# Tutorial 6 – Tasks

## JSON RPC Practice

**Instructions:** Construct the data field for each JSON RPC request based on the provided ERC-20 interface functions.

function name() public view returns (string)

function symbol() public view returns (string)

function decimals() public view returns (uint8)

function totalSupply() public view returns (uint256)

function balanceOf(address \_owner) public view returns (uint256 balance)

function transfer(address \_to, uint256 \_value) public returns (bool success)

function transferFrom(address \_from, address \_to, uint256 \_value) public returns (bool success)

function approve(address \_spender, uint256 \_value) public returns (bool success)

function allowance(address \_owner, address \_spender) public view returns (uint256 remaining)

Refer back to lecture 4 slides on how to construct the “data” field of the request.

## Constant Product

* An AMM liquidity pool initially starts with 1000 TokenA and 1000 TokenB.
* What is the initial constant product (K)?
* Calculate the current price of TokenA in terms of Token B (price of TokenA).
* Now, imagine a trader buys 200 TokenA with TokenB. How will this trade affect the price of TokenA (increase, decrease, or no change)? Briefly explain why.
* After the trade, what is the new price of TokenA?

Refer back to lecture 5 slides for an explanation of how to work with the constant product formula of Uniswap.

## Project Integration

This tutorial is your opportunity to seek help in integrating the concepts from tutorials 1-5 to achieve the solution laid out in the project brief.