Day 4: Programmable Finance

Smart Contracts, DeFi, and Tokenization

Joerg Osterrieder

Digital Finance Course

2025

Day 4 Overview

Day Purpose

This is where crypto meets finance. Learn how smart contracts enable DeFi protocols that replicate (and sometimes improve upon) traditional financial services without intermediaries.

4.1 Smart Contracts – Code as Agreement

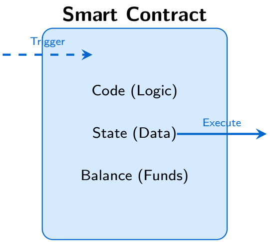
What is a Smart Contract?

Definition

A **smart contract** is a program stored on a blockchain that automatically executes when predetermined conditions are met.

Key Properties:

- **Deterministic**: Same input always produces same output
- **Immutable**: Once deployed, code cannot be changed
- **Transparent**: Anyone can verify the code
- **Self-executing**: No intermediary needed



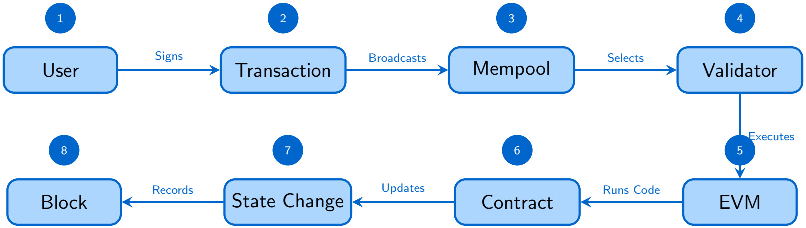
Traditional Contract vs. Smart Contract

Aspect	Traditional Contract	Smart Contract
Enforcement	Courts, lawyers	Code execution
Trust	Counterparties, institutions	Cryptographic verification
Execution	Manual, subject to delay	Automatic, instant
Amendment	Negotiation, paperwork	Requires new deployment
Cost	High (intermediaries)	Low (gas fees only)
Transparency	Private documents	Public, auditable code

Important Distinction

"Trustless" means you don't need to trust *counterparties*—but you still need to trust the *code*, the *blockchain*, and the *oracles*.

Smart Contract Execution Flow



1. User creates and signs transaction calling contract function
2. Transaction broadcast to network
3. Transaction waits in mempool
4. Validator selects transaction for block
5. Ethereum Virtual Machine (EVM) executes bytecode
6. Contract logic runs with provided inputs
7. State changes recorded (balances, storage)
8. Changes finalized in blockchain block

Anatomy of a Smart Contract (Solidity)

```
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.0;

contract SimpleEscrow {
    address public buyer;
    address public seller;
    uint256 public amount;
    bool public released;

    constructor(address _seller) payable {
        buyer = msg.sender;
        seller = _seller;
        amount = msg.value;
    }

    function release() external {
        require(msg.sender == buyer, "Only buyer can release");
        require(!released, "Already released");
        released = true;
        payable(seller).transfer(amount);
    }
}
```

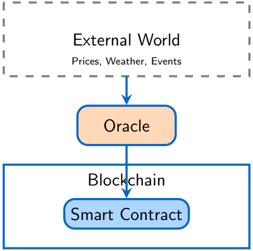
The Oracle Problem

The Challenge

Smart contracts cannot access external data—they only know what's on the blockchain.

Examples requiring oracles:

- Price feeds (ETH/USD)
- Weather data for insurance
- Sports scores for betting
- Real-world asset values



Solutions:

- Chainlink (decentralized)
- API3, Band Protocol
- Optimistic oracles (UMA)

Smart Contract Risks and Limitations

Technical Risks:

- **Bugs**: Code is immutable—bugs are forever
- **Reentrancy**: The DAO hack ($60M, 2016)
- **Integer overflow**: Pre-0.8 Solidity
- **Oracle manipulation**: Flash loan attacks
- **Front-running**: MEV extraction

Gas Considerations:

- Every operation costs gas
- Complex logic = expensive
- Storage is most expensive

Design Limitations:

- Cannot handle ambiguity
- No subjective judgments
- Cannot initiate actions
- Limited to on-chain data

The Immutability Paradox

Immutability provides security guarantees but makes bug fixes impossible. Solutions: proxy patterns, upgradeable contracts—but these reintroduce trust assumptions.

