
HEAVY DUTY CAMP V4
INTERACTUANDO CON LA RED DE SOLANA
CLASE #4

CUENTA TOKEN ASOCIADA

- Es una cuenta token con una dirección específica que es generada utilizando algunas entradas, y que siempre genera la misma dirección para las mismas entradas.
- Esto introduce un concepto clave en el desarrollo de Solana: Dirección derivada de un programa (PDA).

DIRECCIONES DERIVADAS DE PROGRAMAS (PDAS)

- Las PDA son direcciones derivadas utilizando una combinación de semillas definidas por el usuario, un bump o nonce y el identificador de un programa.
- Están diseñadas para no poseer una llave privada asociada, de forma que ninguna entidad puede generar una firma válida para la PDA.

DERIVANDO UNA PDA

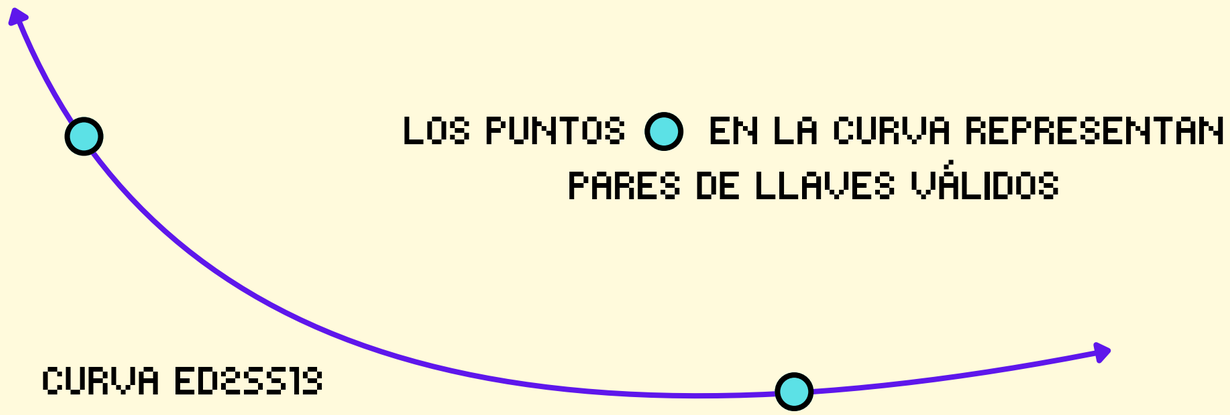
- El proceso de derivación de PDAs en Solana utiliza un algoritmo basado en hashing, generalmente SHA-256.
- Este algoritmo garantiza que las PDAs no caen dentro del espacio de direcciones válidas generadas por el algoritmo Ed25519.

ALGORITMO ED25519

- En Solana, las cuentas normales se basan en pares de claves Ed25519, que consisten en una clave pública y una clave privada.
- Las claves públicas son utilizadas como direcciones de cuentas, y las claves privadas son necesarias para firmar transacciones que afecten estas cuentas.

ALGORITMO ED25519

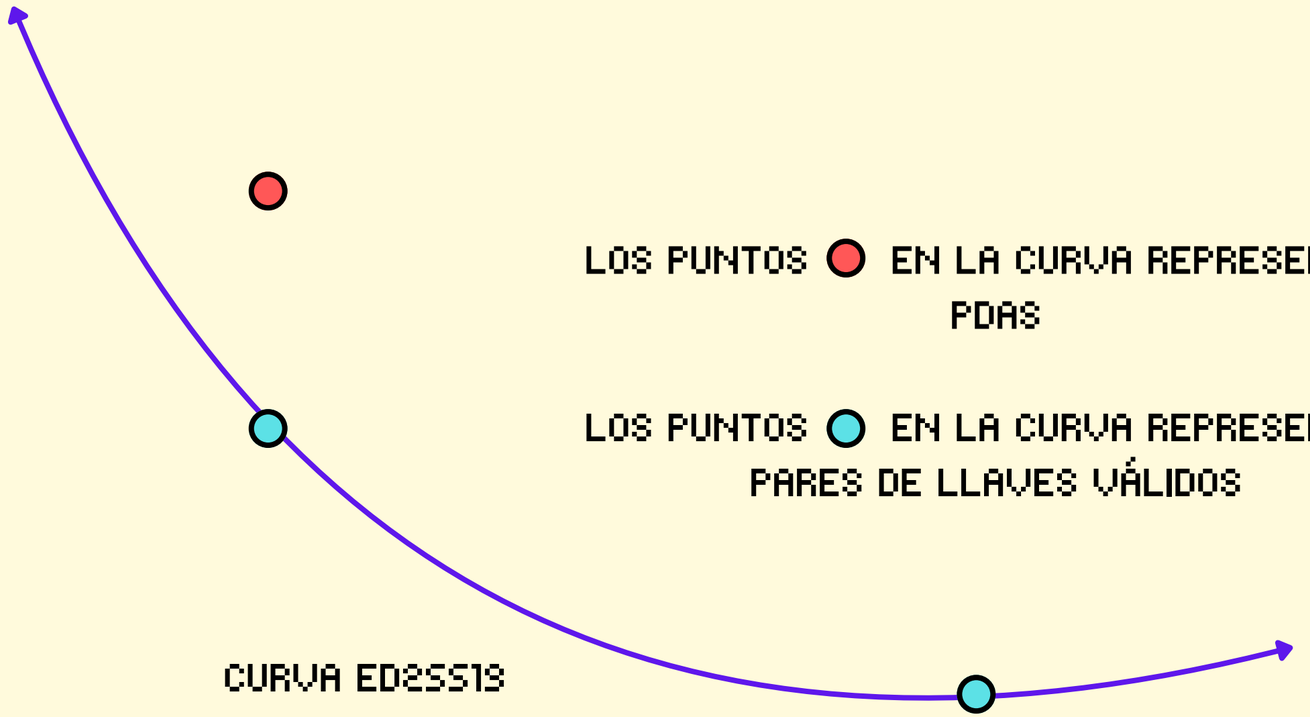
- Los pares de llaves válidos, resultan ser puntos de la curva Ed25519:



PDAS

- Una PDA es un punto que se deriva intencionadamente para caer fuera de la curva Ed25519 utilizando un conjunto predefinido de entradas.
- Un punto que no está en la curva Ed25519 no tiene una llave privada correspondiente válida y no puede utilizarse para operaciones criptográficas.

