



Apertura: martes, 3 de junio de 2025, 10:00 **Cierre:** domingo, 8 de junio de 2025, 23:59



Trabajo Final - Módulo 2

Subasta: Smart Contract

Debes crear un smart contract y desplegarlo desde tu propia dirección.

Requisitos Generales

- · Cada smart contract debe estar:
 - ▶ Publicado en la red Sepolia.
 - Verificado (con el código fuente accesible).
- La URL del contrato publicado y verificado debe incluirse en esta sección:

TRABAJO FINAL MÓDULO 2

· Así mismo una URL del repositorio público en Github.

Funcionalidades Requeridas

Constructor

• Inicializa la subasta con los parámetros necesarios para su funcionamiento.

Función para ofertar

- Permite a los participantes ofertar por el artículo.
- Una oferta es válida si:
 - Es mayor en al menos 5% que la mayor oferta actual.
 - o Se realiza mientras la subasta está activa.

👸 Mostrar ganador

• Devuelve el oferente ganador y el valor de la oferta ganadora.

📜 Mostrar ofertas

• Devuelve la lista de oferentes y sus respectivos montos ofrecidos.

Devolver depósitos

- Al finalizar la subasta:
 - o Se devuelve el depósito a los oferentes no ganadores.
 - Se descuenta una comisión del 2%.

\delta Manejo de depósitos

- Las ofertas deben:
 - o Ser depositadas en el contrato.
 - o Estar asociadas a las direcciones de los oferentes.

Eventos requeridos

- Nueva Oferta: Emitido cuando se realiza una nueva oferta.
- Subasta Finalizada: Emitido cuando finaliza la subasta.

💋 Funcionalidades Avanzadas

Reembolso parcial

Durante la subasta, los participantes pueden retirar el importe por encima de su última oferta válida.

Ejemplo:

Tiempo	Usuario	Oferta
ТО	Usuario 1	1ETH
T1	Usuario 2	2 ETH
T2	Usuario 1	3 ETH

→ Usuario 1 puede pedir el reembolso de la oferta TO (1 ETH).

Consideraciones Adicionales

- Se deben utilizar modificadores cuando sea conveniente.
- Para superar la mejor oferta, la nueva debe ser superior al menos en 5%.
- Si una oferta válida se realiza dentro de los últimos 10 minutos, el plazo de la subasta se extiende 10 minutos más.
- El contrato debe ser seguro y robusto. Manejando adecuadamente los errores y las posibles situaciones excepcionales.
- Se deben utilizar eventos para comunicar los cambios de estado de la subasta a los participantes.
- La documentación del contrato debe ser clara y completa en un archivo README.md en el repositorio en GitHub, explicando:
 - Funciones
 - Variables
 - Eventos

P Entrega

- 1. El trabajo debe ser presentado en la sección TRABAJO FINAL MÓDULO 2, incluyendo solo la URL del contrato publicado y verificado.
- 2. También debe enviarse un repositorio público en Github que sirva a manera de documentación, explicando la manera en que está construida la subasta.

🔟 Fechas importantes para la entrega

- 03.06 → Se informa la consigna del trabajo final.
- 08.06 → **O** Soft deadline:

Quienes entreguen entre el 03.06 y el 08.06 a las 23:59 horas inclusive, podrán recibir correcciones, hacer ajustes y volver a entregar una versión mejorada del trabajo.

• 11.06 → **()** Hard deadline:

Quienes entreguen entre el 09.06 y el 11.06 a las 23:59 horas inclusive, recibirán corrección únicamente sobre lo entregado, sin posibilidad de hacer cambios posteriores.

• No se aceptarán trabajos prácticos una vez cumplido el hard dealine.

