



# UNIVERSIDADE DA CORUÑA

Facultad de Informática

## Práctica 1 – Ejercicio 6: Smart Contract de Lotería

Autor:

Héctor Núñez Fernández  
Fernando Antonio Fung Cen  
Santiago Novo Rodríguez

Profesora:

Paula Fraga Lamas

A Coruña, Noviembre 2023

## INDICE

|  |   |
|--|---|
| <b>1. Análisis y Definición del Escenario.</b>   | 3 |
| <b>2. Diseño.</b>  | 4 |
| a) Datos: dispondrá de los siguientes registros.   | 4 |
| b) Funciones: a nivel general contendrá las siguientes funciones.  | 4 |
| c) Reglas de operación: se tienen estipulados las siguientes reglas del negocio a aplicar en el contrato.                  | 4 |
| d) Usuarios del sistema: tiene contemplado los siguientes.   | 4 |
| e) Aspectos técnicos: características y puntos a tener en cuenta relacionado a la implementación del contrato inteligente. | 4 |
| <b>3. Implementación</b>   | 5 |
| <b>4. Pruebas</b>  | 6 |
| a) Creación del contrato de lotería.   | 6 |
| b) Prueba de validaciones: tratando de comprar un boleto para una lotería cuya fecha es la actual o un día de anterior.    | 6 |
| c) Prueba de compra de un boleto.  | 7 |
| d) Simulación de sorteo (se comenta la validación de estar en la fecha del sorteo acordado).                               | 7 |
| e) Consulta de usuario ganador.  | 8 |

## **1. Análisis y Definición del Escenario.**

El presente trabajo corresponde a un escenario de uso para implementar un sistema de gestión de una lotería en base a contrato inteligente en Ethereum. Actualmente los sistemas de loterías están basados en la confianza que representa la entidad que realiza el juego, y bajo las cuales los jugadores compran o siguen una serie de reglas establecidas y publicadas que se deben seguir para que no exista por un lado alguna forma de explotación de vulnerabilidad por parte de los jugadores y por otra algún abusado por parte de la entidad que sortea el juego.

En este contexto, normalmente el público que juega no tiene acceso directo explícito como tal al bote o los fondos que se publicitan para el juego, sino que tiene que confiar en la cantidad publicitada que es garantizada por un tercero que certifica al que realiza el juego, este aspecto y otros como la practica inexistencia de alguna documentación técnica y detallada de los mecanismos empleados en el juego no proveen una cierta seguridad más allá que el de la confianza para poder realizar el juego de manera justa. Razón por la cual pueden (y han ocurrido) hechos delictivos que han aprovechado estas vulnerabilidades haciendo de esto un problema.

A razón de lo planteado, se expone un sistema de loterías basado en contratos inteligentes que ofrezca una mejora en el aspecto de la seguridad la operación del mismo, y con ello aumentar la confianza por parte del público prospecto, al satisfacer las siguientes necesidades:

- a) Transparencia: en la visualización del monto actual de los fondos en el bote que se está jugando, los procesos y mecanismos para garantizar que la toma del número sea aleatorio e independiente de factores externos (algoritmo).
- b) Registro y anonimato de los jugadores: a fin de proteger la privacidad de información personal de los mismos a través de la exposición de los códigos de billeteras, garantizando a su vez que sea fidedigno por la compra que realizan usando criptomonedas.
- c) Seguridad: en la protección de registros de los jugadores para evitar la alteración ya sea por remoción o adición de jugadores por parte de atacantes externos, incluyendo la imposibilidad de modificación de la fecha del sorteo del mismo en la creación del contrato.

## 2. Diseño.

El contrato inteligente contendrá las siguientes especificaciones:

a) Datos: dispondrá de los siguientes registros.

Lista de participantes: array con las direcciones de todos los usuarios que compren un boleto, y pueden repetirse tantas veces como boletos han comprado.

Valor del premio acumulado: representa el bote de dinero acumulado y aumenta en función de los participantes que compren un boleto.

Fecha límite: fecha máxima para poder comprar boletos y a partir de esta fecha se podrá realizar el sorteo.

Administrador: contiene la dirección (address) del usuario que despliega el contrato, el único que podrá realizar el sorteo.

Ganador: contiene la dirección (address) del participante que ha ganado el sorteo.

b) Funciones: a nivel general contendrá las siguientes funciones.

ComprarBoleto: permite a un usuario comprar un boleto de lotería. Necesita enviar 1 Ether y que la fecha límite no haya pasado.