PDAS



LOS PUNTOS ● EN LA CURVA REPRESENTAN PDAS

LOS PUNTOS ● EN LA CURVA REPRESENTAN PARES DE LLAVES VÁLIDOS

CURVA ED25519

PDAS

- Una PDA puede utilizarse entonces como dirección (identificador único) para una cuenta en la cadena, lo que proporciona un método para almacenar, asignar y recuperar fácilmente el estado del programa.
- Solo el programa del cual se deriva puede realizar operaciones con la cuenta.

COMO DERIVAR UNA PDA

- Para encontrar una dirección PDA válida se requieren para el algoritmo de hash 3 entradas: el identificador del programa que manejará la PDA, la semilla bump y una lista de semillas adicionales que son opcionales.

- La obtención de una PDA no crea automáticamente una cuenta en Solana.

DERIVANDO UNA PDA: ENTRADAS

- Clave Pública del Programa (Program ID): La dirección del programa de Solana que desea derivar una PDA.
- Semillas (seeds): Una o más cadenas de bytes que sirven como identificadores únicos. Estos pueden ser valores como texto, números o cualquier conjunto de bytes.
- Bump Seed: Un número (0-255) que se ajusta para encontrar una dirección válida que no tenga una clave privada asociada.

DERIVANDO UNA PDA: CÁLCULO

- El proceso utiliza una función hash, generalmente SHA-256, para combinar las semillas, la clave pública del programa y el bump seed. La fórmula de derivación es la siguiente:

```
PDA = SHA256(Program ID || seed1 || seed2 || ... || bump_seed)
```

- Donde || representa la concatenación de los elementos.

DERIVANDO UNA PDA: VERIFICACIÓN

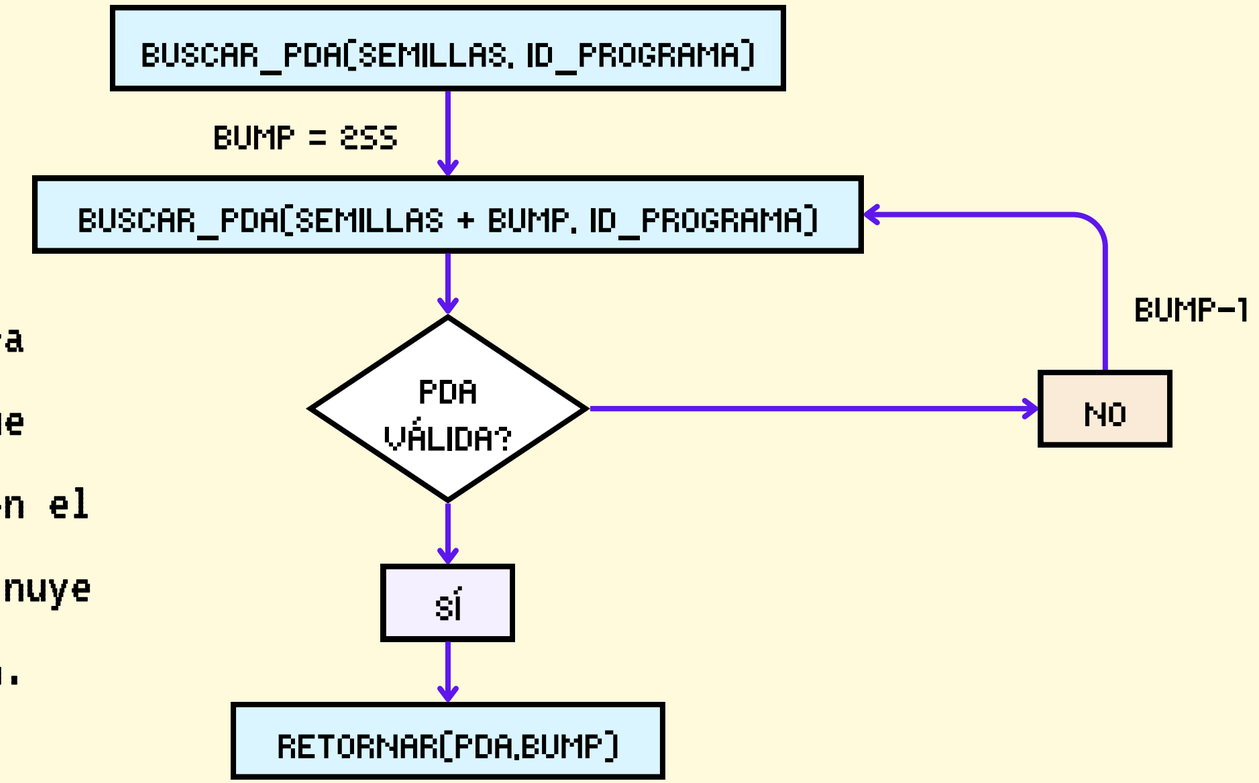
- Una PDA debe cumplir con la condición de que no puede ser una clave pública válida de un par de claves generado por el algoritmo de claves Ed25519.
- De esta forma se verifica que ningún usuario externo puede manipular la cuenta a la que se asigne la dirección de la PDA.

