75.70 Sistemas de programación no convencional de robots

Trabajo Practico 02: Contratos Inteligentes

Objetivo y especificación del trabajo a realizar:
Se requiere crear un contrato inteligente en solidty que cree un *Ethereum token* propio, para ello deberán implementar el estándar para *tokens* llamado ERC20:
https://theethereum.wiki/w/index.php/ERC20_Token_Standard

El *token* creado deberá especificar los siguientes parámetros:

- nombre: nombre del *token*
- símbolo del *token*: una identificación dada por 3 letras mayúsculas, como por ejemplo ETH
- decimales: cantidad de decimales en los que se puede operar con el token (es común usar 18 decimales, pero quedará a criterio del grupo de alumnos)
- Cantidad total de *tokens*: La máxima cantidad de *tokens* que pueden existir.

Ademas el contrato deberá tener un método de tipo *payable* que permita comprar este token con etherum desde la cuenta dueña del contrato, el token se transferirá a cualquier cuenta de etherum valida.

Se podrá asimismo vender tokens a la cuenta dueña siempre que en ella haya ether, en caso contrario enviara un evento notificando el error.

El precio del token en ethers será un parámetro. Este parámetro tendrá un valor inicial fijado por el dueño del contrato. Luego cambiará automáticamente cada vez que alguien compre la moneda (subirá su valor) o venda (bajará su valor). La formula para calcular el precio, queda a criterio de los alumnos, pero deberá ser sencilla.

Así mismo, el contrato principal deberá extender de otro contrato (también creado por el grupo de alumnos) que le permitirá vender “moneda futura”. Se trata de realizar un contrato inteligente entre el creador del *token*, y un usuario cualquiera. Este contrato deberá implementar los siguientes métodos:

ejecutarRegresion(): este método será publico. Podrá ejecutarse a partir de la cuarta transacción. Si se ejecuta antes deberá devolver un error. Sino ejecutará una **regresión lineal simple** con los últimos cuatro valores de operaciones (tiempo y precio del *token*). Una vez ejecutado, almacenará en el contrato los valores correspondientes al desplazamiento y pendiente de la recta. Es decir los parámetros b y m en la siguiente ecuación:

```
y = b+m*t (donde “t” es tiempo e “y” es el precio del token)
```

NOTA: Pueden encontrar una implementación muy sencilla de este modelo en Wikipedia: https://es.wikipedia.org/wiki/Regresi%C3%B3n_lineal. El lenguaje es Javascript, pero pasarla a solidity debería ser trivial.

calcularValorFuturo(fecha): Este método será un método “gratuito” (no consumirá ether) y devolverá el valor del token, para un fecha t pasada por parámetro.

comprarMonedaFutura(fecha, cantidad): Este es un método *payable*, el precio estará dado por la función **calcularValorFuturo.** Y lo que hará es dejar asentado que una dirección dada compró “cantidad” de *tokens* para la fecha estipulada por parámetro. No podrá comprar a más de 90 días.

ConsultarMisComprasFuturas(): método gratuito. Qué permitirá a una dirección (cuenta de etherum), visualizar todas las compras futuras que tiene por cobrar. Indicando si ya puede cobrarla o no.

ConsultarTodasLasComprasFuturas(): método que podrá ejecutar el dueño del contrato para ver todas las compras futuras no ejecutadas aun.

ejecutarMisContratos(): método que podrá ejecutar una dirección para que se acrediten en su cuenta todos los *tokens* comprados con **comprarMonedaFutura().** Siempre y cuando la fecha ya haya pasado.

ejecutarTodosLosContratos(): método que podrá ejecutar el dueño del contrato. Deposita todos los *tokens* comprados a futuro, siempre que la fecha se haya cumplido.

NOTA: claramente, en cada operación de compra y venta del *token* (desde la cuenta principal), se deberá guardar en un registro la fecha y el precio pagado. Solo será necesario mantener los últimos cuatro.

Fecha de entrega: miércoles 29 de mayo

Entrega:
Lo que se espera es que cada grupo entregue uno o varios archivos de extensión “.sol” escrito en solidity. El código deberá poderse ejecutar en la interfaz web Remix: http://remix.ethereum.org/
No hará falta desplegarlo en una blockChain real.

Material de ayuda:
Para ayudarse en el desarrollo de este trabajo, deberán utilizar la IDE Remix, ya mencionada.
Podrán encontrar un tutorial breve pero consiso de como utilizar dicha IDE en el siguiente vídeo de youtube: https://www.youtube.com/watch?v=7I4E78_-B4M
Información para empezar con Solidty y crear una criptoMoneda podrán encontrarla acá: https://solidity.readthedocs.io/en/v0.4.24/
Tutorial interactivo, gratuito y en castellano de solidity: https://cryptozombies.io (Muy recomendable)