What "Trustless" Really Means

Trust Spectrum

Traditional	Smart Contract	Truly Trustless
Banks, lawyers, courts, counterparties	Code, auditors, oracles, validators	Mathematical proofs only

→ Decreasing Trust Requirements

Key Insight

Smart contracts shift trust from *institutions* to *code and cryptography*. This is valuable—but it's a different kind of trust, not the absence of trust.

Key Competency Check: Smart Contracts

You should be able to:

1. Explain how a smart contract executes on a blockchain
2. Define what “trustless” means (and doesn’t mean)
3. Identify key limitations: oracle problem, immutability risks
4. Read and understand the logic of a simple contract

Hands-on: NB08

NB08: Interact with a simple smart contract (token or escrow) on testnet—call functions and observe state changes.

4.2 DeFi Primitives – Lending, Trading, and Yield

What is DeFi?

Definition

Decentralized Finance (DeFi) refers to financial services built on public blockchains that operate without traditional intermediaries.

Core Principles:

- **Permissionless**: Anyone can participate
- **Non-custodial**: Users control their assets
- **Transparent**: All code and transactions public
- **Composable**: Protocols can be combined

DeFi Ecosystem (2024):

- Total Value Locked: $50B+
- Daily trading volume: $2B+
- Active protocols: 500+
- Supported chains: 50+

Major Categories:

- Decentralized Exchanges
- Lending Protocols
- Derivatives
- Yield Aggregators

DeFi vs. Traditional Finance

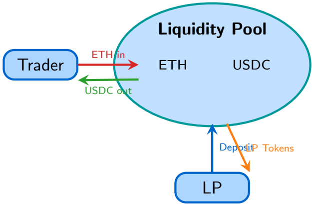
Feature	**Traditional Finance**	**DeFi**
Access	KYC, credit checks	Wallet address only
Hours	Business hours, T+2 settlement	24/7/365, instant
Custody	Institutions hold assets	User self-custody
Transparency	Private ledgers	Public blockchain
Innovation	Regulatory approval needed	Permissionless deployment
Risk	Counterparty, institution	Smart contract, oracle

The Composability Advantage

DeFi protocols are like “money legos”—they can be combined in ways their creators never anticipated. A flash loan can be used in an arbitrage that spans 5 different protocols in a single transaction.

Traditional Exchange:

- Order book with bids/asks
- Market makers provide liquidity
- Requires active management
- Centralized matching engine



AMM Innovation:

- No order book needed
- Liquidity pools replace market makers
- Algorithmic pricing
- Anyone can provide liquidity

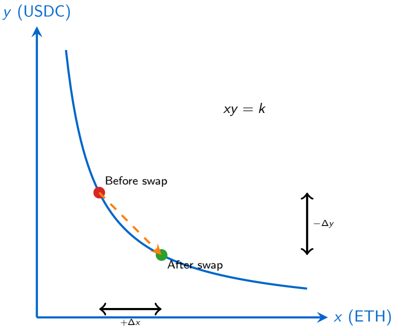
Key Protocols: Uniswap, SushiSwap, Curve, Balancer

Constant Product Formula: $x \cdot y = k$

The Core Equation

$$x \cdot y = k$$

- x = Token A reserves
- y = Token B reserves
- k = Constant (invariant)



Price Determination:

$$\text{Price of A in B} = \frac{y}{x}$$

After swap of Δx:

AMM Numerical Example

Initial Pool State

- 100 ETH + 300,000 USDC
- $k = 100 \times 300,000 = 30,000,000$
- Price: 1 ETH = 3,000 USDC

Trader swaps 10 ETH for USDC:

$$\text{New ETH reserves:} \quad x' = 100 + 10 = 110$$

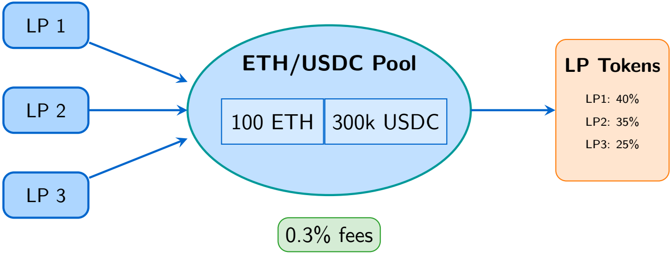
$$\text{New USDC reserves:} \quad y' = \frac{30,000,000}{110} = 272,727.27$$

$$\text{USDC received:} \quad \Delta y = 300,000 - 272,727.27 = 27,272.73$$

Price Impact Analysis

- Expected (no impact): $10 \times 3,000 = 30,000$ USDC
- Actual received: 27,272.73 USDC

Liquidity Pool Mechanics



LP Token Mechanics:

- LP tokens represent proportional claim on pool reserves
- Fees accumulate in pool, increasing LP token value
- Withdrawal returns proportional share of *current* reserves

Impermanent Loss Explained

Definition

Impermanent Loss (IL) is the difference between holding assets in a liquidity pool vs. simply holding them in your wallet.