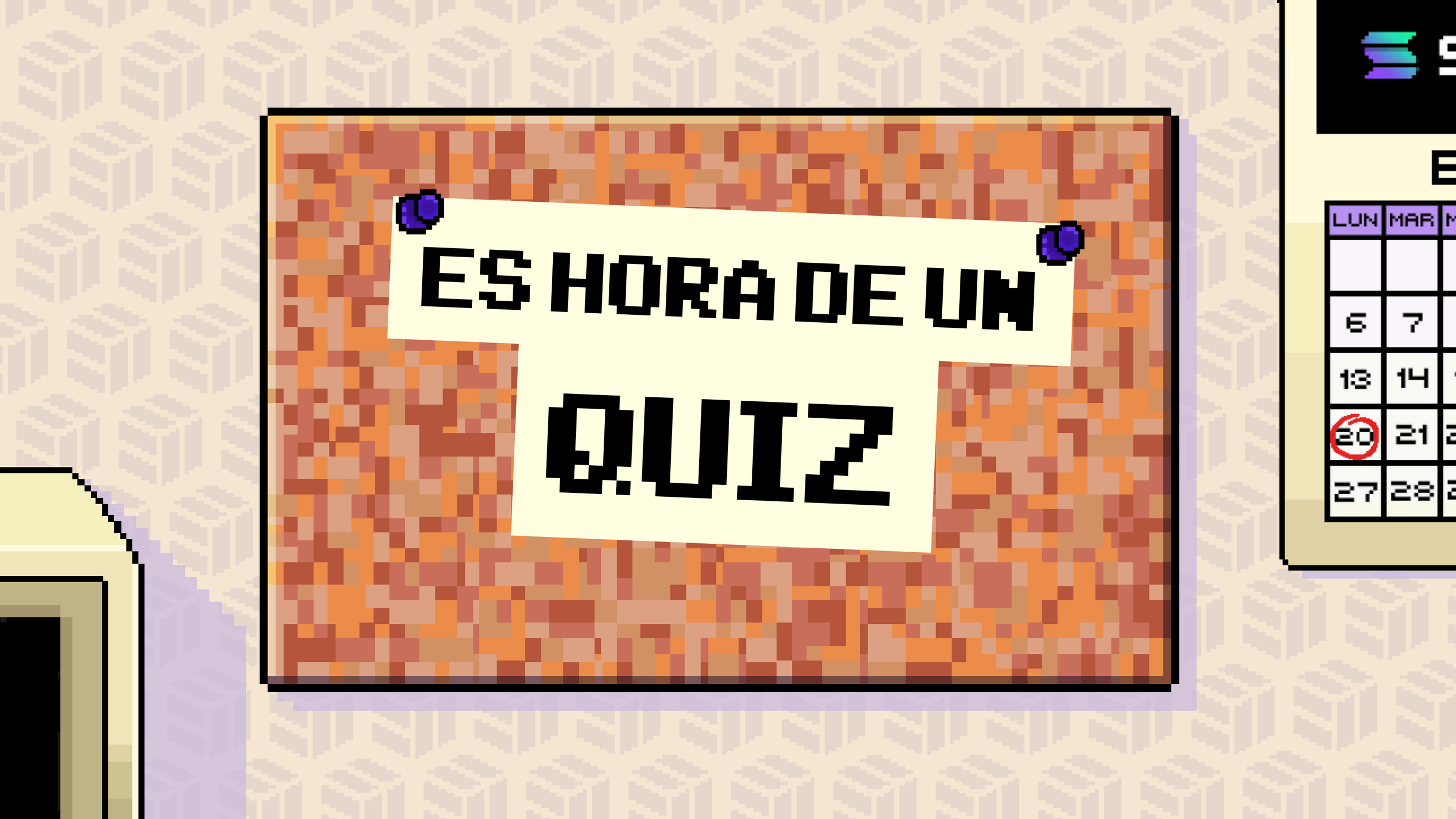
DERIVANDO UNA PDA: ITERANDO EL BUMP

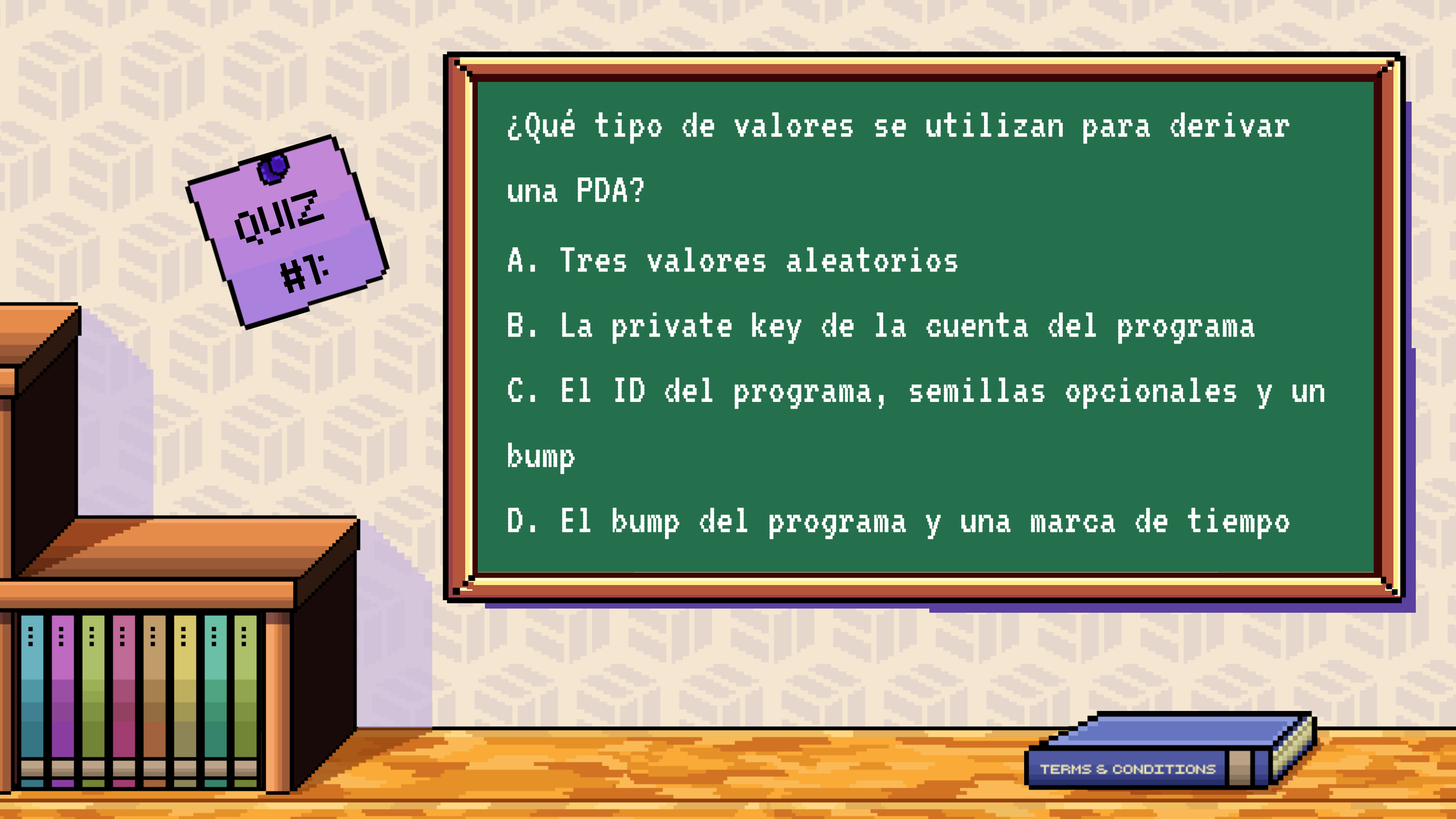
- Si la dirección derivada resulta ser una clave pública válida, el algoritmo incrementa el bump seed y repite el proceso hasta encontrar una dirección que cumpla con la condición de ser una PDA.
- La primera semilla bump entre 0 y 255 que genera una PDA válida se conoce como bump canónico.

