Тестовое задание на позицию Solana / Rust developer

26 октября 2021

На децентрализованных биржах, реализованных на Ethereum, популярен механизм торговли, альтернативный списку лимитных заявок на традиционных биржах, - AMM (Automated market maker). В данном тестовом задании предлагается реализовать этот механизм на Solana.

Модель

Существует два токена A и B. У некоторой распределённой программы есть два соответствующих кошелька, она может отправлять и получать эти токены. Между количествами токенов, которые держит данная программа, поддерживается следующее соотношение: X * Y = K (K - некоторая постоянная величина, X и Y - количества токенов A и B соответственно). В случае, когда некоторый трейдер отправляет программе dX токенов A, для поддержания соотношения она должна вернуть отправителю некоторое количество dY токенов B в соответствии с формулой:

$$(X + dX) * (Y - dY) = K$$
$$dY = Y - K / (X + dX)$$

Предлагается написать on-chain программу для Solana, реализующую этот механизм. Воспроизведение данной логики требует понимания всех основных механик on-chain программ Solana. Мы предлагаем некоторый мнимый способ в целях упрощения задания, но оставляем кандидату право реализовать механизм так, как это должно быть в реальности.

Требования к реализации

Далее перечислены требования к реализации в порядке убывания их приоритета. Решения, реализующие не все пункты, также будут рассмотрены (последние два пункта опциональны).

- На вход программе приходят 4 аккаунта: 2 PDA аккаунта, принадлежащих этой программе и 2 аккаунта принадлежащих пользователю. Аккаунты не содержат данных, количество lamports у одного из PDA аккаунтов соответствует X, а у второго - Y. Таким образом мы считаем, что на этих аккаунтах лежат разные монеты A и B, хотя реально и там, и там лежит нативный токен Solana. Такое допущение упрощает текущее задание. Один из аккаунтов пользователя соответствует кошельку с токеном A, а второй - с токеном B. Инструкция содержит информацию о том, какой токен пользователь хочет перевести на соответствующий кошелёк программы, а также количество для перевода. Постоянная K вычисляется из текущих состояний кошельков программы.

- Созданная программа должна быть протестирована с помощью Solana Program Test или локального тестового валидатора Solana. Во втором случае клиент должен быть написан на Rust. Тестирование подразумевает добавление в тестовую локальную сеть 4 перечисленных аккаунтов с некоторыми изначальными балансами, вызов программы и получение конечных балансов аккаунтов.

- Поддержка второй инструкции, по которой программа самостоятельно создаёт свои два аккаунты и переводит (*с некоторого переданного аккаунта payer*) на них сумму, указанную в инструкции (в инструкции должно быть указано две суммы для двух аккаунтов соответственно). После реализации этого пункта добавление в тестировании пары аккаунтов, принадлежащих программе, становится ненужным.

- Решение проблемы с целочисленным делением. Эту проблему предлагается осознать самостоятельно и самостоятельно подумать над решением

Комментарий

Приведённая выше логика позволяет упростить текущее задание, но в реальности Solana позволяет создавать внутри сети любые кастомные токены с любой кастомной механикой, соответственно выше описанная логика в реальности должна была бы работать с 8 аккаунтами: 2 PDA аккаунта, которые являются владельцами двух SPL аккаунтов, соответствующих кошелькам программы с *реально* разными токенами, 2 аккаунта принадлежащих пользователю и 2 SPL аккаунта, принадлежащих предыдущим соответственно. Также в тестировании было бы необходимо создать внутри локальной сети два своих токена, а на стороне on-chain программы поддержать создание SPL аккаунтов. Изучение механик, связанных с SPL Token и реализация такой логики может занять очень много времени, поэтому мы рекомендуем сфокусироваться на реализации без SPL Token, но при ***исключительно сильном желании и наличии большого количества свободного времени*** можно попробовать реализовать логику, приближенную к реальности.

Рекомендуемые материалы

- solana.com - на главной странице есть обучающие материалы
- helloworld - очень полезный пример
- paulx example, repo - очень подробный гайд о сложных механиках в Solana
- max-block repo - примеры программ, в readme ссылка на видео на русском
- solana-program-library/examples - примеры, предоставленные разработчиками Solana
- Крейты, которые понадобятся: Solana Program, Solana Program Test, Solana SDK, SPL Token, SPL Associated Token Account (для реализации предложенной нами логики должно быть достаточно первых трёх)