RealizarSorteo: realiza un sorteo aleatorio entre los participantes y obtiene un ganador entre todos ellos, sólo puede hacerla el administrador.

ObtenerGanador: devuelve la dirección del ganador del sorteo para consultas futuras.

c) Reglas de operación: se tienen estipulados las siguientes reglas del negocio a aplicar en el contrato.

Los boletos tienen un costo fijo en ether.

El sorteo se realiza una vez que se alcanza la fecha límite

El premio acumulado se distribuye al ganador.

Debe ser indicado la fecha en la que se realizara el sorteo.

La compra de boletos no es permitido una vez alcanzado la fecha fijada.

No puede ser realizado el sorteo si no tiene participantes.

El sorteo lo puede realizar solamente el administrador o propietario del contrato.

d) Usuarios del sistema: tiene contemplado los siguientes.

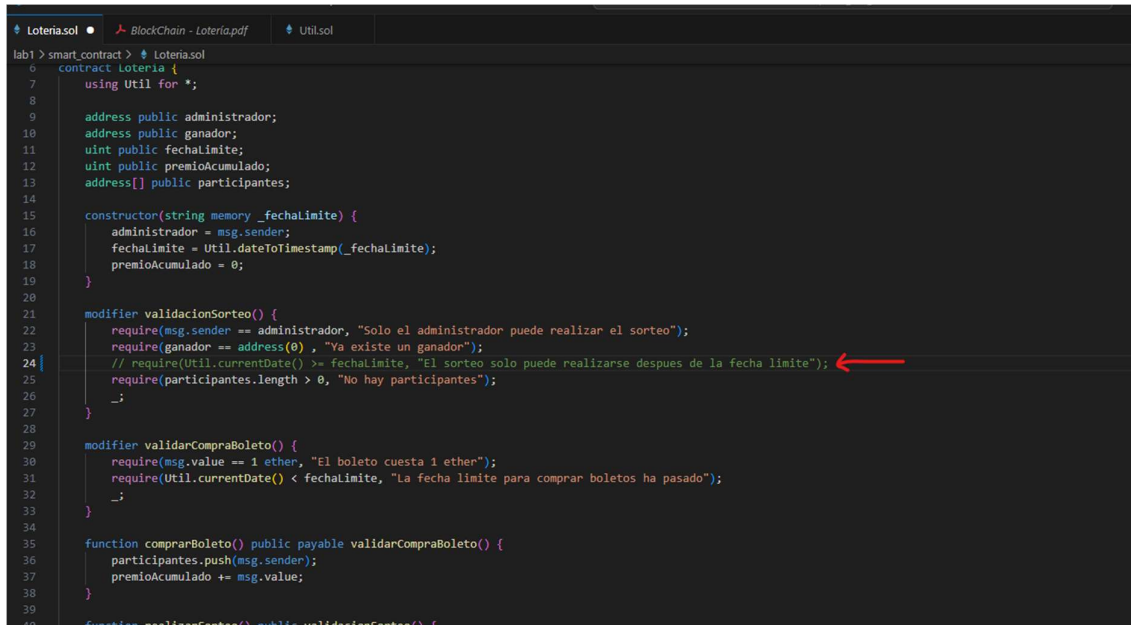
Jugadores participantes conformados por quienes compren los boletos.

Administrador del contrato inteligente quien crea el contrato e inicia el sorteo.

e) Aspectos técnicos: características y puntos a tener en cuenta relacionado a la implementación del contrato inteligente.

Para el despliegue del contrato inteligente es necesario que la fecha del sorteo este indicado en formato YYYY/MM/DD, debido a que a nivel nativo Solidity no tiene soporte para poder manipular fechas en un formato de comprensión humana, y es la razón por la que el control de las fechas está basado en Unix Epoch Time, trabajando en relación a la cantidad de días transcurridas desde la fecha histórica mencionada.