Why it happens:

1. You deposit equal value: 1 ETH ($3,000) + 3,000 USDC
2. ETH price doubles to $6,000
3. Arbitrageurs rebalance the pool
4. Your LP position: 0.707 ETH + 4,243 USDC = $8,485
5. If you had just held: 1 ETH + 3,000 USDC = $9,000
6. **Impermanent Loss: $515 (5.72%)**

Key Insight

Loss is "impermanent" because if prices return to original levels, the loss disappears. It becomes *permanent* when you withdraw at different prices.

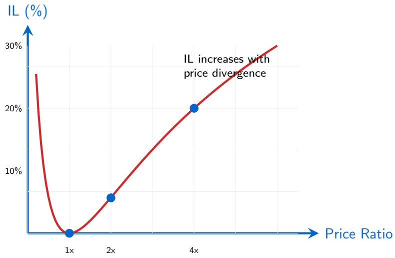
IL Formula:

$$IL = \frac{2\sqrt{r}}{1+r} - 1$$

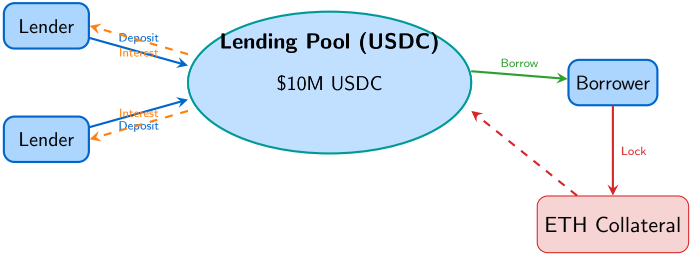
where $r = \frac{P_1}{P_0}$ (price ratio)

Price Change	IL
1.25x (25% up)	0.6%
1.50x (50% up)	2.0%
2x (100% up)	5.7%
3x (200% up)	13.4%
4x (300% up)	20.0%
5x (400% up)	25.5%

Mitigating IL:



DeFi Lending: How It Works



Key Mechanics:

- **Over-collateralization**: Borrow \$1,000 requires \$1,500+ collateral
- **Algorithmic rates**: Interest adjusts with utilization
- **Liquidation**: If collateral falls below threshold, anyone can liquidate

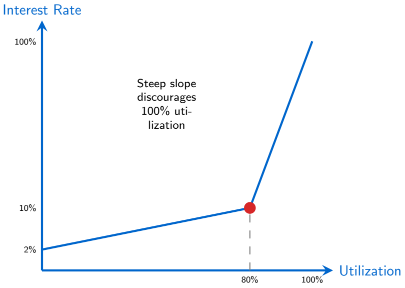
Algorithmic Interest Rates

Utilization Rate:

$$U = \frac{\text{Borrowed}}{\text{Supplied}}$$

Borrow Rate (kinked model):

$$R_{\text{borrow}} = \begin{cases} R_0 + U \cdot R_{\text{slope1}} & U < U_{\text{optimal}} \\ R_0 + U_{\text{opt}} \cdot R_1 + \\ \quad (U - U_{\text{opt}}) \cdot R_2 & U \geq U_{\text{optimal}} \end{cases}$$



Supply Rate:

$$R_{\text{supply}} = R_{\text{borrow}} \times U \times (1 - \text{fee})$$

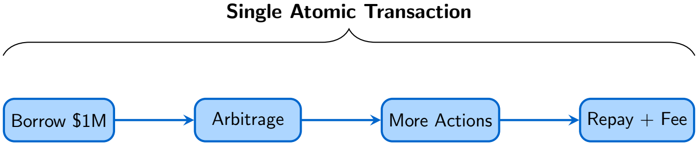
Key Protocols: Aave, Compound, MakerDAO

Hands-on: NB09 – Simulate AMM and observe impermanent loss

Flash Loans: Innovation or Attack Vector?

