

# Smart Contracts - Aposta Par-Ímpar

**Aluno:** Danilo Augusto Nunes - 11611BCC021

Na implementação da atividade foi utilizado o Brownie, além disso, a biblioteca de "Commit" disponibilizada pelo professor.

O jogo funciona da seguinte forma, tanto o dono quanto o jogador apostam com um número, que ficará escondido no contrato pelo protocolo de commit, depois de ambos jogarem, os números são revelados, após serem somados, através do resto por 2, são classificados em par ou ímpar, dessa forma, resultado 0 garante que o dono do contrato venceu, consequentemente, o jogador ganha com resultado 1.

A punição do jogo é para o participante que não revela o commit dentro do limite estipulado, no caso, uma quantidade de blocos definida. Sendo assim, o outro participante vence e fica com o dinheiro. Além disso, caso apenas um revele, e mesmo assim, esse valor não esteja correto, ele vence a aposta, já que o outro participante foi punido por não revelar. Caso os dois não revelem, o dinheiro morre no contrato.

## Contrato: ParImpar

```
pragma solidity >=0.4.25 <0.6.0;

import "./SimpleCommit.sol";

contract parImpar {

    using SimpleCommit for SimpleCommit.CommitType;

    enum StatesType {aguardaJogador, revelaValores, revelado, fim}

    SimpleCommit.CommitType valorDono;
    SimpleCommit.CommitType valorJogador;

    address payable dono;
    address payable jogador;

    uint256 valorAposta;
    uint256 limiteBloco;

    StatesType myState;

    // saber se os jogadores já revelaram o commit
    bool donoRevealed = false;
    bool jogadorRevealed = false;
```

```

constructor (bytes32 _vD) public payable {
    valorDono.commit(_vD);
    valorAposta = msg.value;
    limiteBloco = block.number + 10;
    dono = msg.sender;
    myState = StatesType.aguardaJogador;
}

function entraJogo(bytes32 _vJ) public payable {
    // Dono precisa ter iniciado o jogo
    require (myState == StatesType.aguardaJogador);
    require (msg.value >= valorAposta, "Aposta tem valor minimo!!!");

    // atualiza estado do contrato
    valorJogador.commit(_vJ);
    jogador = msg.sender;

    // armazena _vJ e msg.sender
    myState = StatesType.revelaValores;
}

function donoRevela(byte value, bytes32 nonce) public {
    require (msg.sender == dono);
    require (myState == StatesType.revelaValores);

    valorDono.reveal(nonce,value);
    donoRevealed = true;
}

function jogadorRevela(byte value, bytes32 nonce) public {
    require (msg.sender == jogador);
    require (myState == StatesType.revelaValores);

    valorJogador.reveal(nonce,value);
    jogadorRevealed = true;
}

function pagaVencedor() public {
    // jogadores devem pelo menos ter iniciado o jogo (ambos)
    // Não necessariamente terem revelado o commit
    require (myState == StatesType.revelaValores);
}

```

```

// bloco passou do limite
if (block.number > limiteBloco) {
    // Encerra o jogo
    myState = StatesType.fim;

    // jogador não revelou
    if (donoRevealed && !jogadorRevealed){
        dono.transfer(address(this).balance);
    }
    // dono não revelou
    else if (!donoRevealed && jogadorRevealed){
        jogador.transfer(address(this).balance);
    }
}
else {
    // jogadores precisam ter revelado os commits
    require (donoRevealed && jogadorRevealed);

    // Encerra o jogo
    myState = StatesType.fim;

    // dono não revelou corretamente
    if (!(valorDono.isCorrect()) && valorJogador.isCorrect()){
        jogador.transfer(address(this).balance);
    }
    // jogador não revelou corretamente
    else if (valorDono.isCorrect() && !(valorJogador.isCorrect())){
        dono.transfer(address(this).balance);
    }
    // ambos revelaram correto
    else if (valorDono.isCorrect() && valorJogador.isCorrect()){
        uint soma;
        soma = uint8(valorDono.getValue())+uint8(valorJogador.getValue());

        // dono ganha com par
        if (soma % 2 == 0){
            dono.transfer(address(this).balance);
        }
        // jogador ganha com impar
        else {
            jogador.transfer(address(this).balance);
        }
    }
}
}
}
}

```

Para os testes, foram utilizadas as funções de hash prontas disponibilizadas pelo professor no exemplo de Commit. Tais funções criam o commit concatenando o nonce ao valor.

```
#!/usr/bin/python3

from brownie import *
import brownie
import hashlib

def createNonce(s):
    n = hashlib.sha256()
    n.update(s)
    return n.digest()

def doCommit(n,v):
    c = hashlib.sha256()
    c.update(n)
    c.update(v)
    return c.digest()
```

Como forma de testes, foram criados diversos cenários que mostram o comportamento do contrato diante de diferentes situações.

accounts[0] é o endereço do dono.  
accounts[1] é o endereço do jogador.

## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)

    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)

    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)

    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"20 ether"})

    contrato.donoRevela(valor_dono,nonce_dono,{"from":dono})
    contrato.jogadorRevela(valor_jogador,nonce_jogador,{"from":jogador})


    contrato.pagaVencedor()
```

Ambos revelaram correto

Soma = 2 + 3 = 5

5 % 2 = 1

Jogador venceu

 Ganache

ACCOUNTS

BLOCKS

TRANSACTIONS

CONTRACTS

EVENTS

LOGS

CURRENT BLOCK 6	GAS PRICE 20000000000	GAS LIMIT 6721975	HARDFORK MUIRGLACIER	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:7545	MINING STATUS AUTOMINING
--------------------	--------------------------	----------------------	-------------------------	--------------------	-------------------------------------	-----------------------------

MNEMONIC ?  
nuclear account poet item clutch icon amused trumpet dentist damp practice rigid

ADDRESS	BALANCE
0x07651b882D69Dc234321BBf8c8535d32aDe885E4	90.00 ETH

ADDRESS	BALANCE
0x986591B68f85342dA59e3eadF131A8369807EAeA	110.00 ETH

## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)

    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)

    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)

    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"5 ether"})

    contrato.donoRevela(valor_dono,nonce_dono,{"from":dono})
    contrato.jogadorRevela(valor_jogador,nonce_jogador,{"from":jogador})

    contrato.pagaVencedor()
```

O jogador tentou fazer uma aposta de 5 ether, que é um valor menor do que o mínimo definido pelo dono. No caso, a aposta deveria ser maior ou igual a 10 ether.

```
Running '\Users\danilo\OneDrive\Documentos\Faculdade\2021-2\SEG\praticas\pratica 3\scripts\joga.py::main'...
Transaction sent: 0xc8b27221dbdf72552cfc51c970941ffc49c635909b2651ae4b636fd9ba7575a2
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
SimpleCommit.constructor confirmed Block: 1 Gas used: 230664 (3.43%)
SimpleCommit deployed at: 0x218B9Eb9e08F4C58DCD4a30ca407FCbF8d66656D

Transaction sent: 0xb599523b2810120f5d18c9e2a106ebcb065d1fa3d5754a82c71df95951d6b31f
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
parImpar.constructor confirmed Block: 2 Gas used: 698760 (10.40%)
parImpar deployed at: 0xeDBD8fBc4f1628F08e6fA1A1F2aB8B02cf70e5DF

Transaction sent: 0xa1b6677c7d73919b0ed26f2f925488de65212dcbaa72f89bd6bc5c498744bf73
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 0
parImpar.entraJogo confirmed (Aposta tem valor minimo!!!) Block: 3 Gas used: 23539 (0.35%)

Transaction sent: 0x76d6085994c7ec1b9f6781614a1fbb96879e25d65a91246d48a89f7c38b1e28c
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 2
parImpar.donoRevela confirmed (reverted) Block: 4 Gas used: 23639 (0.35%)

Transaction sent: 0xb04857394b9ec03864fa688e00fc927ccfe69d09b0dd5b1e635440b1ae59250a
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 1
parImpar.jogadorRevela confirmed (reverted) Block: 5 Gas used: 22774 (0.34%)

Transaction sent: 0x3d379b034dab7073626298862e31d01c42caffe2dd2d5e8da9a8e08dd29fd6f2
Gas price: 0.0 gwei Gas limit: 6721975 Nonce: 3
parImpar.pagaVencedor confirmed (reverted) Block: 6 Gas used: 22037 (0.33%)
```

## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)

    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)


    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)

    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"20 ether"})

    contrato.donoRevela(valor_dono,nonce_dono,{"from":dono})
    contrato.jogadorRevela(b'1',nonce_jogador,{"from":jogador})

    contrato.pagaVencedor()
```

Ambos revelaram o commit, mas o jogador revelou incorreto  
Nesse caso, dono ganhou

 Ganache

ACCOUNTS

BLOCKS


TRANSACTIONS

CONTRACTS

EVENTS

LOGS

CURRENT BLOCK	GAS PRICE	GAS LIMIT	HARDFORK	NETWORK ID	RPC SERVER	MINING STATUS
6	20000000000	6721975	MUIRGLACIER	5777	HTTP://127.0.0.1:7545	AUTOMINING

MNEMONIC 

victory indicate deputy text pudding list allow differ bag sunset gentle memory

ADDRESS

BALANCE

0x7C3C5d1D93b93F117F0E101e5685e12706CA578C

120.00 ETH

ADDRESS

BALANCE

0x9454cd119d581f8dFd4FCEe546B082821b8c6173

80.00 ETH

## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)


    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)


    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)


    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"20 ether"})


    for _ in range(20):
        contrato.pagaVencedor()
```