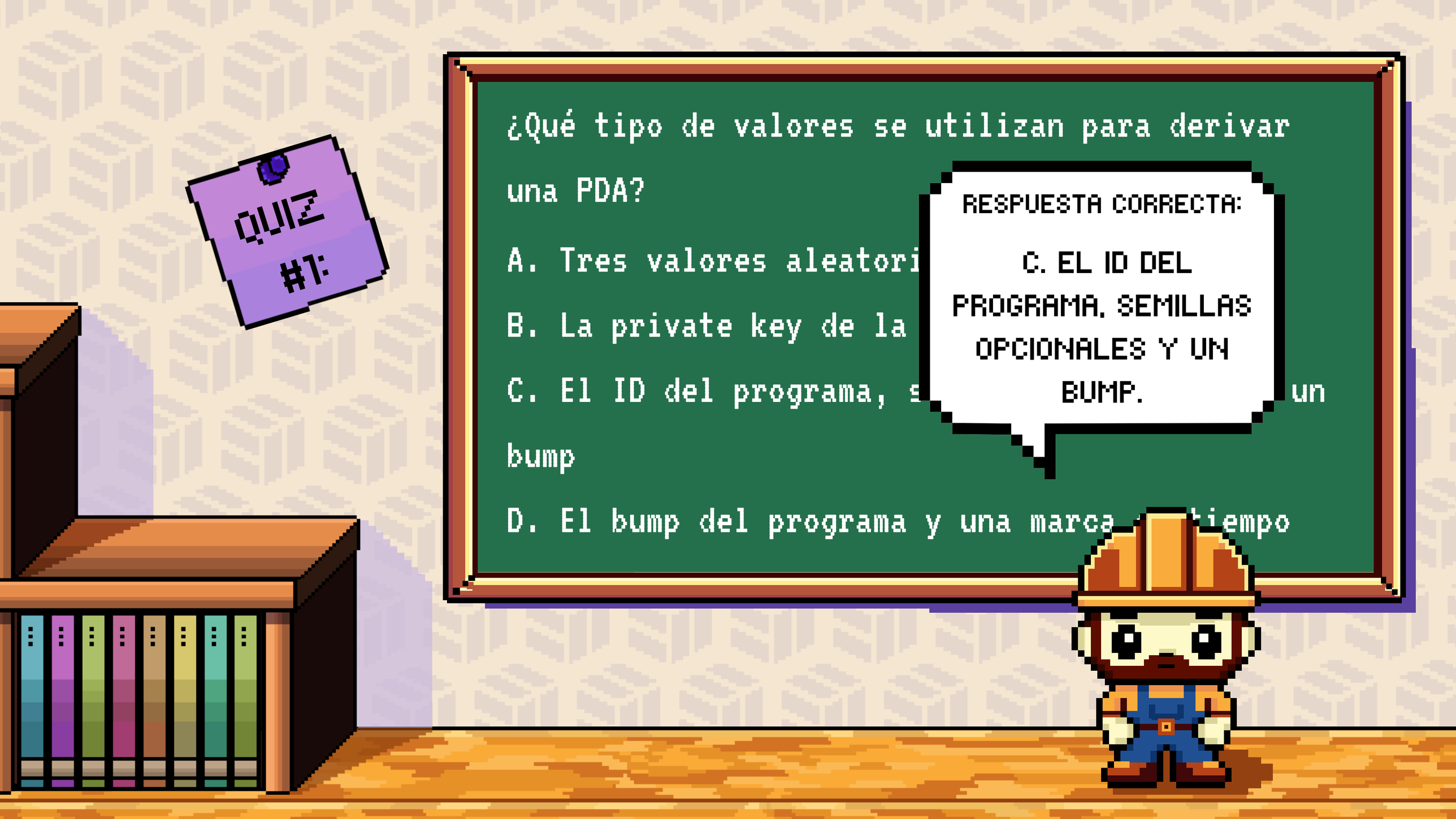
DERIVANDO UNA PDA

El algoritmo itera sobre el bump, que inicia siempre con el valor 255 y disminuye en cada iteración.




ES HORA DE UN
QUIZ
LUN
MAR
6
7
13
14
20
21
27
28


QUIZ
#1:
¿Qué tipo de valores se utilizan para derivar una PDA?
A. Tres valores aleatorios
B. La private key de la cuenta del programa
C. El ID del programa, semillas opcionales y un bump
D. El bump del programa y una marca de tiempo
TERMS & CONDITIONS


QUIZ #1:
¿Qué tipo de valores se utilizan para derivar una PDA?
A. Tres valores aleatori
B. La private key de la
C. El ID del programa, s
un bump
D. El bump del programa y una marca
iempo
RESPUESTA CORRECTA:
C. EL ID DEL PROGRAMA, SEMILLAS OPCIONALES Y UN BUMP.

EJEMPLO: DERIVANDO UNA PDA

- Para derivar una PDA, podemos utilizar el método findProgramAddressSync de @solana/web3.js
- Una vez encontrada una PDA válida, findProgramAddressSync devuelve tanto la dirección (PDA) como la semilla bump utilizada para derivar la PDA.

