Entries

deposit
programs/cp-swap/src/lib.rs

instructions::deposit(

swap_base_input
programs/cp-swap/src/lib.rs

instructions::swap_base_input(ctx, amount_in, minimum_amount_out)

instructions

deposit
programs/cp-swap/src/instructions/deposit.rs

let pool_state = &mut ctx.accounts.pool_state.load_mut()?;
let (total_token_0_amount, total_token_1_amount) = pool_state.vault_amount_without_fee(

let results = CurveCalculator::lp_tokens_to_trading_tokens(
transfer_from_user_to_pool_vault(
transfer_from_user_to_pool_vault(

token_mint_to(

swap_base_input
programs/cp-swap/src/instructions/swap_base_input.rs

let pool_state = &mut ctx.accounts.pool_state.load_mut()?;
let result = CurveCalculator::swap_base_input(

transfer_from_user_to_pool_vault(
transfer_from_pool_vault_to_user(

pool

vault_amount_without_fee
programs/cp-swap/src/states/pool.rs

vault_0.checked_sub(self.protocol_fees_token_0 + self.fund_fees_token_0)

vault_1.checked_sub(self.protocol_fees_token_1 + self.fund_fees_token_1)

transfer_from_user_to_pool_vault
programs/cp-swap/src/utils/token.rs

token_2022::transfer_checked(

token_mint_to
programs/cp-swap/src/utils/token.rs

token_2022::mint_to(

curve

lp_tokens_to_trading_tokens
programs/cp-swap/src/curve/calculater.rs

ConstantProductCurve::lp_tokens_to_trading_tokens(

lp_tokens_to_trading_tokens
programs/cp-swap/src/curve/constant_product.rs

let (token_0_amount, token_1_amount) = match round_direction {

swap_base_input
programs/cp-swap/src/curve/calculater.rs

let destination_amount_swapped = ConstantProductCurve::swap_base_input_without_fees(

swap_base_input_without_fees
programs/cp-swap/src/curve/constant_product.rs

let numerator = source_amount.checked_mul(swap_destination_amount).unwrap();

let denominator = swap_source_amount.checked_add(source_amount).unwrap();

let destinsation_amount_swapped = numerator.checked_div(denominator).unwrap();