A nivel de pruebas es importante tener en cuenta la zona horaria, ejemplo de ello es que a nivel de las VMs usadas en Remix Ethereum, internamente usan como zona horario UTC-0 y con ello asumir mal interpretaciones cuando a la fecha esperada para la ejecución del sorteo no sea permitido realizarse ya que no está en la fecha del sorteo establecido a nivel del sistema. Este aspecto por motivos de simulación y evaluación puede ser obviada al comentar dicha validación como se indica en la siguiente imagen.



```
lab1 > smart_contract > Loteria.sol
6  contract Loteria {
7      using Util for *;
8
9      address public administrador;
10     address public ganador;
11     uint public fechalimite;
12     uint public premioAcumulado;
13     address[] public participantes;
14
15     constructor(string memory _fechalimite) {
16         administrador = msg.sender;
17         fechalimite = Util.dateToTimestamp(_fechalimite);
18         premioAcumulado = 0;
19     }
20
21     modifier validacionSorteo() {
22         require(msg.sender == administrador, "Solo el administrador puede realizar el sorteo");
23         require(ganador == address(0), "Ya existe un ganador");
24         // require(Util.currentDate() >= fechalimite, "El sorteo solo puede realizarse despues de la fecha limite");
25         require(participantes.length > 0, "No hay participantes");
26         _;
27     }
28
29     modifier validarCompraBoleto() {
30         require(msg.value == 1 ether, "El boleto cuesta 1 ether");
31         require(Util.currentDate() < fechalimite, "La fecha limite para comprar boletos ha pasado");
32         _;
33     }
34
35     function comprarBoleto() public payable validarCompraBoleto() {
36         participantes.push(msg.sender);
37         premioAcumulado += msg.value;
38     }
39
40     function realizarSorteo() public validacionSorteo() {
```

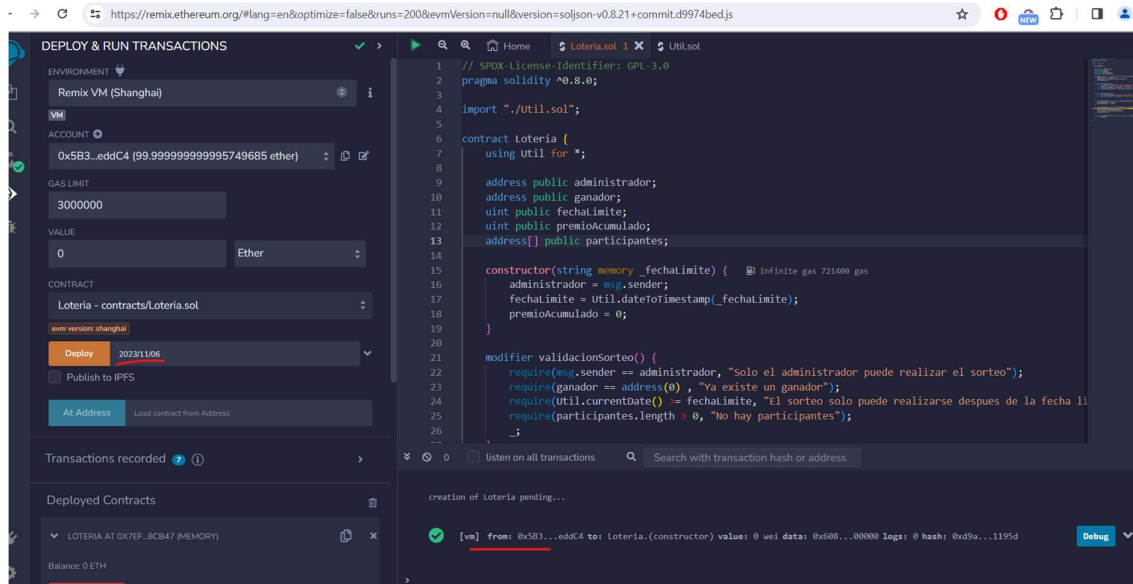
### 3. Implementación

La implementación del contrato inteligente se encuentra dentro del archivo comprimido loteria.rar