BUMP CANÓNICO

- El "bump canónico" se refiere a la primera semilla bump (comenzando en 255 y disminuyendo en 1) que deriva una PDA válida.
- Es utilizado para generar PDAs en Solana de forma segura.
- Otras semillas bump diferentes al bump canónico pueden obtener direcciones PDA válidas.

BUMP CANÓNICO

- Al crear programas en Solana, se recomienda incluir comprobaciones de seguridad que validen que una PDA pasada al programa se deriva usando el bump canónico.
- No hacerlo puede introducir vulnerabilidades que permitan suministrar cuentas inesperadas a un programa.

EJEMPLO: OBTENIENDO EL BUMP CANÓNICO

- La función findProgramAddressSync añade iterativamente el bump a las semillas y llamará al método createProgramAddressSync.
- Podemos emular este comportamiento, pasando de forma manual un bump específico y disminuyéndolo en cada iteración hasta encontrar la primera PDA.

VENTAJAS DE LAS PDAS

- Estas direcciones especializadas permiten a los programas interactuar de manera segura con cuentas, gestionando datos y recursos en la red sin necesidad de una entidad externa al programa.
- Solo el programa que derivó la PDA puede autorizar transacciones desde esa dirección, reduciendo el riesgo de accesos no autorizados.

VENTAJAS DE LAS PDAS

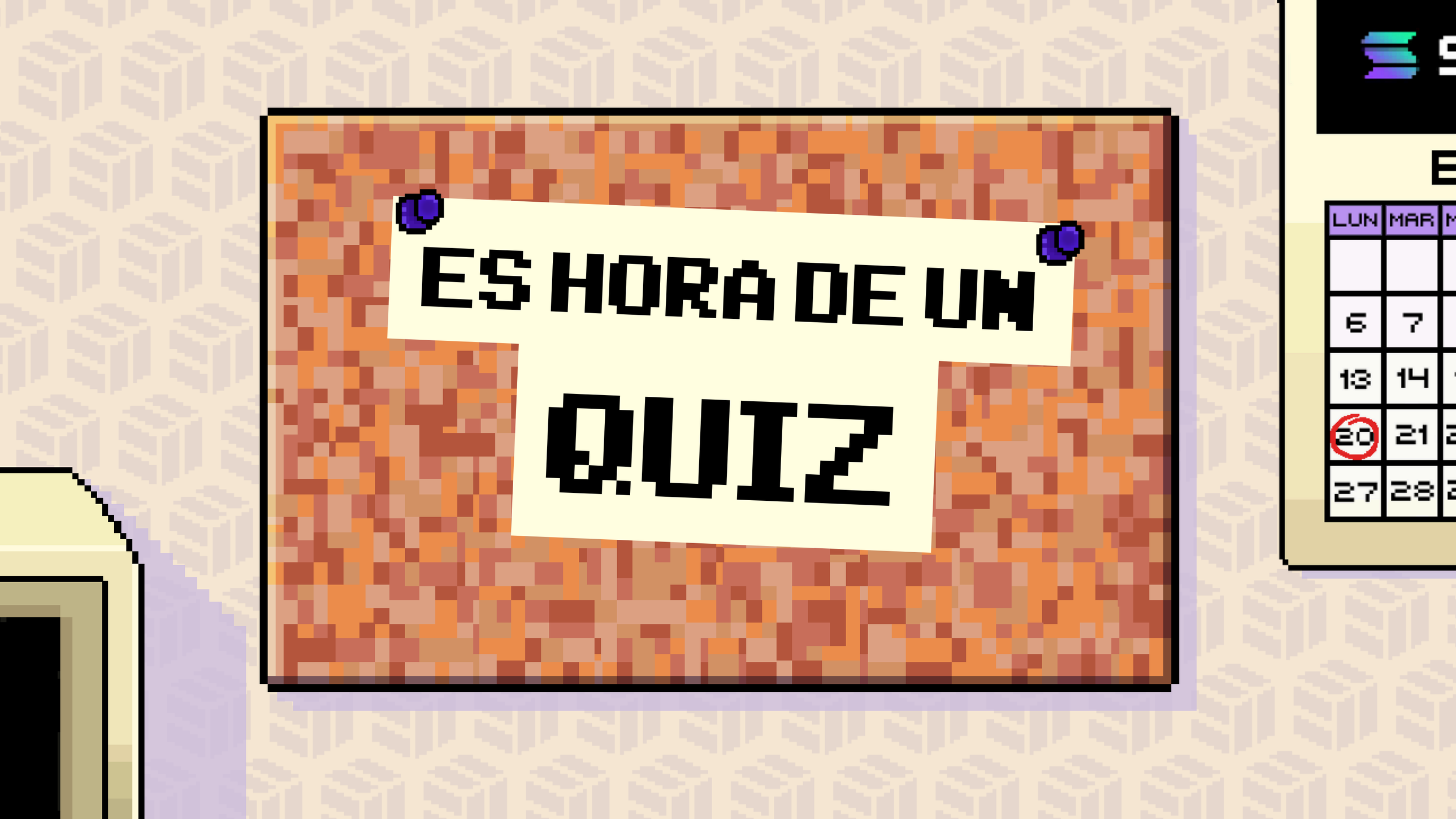
- Al ser derivadas de manera determinística usando semillas específicas y la clave pública del programa, no es necesario recordar la dirección de la PDA, ya que siempre se puede calcular si conoce las semillas y el programa, sin necesidad de interacción previa.

