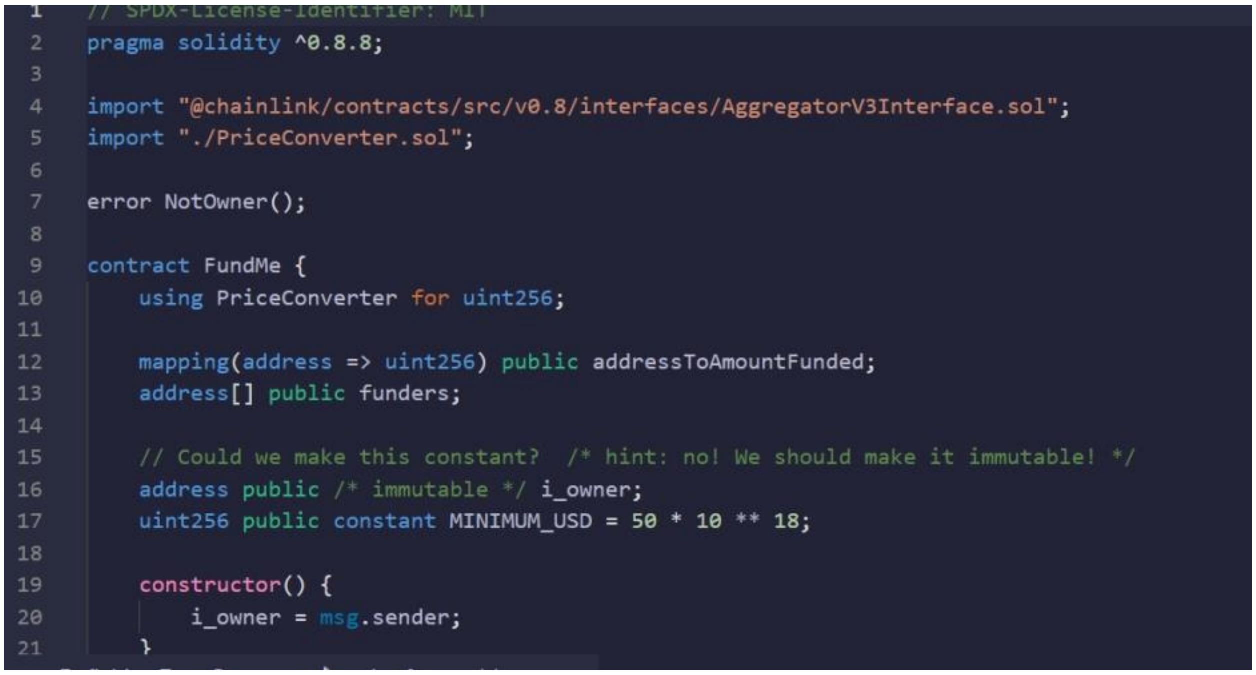
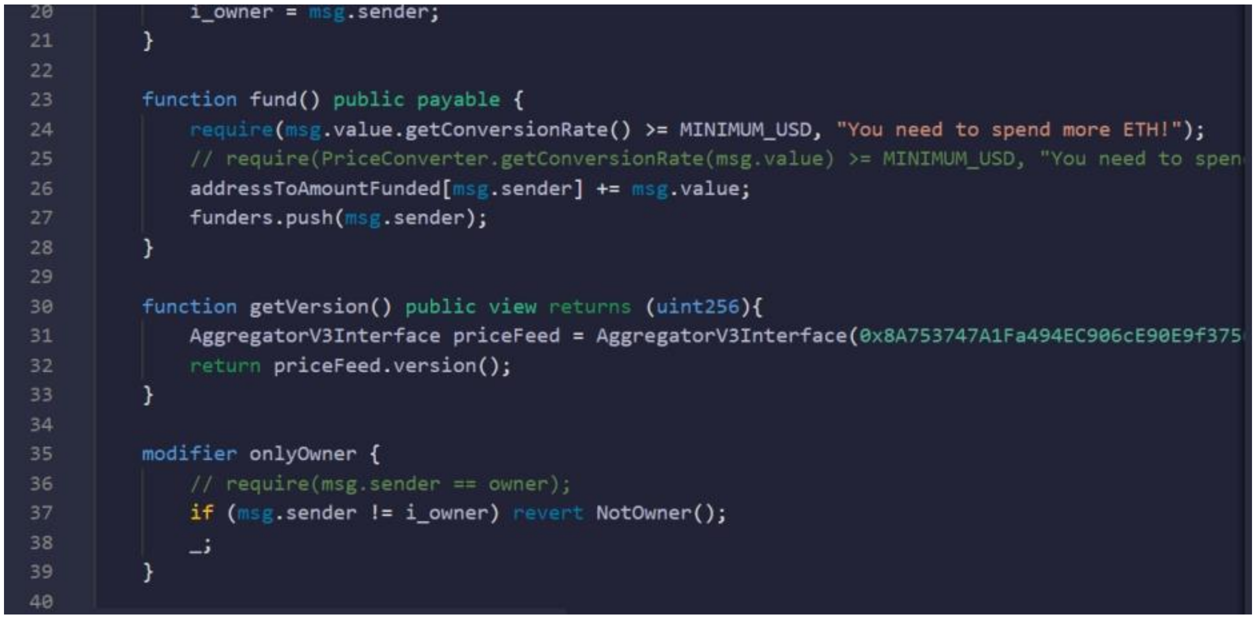
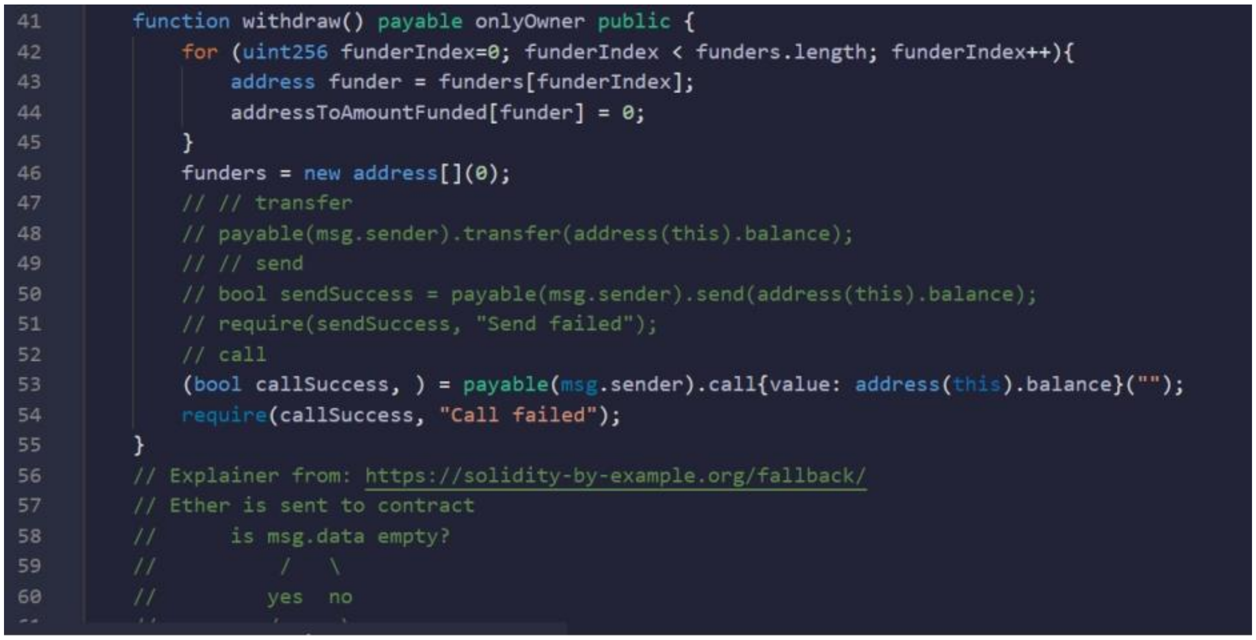
Remix Fund Me



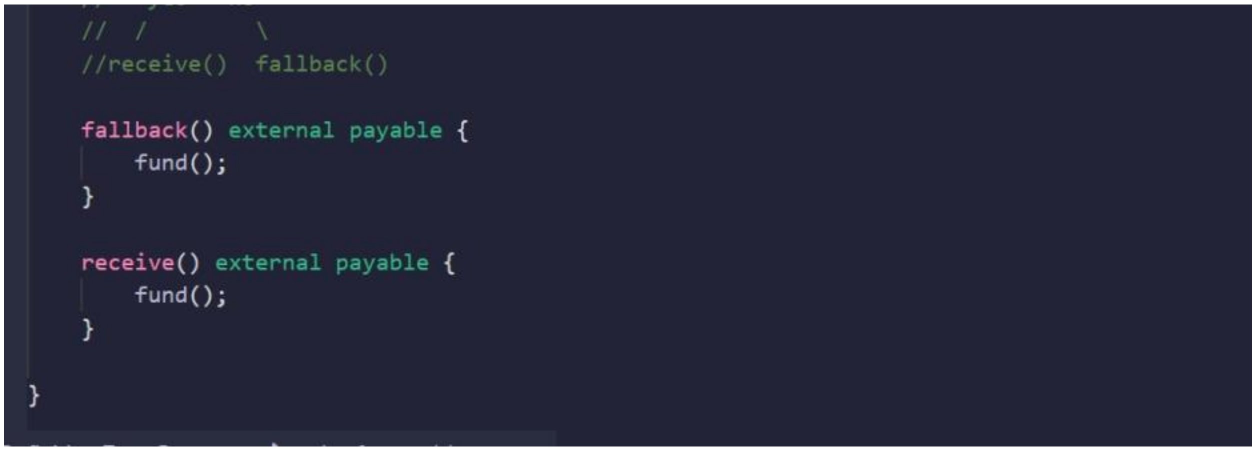
```solidity
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.8;

import "@chainlink/contracts/src/v0.8/interfaces/AggregatorV3Interface.sol";
import "./PriceConverter.sol";

error NotOwner();

contract FundMe {
    using PriceConverter for uint256;

    mapping(address => uint256) public addressToAmountFunded;
    address[] public funders;

    // Could we make this constant?  /* hint: no! We should make it immutable! */
    address public /* immutable */ i_owner;
    uint256 public constant MINIMUM_USD = 50 * 10 ** 18;

    constructor() {
        i_owner = msg.sender;
    }
```



```solidity
        i_owner = msg.sender;
    }