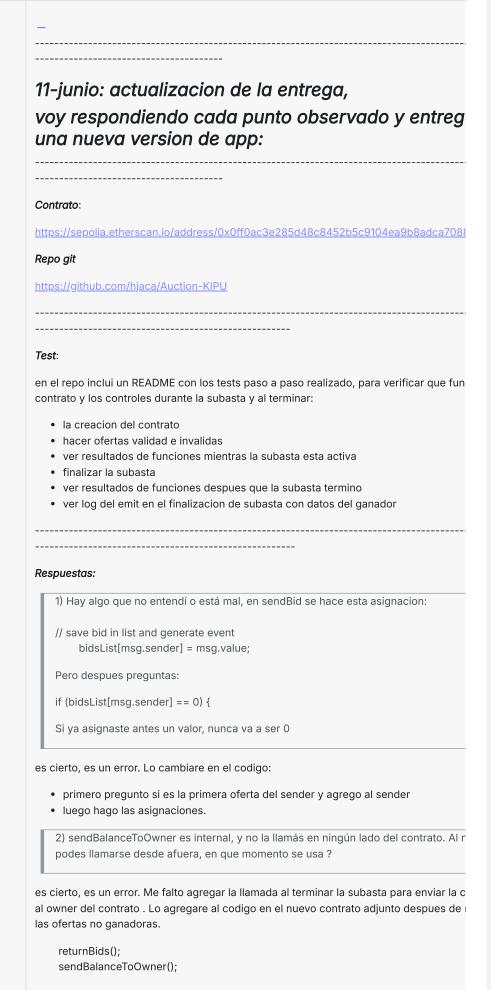
Editar entrega

Borrar entrega

Estado de la entrega

Estado de la entrega	Enviado para calificar
Estado de la calificación	Calificado
Tiempo restante	La tarea fue enviada 2 días 23 horas después de la fecha límite

Última modificación miércoles, 11 de junio de 2025, 23:42



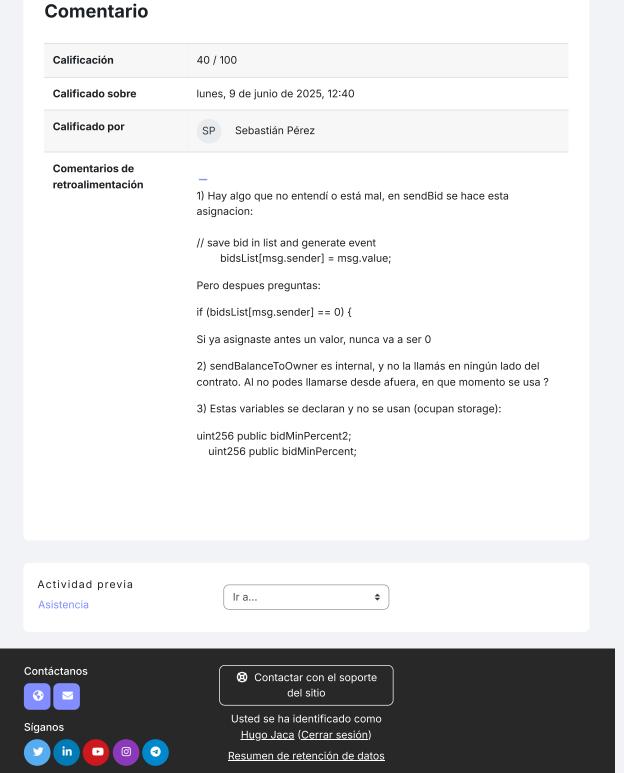
Texto en línea

uint256 public bidMinPercent2; uint256 public bidMinPercent; habia quedado como parte del desarrollo, las pensaba usar para hacer variable el % de comision (2%) y de mejor oferta (5%), pero no me funcionaba. Lo sacare del codigo el nuevo contrato adjunto. Espero que con estos cambios quede listo, lo probe y funciona con varias address ofer Se puede ver el test realizado, con imagenes, en el git PRIMERA ENTREGA 8-junio Adjunto entrega del TP2 Contrato: 0xadA6Fff7ce381F3702d73B6C139D05e3af43CC53 https://sepolia.etherscan.io/address/0xada6fff7ce381f3702d73b6c139d05e3af43cc53f Repo git https://github.com/hjaca/Auction-KIPU NOTAS: • el git tiene una estructura estandard para solidity • defini carpetas para contratos, scripts, test como estan en Remix • deje un file de asumpciones del contrato • la documentacion la escribi en el README del git copiando el enunciado y agrega definiciones del contrato, constructor, funciones, eventos • el codigo tiene documentacion comentada · el contrato fue verificado en sepolia ✓ Comentarios (1) HJ <u>Hugo Jaca</u> - mié., 11 de jun. de 2025, 23:51 Acciones realizadas: 1- actualice el contrato con los 3 puntos de los comentarios y el repo en git 2- Agregue en el git un README con el resultado del test paso a paso realizado con pruebas incluyendo la sul activa, la finalizacion de subasta, y pruebas con subasta inactiva 3- me qustaria saber que puntos de mis errores impactaron mas en el puntaje final de la primera version, de 4 para entender donde poner mas atencion para los siguientes modulos. muchas gracias por su feedback! saludos Agregar un comentario... Guardar comentario | Cancelar

Comentarios

de la entrega

3) Estas variables se declaran y no se usan (ocupan storage):



Este tema fue desarrollado por conecti.me