Rust 区块链项目说明

1. 项目概述

本项目是一个**基于 Rust 语言的轻量级区块链系统**，采用 **PoW（工作量证明）** 机制进行挖矿，支持交易管理、区块存储、数据持久化，并提供 CLI 交互功能。

🔹 项目特点

- 轻量级 Rust 实现，适合学习和研究
- 支持 PoW 挖矿机制，确保区块链安全
- 交易系统存储在区块链中
- CLI 交互方式，提供便捷的管理操作
- 数据持久化存储到 JSON，支持加载与保存

---

2. 主要功能

功能	描述
CLI 交互	通过命令行操作区块链
挖矿机制	PoW 计算 `nonce` 直至满足难度
交易系统	存储 & 验证交易数据
数据持久化	以 JSON 方式存储区块数据
区块完整性检查	确保数据一致性和安全性

---

3. 目录结构

文件/目录	说明
`simple_blockchain/`	项目根目录
├── `src/`	源代码目录
│ ├── `main.rs`	入口文件，启动程序
│ ├── `block.rs`	定义区块结构
│ ├── `blockchain.rs`	处理区块链逻辑
│ ├── `transaction.rs`	交易结构与处理
│ ├── `pow.rs`	PoW 工作量证明挖矿
│ ├── `cli.rs`	命令行交互（CLI）
│ ├── `storage.rs`	数据存储（文件/数据库）

文件/目录	说明
├── Cargo.toml	Rust 依赖管理文件

4. 核心技术

- **语言**：Rust
- **共识机制**：PoW（Proof of Work）
- **哈希算法**：SHA-256
- **存储方式**：JSON 文件持久化
- **交互方式**：命令行（CLI）

5. 部分代码实现

Block 实现

```rust
/// 区块结构
#[derive(Serialize, Deserialize, Debug, Clone)]
pub struct Block {
    pub index: u64,
    pub timestamp: u128,
    pub previous_hash: String,
    pub hash: String,
    pub nonce: u64,
    pub transactions: Vec<Transaction>, // ✅ 交易数据
}
```

BlockChain 实现

```rust
impl Blockchain {
    /// 初始化区块链（创世区块）
    pub fn new(difficulty: usize) -> Blockchain {
        let genesis_block = Block::new(0, "0".to_string(), vec![], difficulty);
        Blockchain {
            chain: vec![genesis_block],
            difficulty,
        }
    }

    /// 添加新区块
    pub fn add_block(&mut self, transactions: Vec<Transaction>) {
        let previous_block = self.chain.last().unwrap();
        let new_block = Block::new(
            previous_block.index + 1,
            previous_block.hash.clone(),
            transactions,
            self.difficulty,
        );
        self.chain.push(new_block);
    }
}
```

Storage 实现

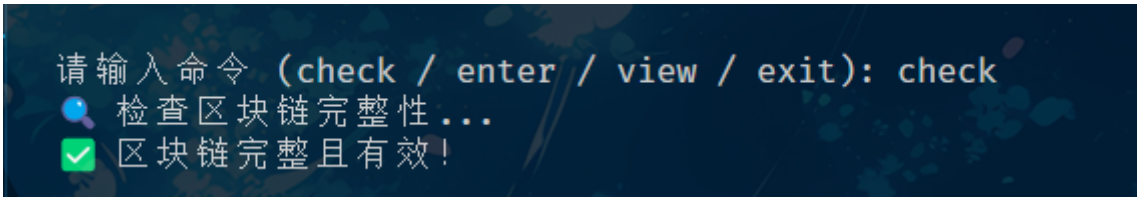
```rust
/// 保存区块链数据
pub fn save_blockchain(blockchain: &Blockchain, filename: &str) {
    let json = serde_json::to_string_pretty(blockchain).unwrap();
    fs::write(filename, json).expect("无法写入文件");
}

/// 读取区块链数据
pub fn load_blockchain(filename: &str) -> Blockchain {
    let data = fs::read_to_string(filename).expect("无法读取文件");
    serde_json::from_str(&data).unwrap()
}
```