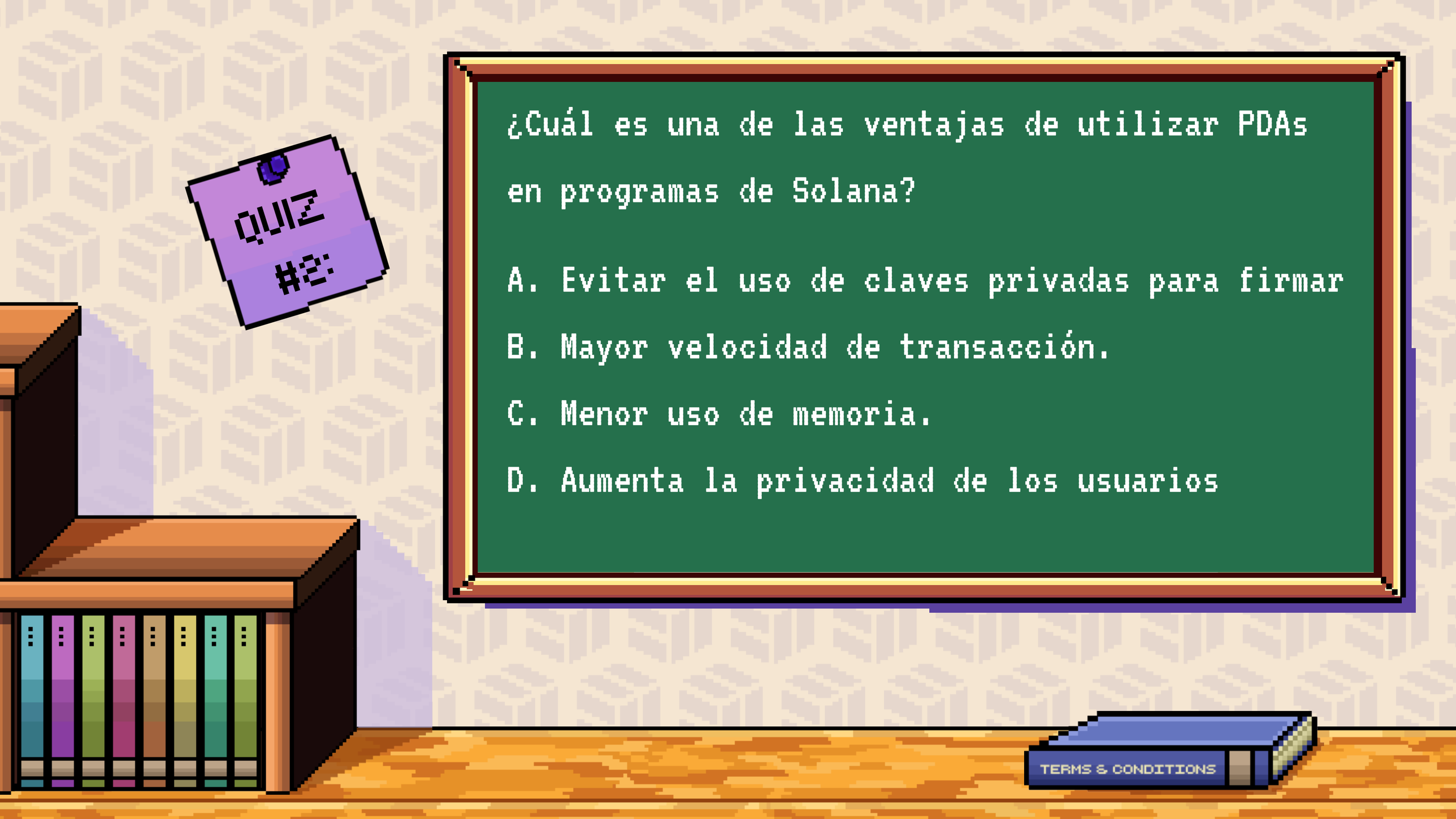
CASOS DE USO

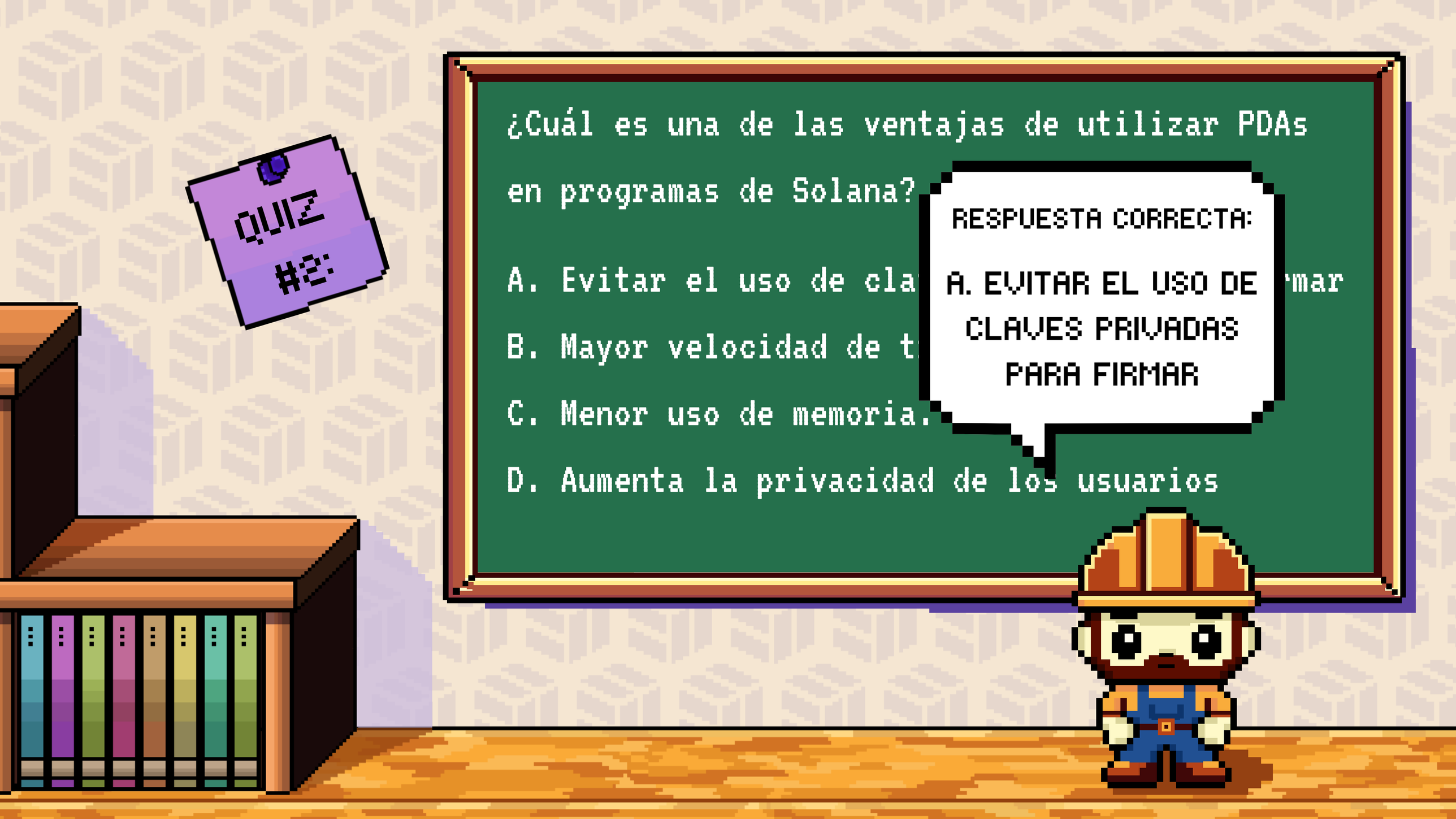
- Las PDAs permiten que los programas gestionen recursos sin necesidad de intervención humana, lo que es crucial para aplicaciones automatizadas y contratos inteligentes que operan sin supervisión directa.

