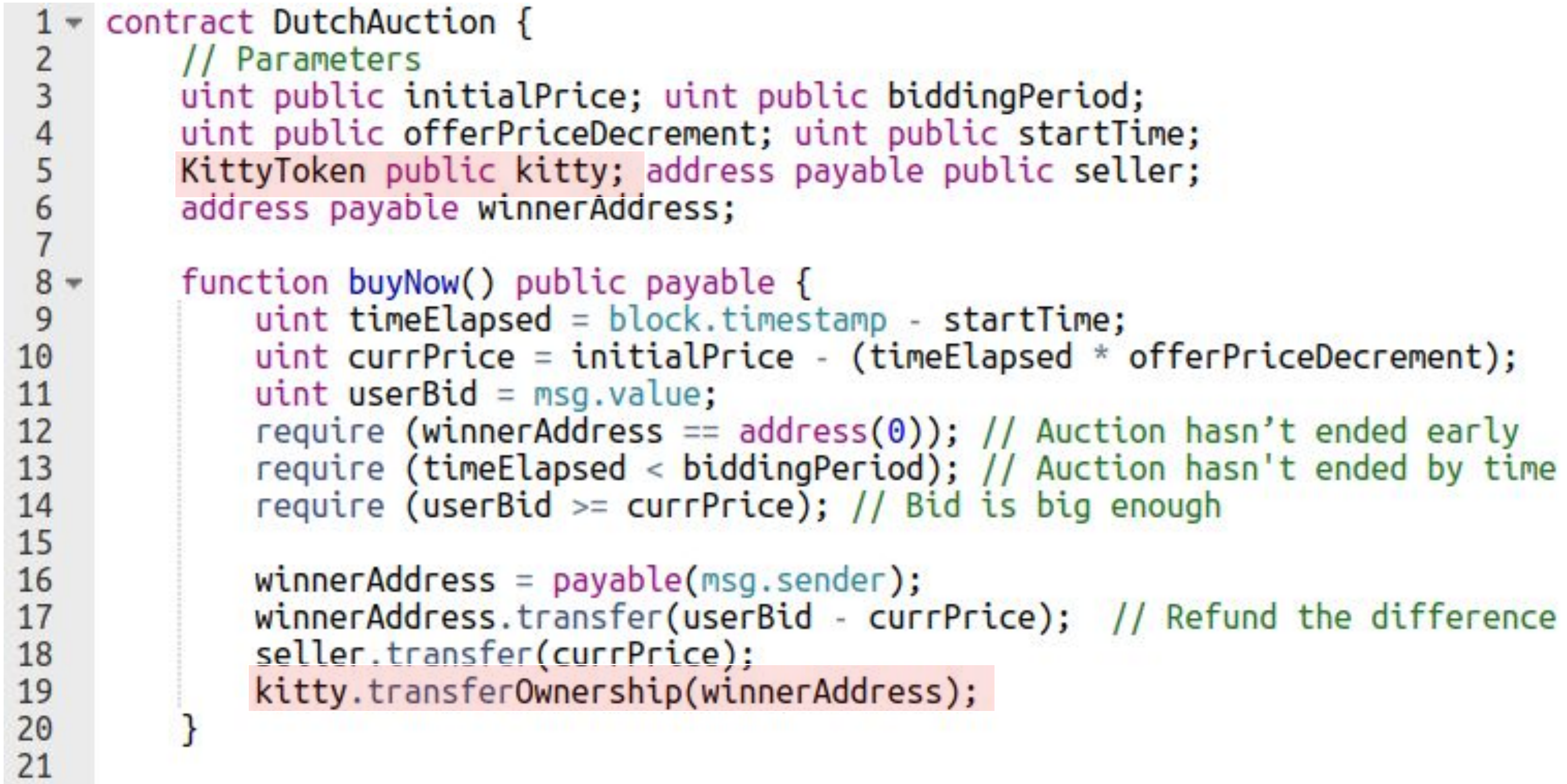


```

contract AtomicityExample {
        bool public wasMethodAcalled;
        bool public wasMethodBcalled;
        function methodA() internal {
            wasMethodAcalled = true;
        }
        function methodB() internal {
            require(false); // this raises an exception every time
            wasMethodBcalled = true;
        }
        function tryIt() public {
            methodA();
            methodB();
        }

        // Could it ever be the case that (wasMethodAcalled != wasMethodBcalled) ??
}
```

Uncaught exception reverts the effects of the *entire* method call. This is UNLIKE java, python, etc.

Effects of both method calls reverted!

Dutch Auction as a legal contract?



```
contract DutchAuction {
    // Parameters
    uint public initialPrice; uint public biddingPeriod;
    uint public offerPriceDecrement; uint public startTime;
    KittyToken public kitty; address payable public seller;
    address payable winnerAddress;

    function buyNow() public payable {
        uint timeElapsed = block.timestamp - startTime;
        uint currPrice = initialPrice - (timeElapsed * offerPriceDecrement);
        uint userBid = msg.value;
        require (winnerAddress == address(0)); // Auction hasn’t ended early
        require (timeElapsed < biddingPeriod); // Auction hasn't ended by time
        require (userBid >= currPrice); // Bid is big enough

        winnerAddress = payable(msg.sender);
        winnerAddress.transfer(userBid - currPrice);  // Refund the difference
        seller.transfer(currPrice);
        kitty.transferOwnership(winnerAddress);
    }

```