| 📄 tsconfig.json | Initial commit | 3 weeks ago |

📅 Review the assignment due date

Bitcoin Protocol Development - Week 3: Mine your first block

Overview

In this challenge, you are tasked with the simulation of mining process of a block, which includes validating and including transactions from a given set of transactions. The repository contains a folder `mempool` which contains JSON files. These files represent individual transactions. Your goal is to successfully mine a block by including some of these transactions, following the specific requirements outlined below.

Objective

Your primary objective is to write a script that processes a series of transactions, validates them, and then mines them into a block. The output of your script should be a file named `out.txt` that follows a specific format.

Place your solution in the appropriate directory based on your chosen language:

- [bash](#)
- [javascript](#)
- [python](#)
- [rust](#)

Requirements

Input

- You are provided with a folder named `mempool` containing several JSON files. Each file represents a transaction that includes all necessary information regarding the transaction.

Output

Your script must generate an output file named `out.txt` with the following structure:

- First line: The block header.
- Second line: The serialized coinbase transaction.
- Following lines: The transaction IDs (txids) of the transactions mined in the

block, in order. The first txid should be that of the coinbase transaction

Difficulty Target

The difficulty target is
`0000ffff00000000000000000000000000000000000000000000000000000000` .
This is the value that the block hash must be less than for the block to be
successfully mined.

Previous Block Hash

You can use any value for the previous block hash as long as it meets the
difficulty target.

Execution

To test your solution locally:

- Uncomment the line corresponding to your language in run.sh.
- Execute `local.sh` .

If your code works, you will see the test completed successfully.

Evaluation Criteria

Your submission will be evaluated based on:

- **Autograder**: Your code must pass the autograder test script.
- **Explainer Comments**: Include comments explaining each step of your

code.

- **Code Quality**: Your code should be well-organized, commented, and adhere to best practices.

Plagiarism Policy

Our plagiarism detection checker thoroughly identifies any instances of copying or cheating. Participants are required to publish their solutions in the designated repository, which is private and accessible only to the individual and the administrator. Solutions should not be shared publicly or with peers. In case of plagiarism, both parties involved will be directly disqualified to maintain fairness and integrity.

AI Usage Disclaimer

You may use AI tools like ChatGPT to gather information and explore alternative approaches, but avoid relying solely on AI for complete solutions. Verify and validate any insights obtained and maintain a balance between AI assistance and independent problem-solving.

Why These Restrictions?

These rules are designed to enhance your understanding of the technical aspects of Bitcoin. By completing this assignment, you gain practical experience with the technology that secures and maintains the trustlessness of Bitcoin. This challenge not only tests your ability to develop functional Bitcoin