CASOS DE USO

- El programa puede firmar transacciones para la PDA, lo que permite que las dApps manejen flujos de trabajo complejos que requieren transacciones automáticas, como liquidaciones de préstamos, distribución de recompensas o actualizaciones de estado.


ES HORA DE UN
QUIZ
LUN
MAR
6
7
13
14
20
21
27
28


QUIZ #2:
¿Cuál es una de las ventajas de utilizar PDAs en programas de Solana?
A. Evitar el uso de claves privadas para firmar
B. Mayor velocidad de transacción.
C. Menor uso de memoria.
D. Aumenta la privacidad de los usuarios
TERMS & CONDITIONS


QUIZ #2:
¿Cuál es una de las ventajas de utilizar PDAs en programas de Solana?
A. Evitar el uso de cla
B. Mayor velocidad de t
C. Menor uso de memoria.
D. Aumenta la privacidad de los usuarios
RESPUESTA CORRECTA:
A. EVITAR EL USO DE CLAVES PRIVADAS PARA FIRMAR

DEFINIENDO PDAS EN ANCHOR

- De la misma forma que con las cuentas discutidas anteriormente, definir una cuenta con PDA se hace utilizando la macro #[account].
- Dentro de #[account] se definen los mismos atributos que para una cuenta regular y además se definen las semillas que se utilizarán para el cálculo de la PDA.

DEFINIENDO PDAS EN ANCHOR

- Adicional a las semillas opcionales, se debe proporcionar un bump a la macro #[account], que almacenará el bump canónico utilizado para generar la PDA y que luego puede ser accedido una vez creada la cuenta.

SEMILLAS PARA UNA PDA

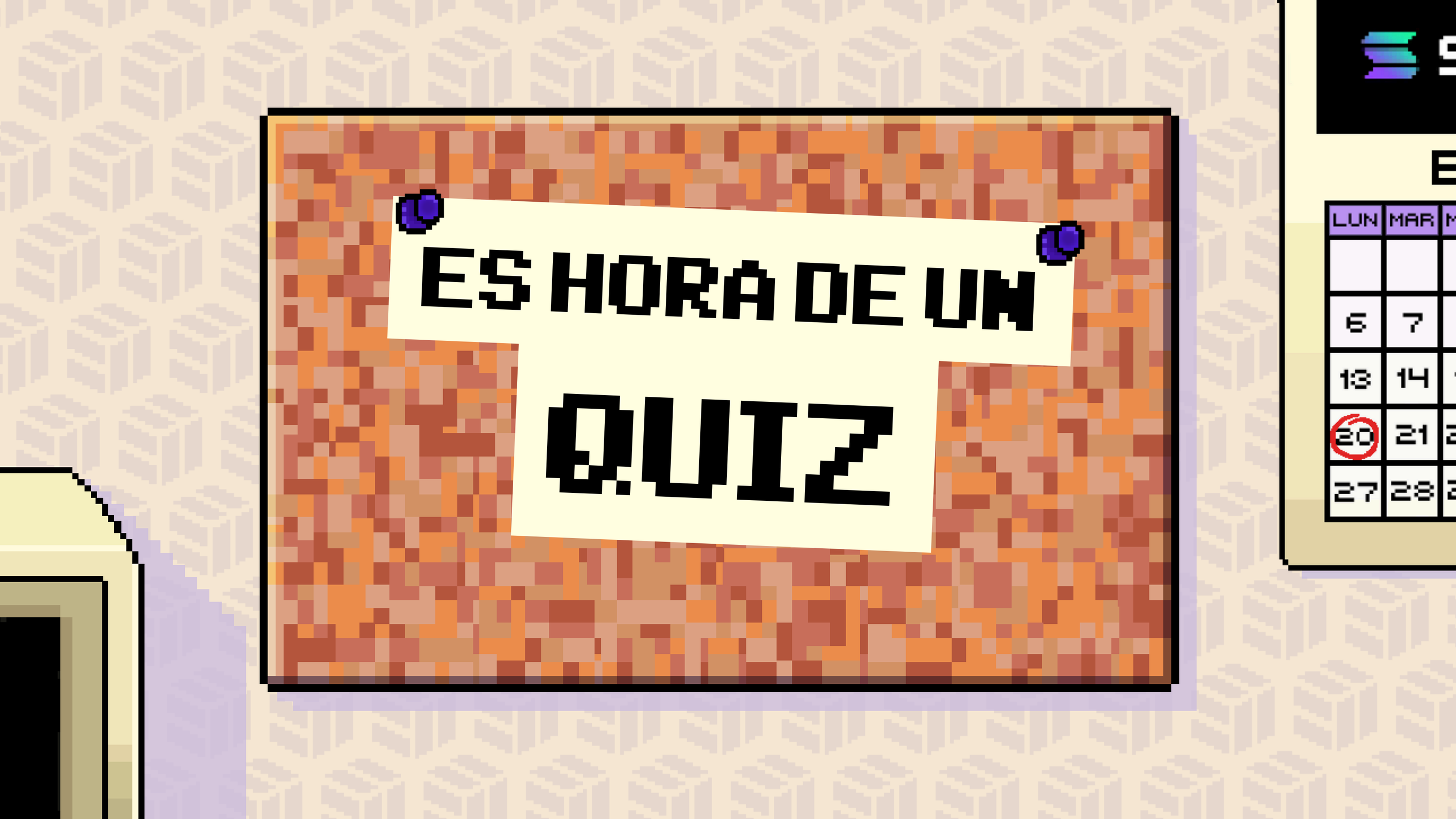
- Las semillas seleccionadas para generar una PDA deben ser únicas para cada cuenta dentro del programa que utilice PDAs.
- El tamaño máximo de las semillas opcionales es de 128 bytes entre todas ellas.