Nenhum dos jogadores revelaram o commit  
Os dois perderam dinheiro


 Ganache


 ACCOUNTS

 BLOCKS


 TRANSACTIONS

 CONTRACTS

 EVENTS

 LOGS

CURRENT BLOCK 14	GAS PRICE 20000000000	GAS LIMIT 6721975	HARDFORK MUIRGLACIER	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:7545	MINING STATUS AUTOMINING
---------------------	--------------------------	----------------------	-------------------------	--------------------	-------------------------------------	-----------------------------

MNEMONIC 

child paddle champion tip will cycle assist surprise victory laundry regular few

ADDRESS	BALANCE
0x258878e5d9112A0C830a48166f55Ca451aa83fEb	90.00 ETH

ADDRESS	BALANCE
0x3d79367BECC278B69Ab4aD45B72D6AD373095834	80.00 ETH



## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)

    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)


    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)


    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"20 ether"})


    contrato.donoRevela(valor_dono,nonce_dono,{"from":dono})


    for _ in range(20):
        contrato.pagaVencedor()
```


Somente o dono revelou o commit, ganhando a aposta


 Ganache


 **ACCOUNTS**

 BLOCKS


 TRANSACTIONS

 CONTRACTS

 EVENTS

 LOGS

CURRENT BLOCK 24	GAS PRICE 20000000000	GAS LIMIT 6721975	HARDFORK MUIRGLACIER	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:7545	MINING STATUS AUTOMINING
---------------------	--------------------------	----------------------	-------------------------	--------------------	-------------------------------------	-----------------------------

**MNEMONIC** 

dove silly fat electric ride silly avoid adapt scale choice humor bring

ADDRESS	BALANCE
0xe007BEfe6D88c9a87d8eD8602AE786572bB99747	120.00 ETH
ADDRESS	BALANCE
0x0850BAe962dC1A9Fac7d6482aF3F838ca186Dde6	80.00 ETH

## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)

    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)


    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)


    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"20 ether"})
    contrato.jogadorRevela(valor_jogador,nonce_jogador,{"from":jogador})

    for _ in range(20):
        contrato.pagaVencedor()


    contrato.donoRevela(valor_dono,nonce_dono,{"from":dono})
    contrato.pagaVencedor()
```

Dono quis revelar commit após o limite estipulado, nesse caso, o jogador já tinha ganhado.


 Ganache




ACCOUNTS




BLOCKS




TRANSACTIONS



CONTRACTS



EVENTS



LOGS

CURRENT BLOCK  
26

GAS PRICE  
20000000000

GAS LIMIT  
6721975

HARDFORK  
MUIRGLACIER

NETWORK ID  
5777

RPC SERVER  
HTTP://127.0.0.1:7545

MINING STATUS  
AUTOMINING

MNEMONIC ?

antenna ocean monster polar predict dinner reduce present stamp antique pass carpet

ADDRESS	BALANCE
0x89A70b67f29435a4963669B856CA89C323Db6749	90.00 ETH
ADDRESS	BALANCE
0xB744fFc7fEe982AF339e3cA978E0dCCF85524bB7	110.00 ETH

## Teste:

```
def main():
    # deploy da biblioteca de commit
    dono = accounts[0]
    SimpleCommit.deploy({'from':dono})

    valor_dono = b'2'
    nonce_dono = createNonce(b'nonce1')
    commit_dono = doCommit(nonce_dono,valor_dono)


    # deploy do jogo
    parImpar.deploy(commit_dono,{"from":dono, "value":"10 ether"})
    address = parImpar[0].address
    contrato = parImpar.at(address)


    jogador = accounts[1]
    valor_jogador = b'3'
    nonce_jogador = createNonce(b'nonce2')
    commit_jogador = doCommit(nonce_jogador,valor_jogador)


    contrato.entraJogo(commit_jogador,{"from":jogador,'value':"20 ether"})
    contrato.jogadorRevela(b'1',nonce_jogador,{"from":jogador})


    for _ in range(20):
        contrato.pagaVencedor()
```


Mesmo o jogador tendo revelado incorretamente o commit, ele ganha a aposta, já que o dono não revelou.


 Ganache


 **ACCOUNTS**

 **BLOCKS**


 **TRANSACTIONS**

 **CONTRACTS**

 **EVENTS**

 **LOGS**

CURRENT BLOCK 26	GAS PRICE 20000000000	GAS LIMIT 6721975	HARDFORK MUIRGLACIER	NETWORK ID 5777	RPC SERVER HTTP://127.0.0.1:7545	MINING STATUS AUTOMINING
---------------------	--------------------------	----------------------	-------------------------	--------------------	-------------------------------------	-----------------------------

**MNEMONIC**   
because layer bounce review vendor guess picnic caution domain vanish resist flame

ADDRESS	BALANCE
0x2b6f4E73bfa3d252C2C2176cC61176784AB2d529	90.00 ETH

ADDRESS	BALANCE
0x412b350e541A07540E1037be201Ab311cA01988d	110.00 ETH