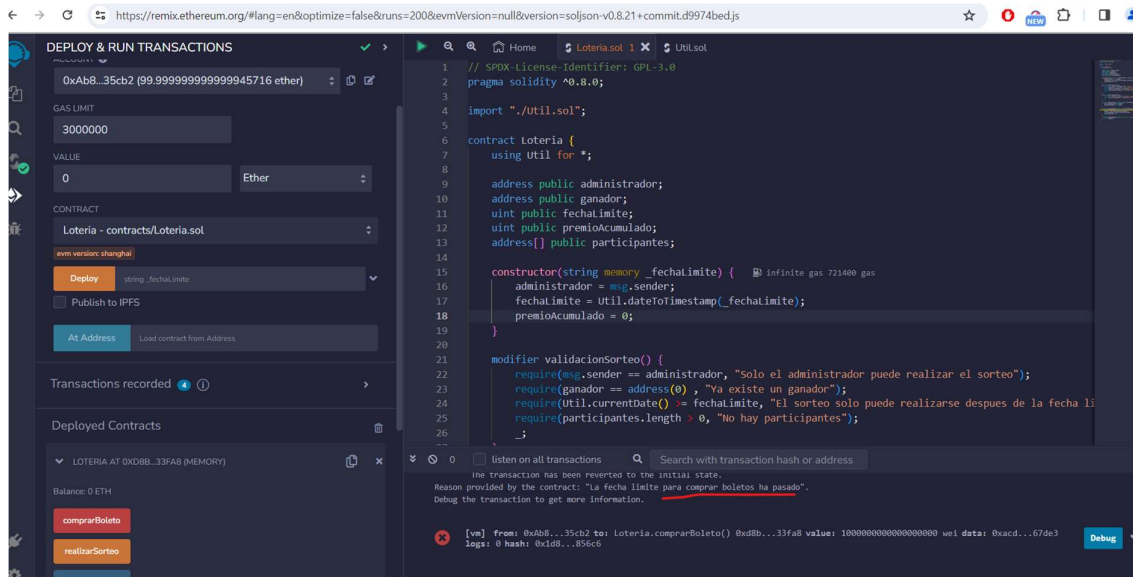
## 4. Pruebas

Las pruebas fueron realizadas en Remix Ethereum IDE, ilustrando a continuación el uso de la misma

### a) Creación del contrato de lotería.



### b) Prueba de validaciones: tratando de comprar un boleto para una lotería cuya fecha es la actual o un día de anterior.



c) Prueba de compra de un boleto.

The screenshot shows the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel is active. The contract 'Loteria - contracts/Loteria.sol' is selected. The 'Deploy' button is highlighted. Below it, the 'At Address' button is visible. The 'Transactions recorded' section shows a list of transactions. The 'Deployed Contracts' section shows the contract 'LOTERIA AT 0x7EF...8CB47 (MEMORY)' with a balance of 1 ETH. The 'comprarBoleto' button is highlighted. On the right, the Solidity code for the 'Loteria' contract is displayed. The code includes a constructor, a 'validacionSorteo' modifier, and a 'validarCompraBoleto' modifier. The transaction log at the bottom shows a successful transaction to 'Loteria.comprarBoleto()' with a value of 1000000000000000000 wei.

d) Simulación de sorteo (se comenta la validación de estar en la fecha del sorteo acordado)

The screenshot shows the Remix IDE interface. On the left, the 'DEPLOY & RUN TRANSACTIONS' panel is active. The contract 'Loteria - contracts/Loteria.sol' is selected. The 'Deploy' button is highlighted. Below it, the 'At Address' button is visible. The 'Transactions recorded' section shows a list of transactions. The 'Deployed Contracts' section shows the contract 'LOTERIA AT 0xD2A...FD005 (MEMORY)' with a balance of 0 ETH. The 'comprarBoleto' button is highlighted. On the right, the Solidity code for the 'Loteria' contract is displayed. The code includes a constructor, a 'validacionSorteo' modifier, and a 'validarCompraBoleto' modifier. The transaction log at the bottom shows a successful transaction to 'Loteria.realizarSorteo()' with a value of 0 wei.

e) Consulta de usuario ganador

← → ↻ <https://remix.ethereum.org/#lang=en&optimize=false&runs=200&e>

### DEPLOY & RUN TRANSACTIONS

ACCOUNT

- 0x5B3...eddC4 (99.9999999999994006252 ether)
- 0x5B3...eddC4 (99.9999999999994006252 ether)
- 0xAb8...35cb2 (101.99999999999975866 ether)**
- 0x4B2...C02db (97.999999999999881344 ether)
- 0x787...cabaB (97.999999999999857603 ether)
- 0x617...5E7f2 (98.99999999999940672 ether)
- 0x17F...8c372 (100 ether)
- 0x5c6...21678 (100 ether)
- 0x03C...D1Ff7 (100 ether)
- 0x1aE...E454C (100 ether)
- 0x0A0...C70DC (100 ether)
- 0xCA3...a733c (100 ether)
- 0x147...C160C (100 ether)
- 0x4B0...4D2dB (100 ether)
- 0x583...40225 (100 ether)
- 0xD8...92148 (100 ether)

At Address Load contract from Address

Transactions recorded 20 ⓘ

### Deployed Contracts

▼ LOTERIA AT 0XD2A...FD005 (MEMORY) ⓘ ✕

Balance: 0 ETH

comprarBoleto

realizarSorteo

administrador

fechaLimite

ganador

obtenerGanador

0: address: 0xAb8483F64d9C6d1EcF9b849Ae677dD3315835cb2

participantes uint256