6. 项目截图

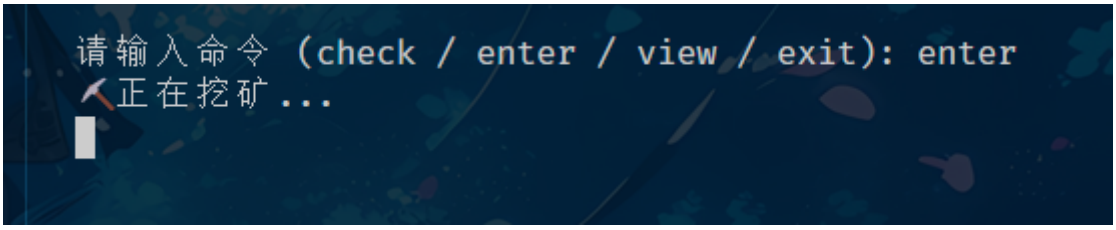
下面是项目的运行截图示例

CLI 交互

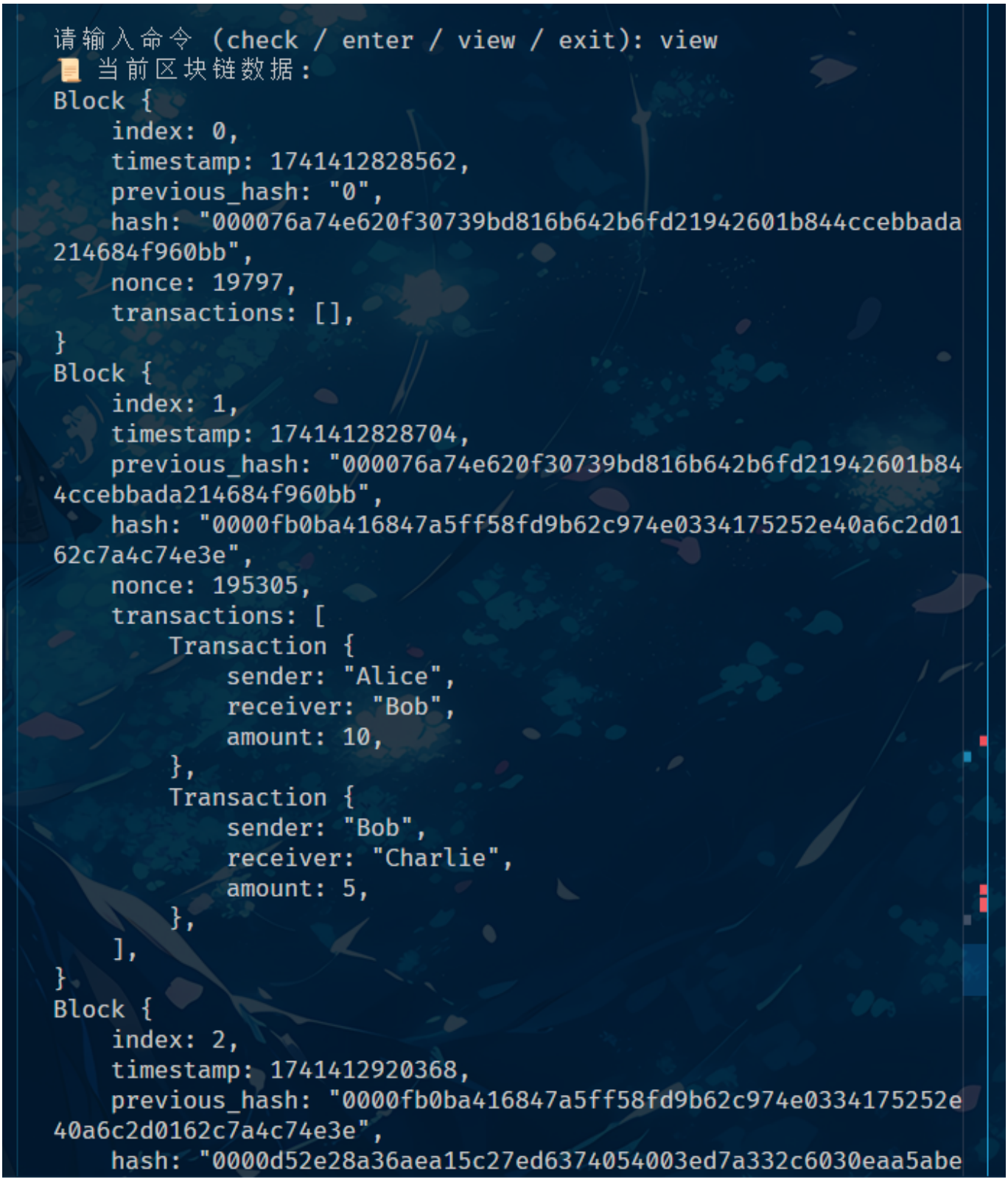
检查区块链 (check)



挖矿过程 (mining)



区块数据 (view)


请输入命令 (check / enter / view / exit): view
📜 当前区块链数据:
Block {
index: 0,
timestamp: 1741412828562,
previous_hash: "0",
hash: "000076a74e620f30739bd816b642b6fd21942601b844ccebbada
214684f960bb",
nonce: 19797,
transactions: [],
}
Block {
index: 1,
timestamp: 1741412828704,
previous_hash: "000076a74e620f30739bd816b642b6fd21942601b84
4ccebbada214684f960bb",
hash: "0000fb0ba416847a5ff58fd9b62c974e0334175252e40a6c2d01
62c7a4c74e3e",
nonce: 195305,
transactions: [
Transaction {
sender: "Alice",
receiver: "Bob",
amount: 10,
},
Transaction {
sender: "Bob",
receiver: "Charlie",
amount: 5,
},
],
}
Block {
index: 2,
timestamp: 1741412920368,
previous_hash: "0000fb0ba416847a5ff58fd9b62c974e0334175252e
40a6c2d0162c7a4c74e3e",
hash: "0000d52e28a36aea15c27ed6374054003ed7a332c6030eaa5abe

退出程序 (exit)


请输入命令 (check / enter / view / exit): exit
👋 退出程序...
~/tr/pr/simple_blockchain master ?11
>