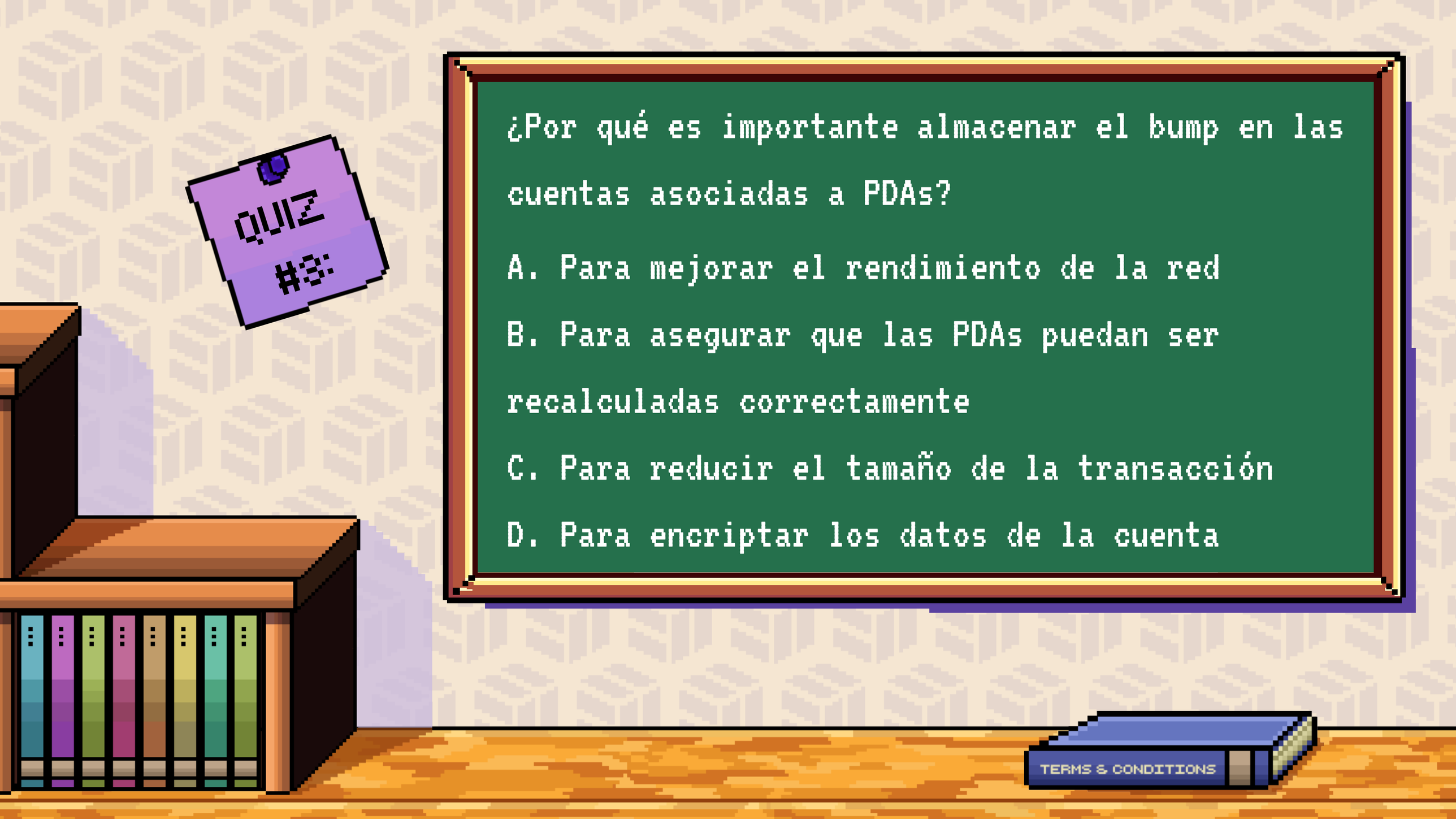
ALMACENANDO EL BUMP DE UNA PDA

- Las semillas seleccionadas para generar una PDA deben ser únicas para cada cuenta dentro del programa que utilice PDAs.
- El tamaño máximo de las semillas opcionales es de 128 bytes entre todas ellas.

EJEMPLO: CREANDO UNA CUENTA PDA

- Derivar una PDA no implica que se crea una cuenta con esta dirección asociada automáticamente.
- Con Anchor, el proceso de derivar una PDA y crear la cuenta está abstraído y simplificado, por lo que sólo es necesario agregar las semillas que se utilizarán en el cálculo de la PDA en el contexto de la instrucción que crea la cuenta PDA.


ES HORA DE UN
QUIZ
LUN
MAR
6
7
13
14
20
21
27
28


QUIZ #3:
¿Por qué es importante almacenar el bump en las cuentas asociadas a PDAs?
A. Para mejorar el rendimiento de la red
B. Para asegurar que las PDAs puedan ser recalculadas correctamente
C. Para reducir el tamaño de la transacción
D. Para encriptar los datos de la cuenta
TERMS & CONDITIONS

QUIZ #3:

¿Por qué es importante almacenar el bump en las cuentas asociadas a PDA

A. Para mejorar el rend

B. Para asegurar que la

recalculadas correctamen

C. Para reducir el tamaño de l transacción

D. Para encriptar los datos de la

RESPUESTA CORRECTA:

B. PARA ASEGURAR QUE LAS PDAS PUEDAN SER RECALCULADAS CORRECTAMENTE


HEAVY DUTY CAMP V4
NOS VEMOS EN LA PROXIMA CLASE!
CLASE #4