    function fund() public payable {
        require(msg.value.getConversionRate() >= MINIMUM_USD, "You need to spend more ETH!");
        // require(PriceConverter.getConversionRate(msg.value) >= MINIMUM_USD, "You need to spen
        addressToAmountFunded[msg.sender] += msg.value;
        funders.push(msg.sender);
    }

    function getVersion() public view returns (uint256){
        AggregatorV3Interface priceFeed = AggregatorV3Interface(0x8A753747A1Fa494EC906cE90E9f375
        return priceFeed.version();
    }

    modifier onlyOwner {
        // require(msg.sender == owner);
        if (msg.sender != i_owner) revert NotOwner();
        _;
    }

```



```
    function withdraw() payable onlyOwner public {
        for (uint256 funderIndex=0; funderIndex < funders.length; funderIndex++){
            address funder = funders[funderIndex];
            addressToAmountFunded[funder] = 0;
        }
        funders = new address[](0);
        // // transfer
        // payable(msg.sender).transfer(address(this).balance);
        // // send
        // bool sendSuccess = payable(msg.sender).send(address(this).balance);
        // require(sendSuccess, "Send failed");
        // call
        (bool callSuccess, ) = payable(msg.sender).call{value: address(this).balance}("");
        require(callSuccess, "Call failed");
    }
    // Explainer from: https://solidity-by-example.org/fallback/
    // Ether is sent to contract
    //      is msg.data empty?
    //          /   \
    //         yes  no
```



```
    //  /        \
    //receive()  fallback()

    fallback() external payable {
        fund();
    }

    receive() external payable {
        fund();
    }

}
```