Definition

A **flash loan** is an uncollateralized loan that must be borrowed and repaid within a single transaction.



Legitimate Uses: Arbitrage, collateral swaps, self-liquidation
Attack Uses: Oracle manipulation, governance attacks, protocol exploits

The Double-Edged Sword

Flash loans democratize access to capital but have enabled over $500M in DeFi exploits.

Key Competency Check: DeFi Primitives

You should be able to:

1. Explain how an AMM prices assets using $x \cdot y = k$
2. Calculate price impact/slippage for a given trade
3. Understand how DeFi lending determines interest rates algorithmically
4. Calculate impermanent loss for price changes

Hands-on: NB09

NB09: Simulate a constant-product AMM—provide liquidity, execute swaps, observe price impact and impermanent loss.

4.3 Stablecoins – The Bridge Between Two Worlds

Why Stablecoins Matter

Definition

Stablecoins are cryptocurrencies designed to maintain a stable value relative to a reference asset (typically USD).

The Problem They Solve:

- Crypto volatility makes it unsuitable for payments
- Traditional banking hours and fees
- Need for on-chain "cash" in DeFi
- Cross-border payment friction

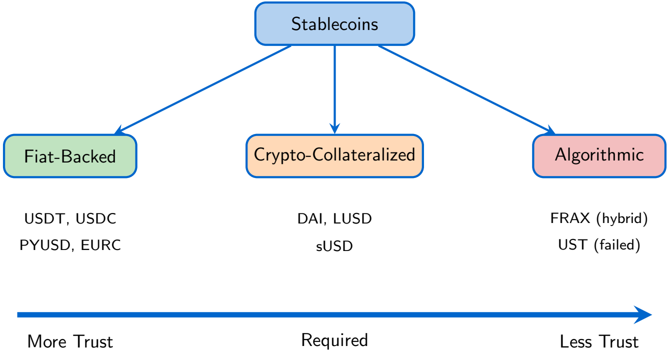
Market Size (2024):

- Total supply: \$170B+
- Daily volume: \$50B+
- Surpasses Visa in some metrics

Use Cases:

- Trading pairs on exchanges
- DeFi collateral and lending
- Remittances
- Savings in dollarized economies

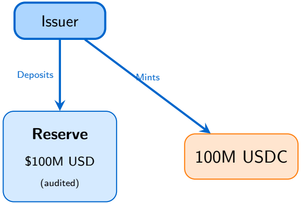
Stablecoin Design Taxonomy



Fiat-Backed Stablecoins

How It Works:

1. User sends $1 to issuer
2. Issuer holds $1 in reserves
3. Issuer mints 1 stablecoin
4. User can redeem anytime



Examples:

- **USDT** (Tether): $110B, largest
- **USDC** (Circle): $35B, regulated
- **PYUSD** (PayPal): Newest entrant

Risks:

- Centralized—can freeze funds
- Reserve quality concerns
- Regulatory uncertainty
- Banking system dependence

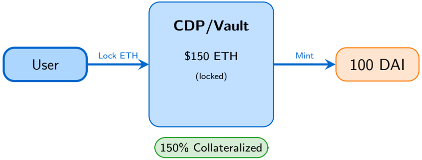
Crypto-Collateralized Stablecoins

How It Works:

1. User deposits \$150 ETH
2. Protocol mints 100 DAI
3. Collateral ratio: 150%
4. If ETH drops, liquidation

MakerDAO/DAI:

- Decentralized governance
- Multiple collateral types
- Stability fee (interest)
- Liquidation at 150%



Advantages:

- Decentralized
- Transparent reserves
- No counterparty risk

Disadvantages:

- Capital inefficient
- Liquidation risk
- Complexity

Algorithmic Stablecoins: The Experiment

Pure Algorithmic (Failed):

- No collateral backing
- Dual-token: stable + governance
- Expand supply when above peg
- Contract supply when below peg

The Death Spiral:

1. Price drops below peg
2. Redemptions spike
3. Confidence collapses
4. Governance token crashes
5. System becomes insolvent

Case Study: UST/LUNA (May 2022)

- Peak market cap: $18B
- Collapsed in 72 hours
- $60B total value destroyed
- Triggered crypto contagion

Hybrid Models (Surviving):

- **FRAX**: Partially collateralized
- Dynamic collateral ratio
- More resilient to de-peg

