**ACTIVIDAD PRÁCTICA # 2**

**Actividad:**

Supongamos que una empresa de bienes raíces desea tokenizar una propiedad comercial para permitir a los inversores individuales adquirir fracciones de la propiedad. La propiedad tiene un valor total de $1,000,000 y la empresa desea emitir tokens que representen fracciones del 10% de la propiedad cada uno. Diseña un contrato inteligente en Ethereum que permita a los inversores comprar y vender estos tokens, así como recibir dividendos proporcionales a sus participaciones en la propiedad.

Solución (en pseudocódigo):---->Solidity

// Declaración del contrato inteligente

contract TokenizedProperty {

 // Declaración de variables

 address public owner;

 string public propertyName;

 uint public propertyValue;

 uint public tokenPrice;

 mapping(address => uint) public balances;

 // Eventos

 event TokensPurchased(address buyer, uint amount);

 event TokensSold(address seller, uint amount);

 // Constructor

 constructor(string memory \_name, uint \_value, uint \_tokenPrice) {

 owner = msg.sender;

 propertyName = \_name;

 propertyValue = \_value;

 tokenPrice = \_tokenPrice;

 }

 // Función para que los inversores compren tokens

 function buyTokens(uint \_amount) public payable {

 require(msg.value == \_amount \* tokenPrice, "Amount sent does not match token price");

 balances[msg.sender] += \_amount;

 emit TokensPurchased(msg.sender, \_amount);

 }

 // Función para que los inversores vendan tokens

 function sellTokens(uint \_amount) public {

 require(balances[msg.sender] >= \_amount, "Insufficient balance");

 balances[msg.sender] -= \_amount;

 payable(msg.sender).transfer(\_amount \* tokenPrice);

 emit TokensSold(msg.sender, \_amount);

 }

 // Función para calcular los dividendos

 function calculateDividends() public view returns (uint) {

 return address(this).balance / propertyValue;

 }

}

Este contrato inteligente permite a los inversores comprar y vender tokens que representan fracciones de la propiedad comercial tokenizada. Los inversores pueden comprar tokens enviando Ether al contrato y luego usar esos tokens para recibir dividendos proporcionales a sus participaciones en la propiedad.

**NOTA:** Recuerde que lo anterior es un pseudocódigo, una base algorítmica para desarrollar el ejercicio, esto puede variar según las consideraciones del ejercicio.

**Pasos para ejecutar:**

* **Instalar un entorno de desarrollo Ethereum:** Antes de compilar un contrato inteligente, necesitarás tener un entorno de desarrollo Ethereum configurado en tu computadora. Puedes utilizar herramientas como Truffle, Remix, o Hardhat para este propósito. Estas herramientas proporcionan un entorno de desarrollo integrado (IDE) que simplifica el proceso de desarrollo y compilación de contratos inteligentes.
* **Crear el archivo del contrato:** Utiliza un editor de texto o un IDE para escribir el código de tu contrato inteligente en Solidity. Guarda el código en un archivo con la extensión .sol.
* **Abrir el entorno de desarrollo Ethereum:** Abre tu entorno de desarrollo Ethereum elegido (por ejemplo, Truffle, Remix o Hardhat).
* **Importar el archivo del contrato:** Importa el archivo del contrato inteligente en el entorno de desarrollo Ethereum.
* **Compilar el contrato:** Utiliza las herramientas proporcionadas por el entorno de desarrollo para compilar el contrato inteligente. Esto generará un bytecode que puede ser ejecutado en la Ethereum Virtual Machine (EVM).
* **Revisar errores de compilación:** Verifica que el contrato se haya compilado correctamente y revisa si hay errores de compilación. Si hay errores, corrígelos y vuelve a compilar el contrato.
* **Desplegar el contrato:** Una vez que el contrato se ha compilado con éxito, puedes desplegarlo en una red Ethereum utilizando las herramientas proporcionadas por el entorno de desarrollo. Esto te permitirá interactuar con el contrato y ejecutar sus funciones en la red Ethereum.

O también se puede hacer online en Remix IDE:

* **Acceder a Remix:** Abre tu navegador web y ve a la página web de Remix en<https://remix.ethereum.org/>.
* **Crear un nuevo archivo:** En Remix, haz clic en el icono de "+" en el panel de la barra lateral izquierda para crear un nuevo archivo. Selecciona "Solidity" como el tipo de archivo.
* **Escribe el código del contrato:** En el nuevo archivo, escribe el código de tu contrato inteligente en Solidity.
* **Compilar el contrato:** Una vez que hayas escrito el código del contrato, haz clic en la pestaña "Solidity Compiler" en la barra lateral derecha. Luego, haz clic en el botón "Compile" para compilar el contrato.
* **Revisar errores de compilación:** Después de compilar el contrato, verifica si hay errores de compilación en la pestaña "Compiler Output" en la parte inferior de la pantalla. Si hay errores, corrígelos y vuelve a compilar el contrato.
* **Desplegar el contrato (opcional):** Si deseas probar tu contrato en una red Ethereum, puedes utilizar Remix para desplegarlo en una red de prueba o en la red principal. Para hacer esto, ve a la pestaña "Deploy & Run Transactions" en la barra lateral derecha, selecciona el contrato que deseas desplegar y haz clic en el botón "Deploy".