Practical 4: Balance transfer between blocks

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract BalanceTransfer {
    // Keep track of balances for users inside the contract
    mapping(address => uint256) public balances;

    // Deposit ether into contract
    function deposit() public payable {
        balances[msg.sender] += msg.value;
    }

    // Transfer balance from sender to another address
    function transfer(address payable _to, uint256 _amount) public {
        require(balances[msg.sender] >= _amount, "Insufficient balance");

        // Deduct from sender
        balances[msg.sender] -= _amount;

        // Add to receiver
        balances[_to] += _amount;
    }

    // Withdraw ether from contract
    function withdraw(uint256 _amount) public {
        require(balances[msg.sender] >= _amount, "Not enough balance");

        balances[msg.sender] -= _amount;
        payable(msg.sender).transfer(_amount);
    }

    // Check contract balance
    function getContractBalance() public view returns (uint256) {
        return address(this).balance;
    }
}
```