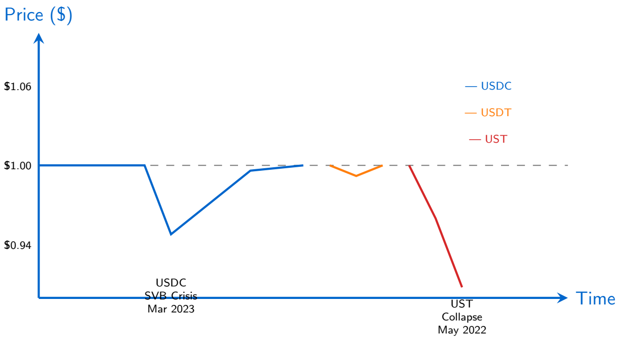
Stablecoin Design Comparison

Attribute	Fiat-Backed	Crypto-Collateral	Algorithmic
Collateral	Fiat in bank	Crypto (150%+)	None/Partial
Centralization	High	Medium	Low
Capital Efficiency	High (1:1)	Low (over-collateral)	High (0 collateral)
Scalability	Limited by reserves	Limited by collateral	Theoretically unlimited
Peg Stability	Strong	Strong	Weak
Regulatory Risk	High	Medium	Low
Censorship Risk	High	Low	Very Low

The Stablecoin Trilemma

You can optimize for two of three: **Decentralization**, **Stability**, **Capital Efficiency**

De-Peg Events: Historical Analysis



Key Lessons:

- Fiat-backed: Banking system dependencies (SVB, Silvergate)
- Algorithmic: Fundamental design flaws lead to death spirals
- All designs: Confidence is fragile and self-reinforcing

Stablecoin Regulation

Why Regulators Care:

- Systemic risk (too big to fail?)
- Consumer protection
- Money laundering concerns
- Monetary policy implications
- Bank-like activities

Regulatory Developments:

- EU: MiCA framework (2024)
- US: Congressional debate ongoing
- Singapore: Clear framework
- China: Banned

Key Requirements Emerging

- 1:1 reserve backing
- Regular audits/attestations
- Redemption guarantees
- Segregated reserves
- Licensing requirements

Impact on Market:

- USDC: Embracing regulation
- USDT: Offshore strategy
- DAI: Decentralization defense

Key Competency Check: Stablecoins

You should be able to:

1. Classify stablecoins by design type (fiat-backed, crypto-collateralized, algorithmic)
2. Explain the tradeoffs of each design approach
3. Assess de-peg risk factors for different stablecoins
4. Articulate why stablecoins face heavy regulatory scrutiny

Hands-on: NB10

NB10: Analyze stablecoin price stability data—examine de-peg events and compare resilience across designs.

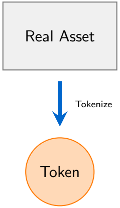
Hands-on: NB10 – Stablecoin price stability analysis

4.4 Tokenization and CBDCs

What is Tokenization?

Definition

Tokenization is the process of creating a digital representation of a real-world asset on a blockchain.



✓ Fractional ownership
✓ 24/7 trading
✓ Instant settlement
✓ Global access

What Can Be Tokenized:

- Real estate
- Securities (stocks, bonds)
- Commodities (gold, oil)
- Art and collectibles
- Intellectual property
- Carbon credits

Real-World Asset (RWA) Tokenization

Traditional

$1M minimum
T+2 settlement
Business hours only
Geographic restrictions
Multiple intermediaries

$\xrightarrow{\text{Tokenization}}$

Tokenized

$100 minimum
Instant settlement
24/7/365 trading
Global access
Direct P2P

Market Size Projections:

- Boston Consulting Group: $16 trillion by 2030
- BlackRock, JP Morgan actively building infrastructure
- US Treasuries on-chain: $1B+ (2024)

RWA Tokenization Examples

Asset Class	Example	Platform	Value Proposition
Real Estate	Tokenized apartments	RealT, Lofty	Fractional ownership, rental income
US Treasuries	T-bills on-chain	Ondo, Franklin Templeton	DeFi-compatible yield
Private Credit	Corporate loans	Centrifuge, Goldfinch	Access to institutional yields
Commodities	Gold tokens	Paxos Gold (PAXG)	Redeemable for physical
Art	Fractionalized art	Masterworks	Access to blue-chip art

The Legal Challenge

Tokens represent claims on assets, but enforcement still requires legal systems. "Code is law" doesn't apply when real-world assets need real-world courts.

Central Bank Digital Currencies (CBDCs)

Definition

A **CBDC** is a digital form of central bank money, denominated in the national unit of account and a direct liability of the central bank.

Global Status (2024):

- 130+ countries exploring
- 3 fully launched (Bahamas, Nigeria, Jamaica)
- 20+ in pilot phase
- Major economies in research

Two Main Types:

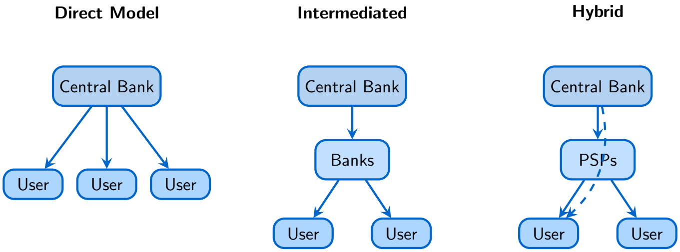
Retail CBDC:

- For general public
- Replaces/complements cash
- Direct central bank relationship

Wholesale CBDC:

- For financial institutions
- Interbank settlements
- Less disruptive to banking

CBDC Architecture Models



Trade-offs:

- Direct: Maximum control, but central bank becomes retail bank
- Intermediated: Preserves banking system, but less innovative
- Hybrid: Balance, but complex implementation

CBDC vs. Stablecoin Comparison

Attribute	CBDC	Stablecoin
Issuer	Central bank (sovereign)	Private company
Liability	Central bank balance sheet	Private balance sheet
Legal Status	Legal tender	Private money
Backing	Full faith of government	Reserves (varies)
Programmability	Policy-controlled	Open/permissionless
Privacy	Policy-dependent	Pseudonymous (public chains)
Innovation Speed	Slow (government)	Fast (private)
Interoperability	National focus	Global by default
Risk Profile	Sovereign risk only	Counterparty + operational

Key Question

Will CBDCs complement, compete with, or regulate away private stablecoins?

Programmable Money: The Convergence

What Programmability Enables:

- Conditional payments
- Automatic tax withholding
- Stimulus with expiration dates
- Supply chain financing
- Smart contract integration

Opportunities:

- Financial inclusion
- Reduced fraud
- Efficient policy transmission
- New business models

Concerns

- **Privacy**: Complete transaction surveillance
- **Control**: Money that "expires" or can be frozen
- **Exclusion**: Programmable discrimination
- **Security**: Single point of failure

Design Choices Matter:

- Token-based vs. account-based
- Privacy-preserving tech
- Offline capability

The Convergence Thesis: Traditional finance (TradFi) and decentralized finance (DeFi) are converging. The future is not "either/or" but hybrid systems combining the best of both.

Key Competency Check: Tokenization and CBDCs

You should be able to:

1. Explain the mechanics and value proposition of RWA tokenization
2. Compare different CBDC architectural models
3. Evaluate the tradeoffs between CBDCs and stablecoins
4. Assess the implications of programmable sovereign money

Discussion Questions

- Should central banks issue retail CBDCs? What are the risks?
- Will tokenization democratize access to investments or create new risks?
- How should programmable money be governed?

Day 4: Key Takeaways

4.1 Smart Contracts:

- Self-executing code on blockchain
- "Trustless" shifts trust, doesn't eliminate it
- Oracle problem limits external data access

4.2 DeFi Primitives:

- AMMs use $x \cdot y = k$ for pricing
- Impermanent loss affects liquidity providers
- Composability enables innovation (and attacks)

4.3 Stablecoins:

- Three designs: fiat-backed, crypto-collateral, algorithmic
- Stablecoin trilemma constrains design
- Heavy regulatory focus globally

4.4 Tokenization & CBDCs:

- RWA tokenization unlocks liquidity
- CBDCs represent sovereign response
- Programmable money raises privacy concerns

Hands-On Notebooks for Day 4

Notebook	Topic	Key Activities
NB08	Smart Contract Interaction	Interact with contracts on testnet, call functions, observe state changes
NB09	AMM Simulation	Provide liquidity, execute swaps, measure impermanent loss
NB10	Stablecoin Analysis	Analyze price stability data, examine de-peg events

Preparation for Day 5

Tomorrow we explore the **AI-Finance Intersection**: Machine learning for markets, LLMs in finance, and the automation of financial decisions.

Questions?

Day 4: Programmable Finance

Smart Contracts, DeFi, and Tokenization

Questions & Discussion

Next: Day 5 – AI and Finance