Quantstamp

Secured Finance

Executive Summary

This audit report was prepared by Quantstamp, the leader in blockchain security.

Type	Lending Protocol
Timeline	2023-10-16 through 2023-11-30
Language	Solidity
Methods	Architecture Review, Unit Testing, Functional Testing, Computer-Aided Verification, Manual Review
Specification	Public Documentation Secured Finance Blog
Source Code	• Secured-Finance/contracts #92f4e85
Auditors	• Guillermo Escobero Auditing Engineer • Valerian Callens Senior Auditing Engineer • Mustafa Hasan Senior Auditing Engineer

Documentation quality	Medium
Test quality	Medium
Total Findings	45 Fixed: 23 Acknowledged: 17 Mitigated: 5
High severity findings	4 Fixed: 4
Medium severity findings	10 Fixed: 6 Acknowledged: 3 Mitigated: 1
Low severity findings	16 Fixed: 4 Acknowledged: 9 Mitigated: 3
Undetermined severity findings	4 Fixed: 2 Acknowledged: 2
Informational findings	11 Fixed: 7 Acknowledged: 3 Mitigated: 1

Summary of Findings

Quantstamp performed a security audit of the smart contracts implementing the Secured Finance lending protocol based on the code present in the listed repositories.

Secured Finance operates as a decentralized finance (DeFi) platform, addressing liquidity challenges within the industry. The platform's protocol facilitates peer-to-peer lending and derivatives trading, emphasizing fixed-income investments and hedging. This solution aims to offer a more efficient and cost-effective alternative to conventional financial institutions.

The protocol integrates lending markets, drawing inspiration from bond markets. Users can place lending and borrowing orders, mirroring the process of buying or selling zero-coupon bonds with varying maturity periods. Notably, these orders are maintained within an on-chain order book, eliminating the necessity for additional systems or privileged roles for order matching.

All issues and recommendations are discussed in the *Findings* section of this document. After that, recommendations about documentation and best practices are discussed. We strongly recommend addressing all the issues before deployment.

High and medium-severity issues were found, mainly related to missing validations and their potential misuse by privileged addresses, which can lead to blocked tokens or massive liquidations. The allowed operations of privileged roles controlled by the Secured Finance team are discussed in "Privileged Roles And Ownership", as well as its ability to upgrade any contract at any time ("Upgradability").

The documentation quality is good. Public and internal documentation was provided by the Secured Finance team. However, it is recommended to add detailed and updated public documentation focusing on critical parts of the protocol, as some pages are outdated and not detailed enough. Some examples could be lazy evaluation, haircuts, privileged accounts, and a list of addresses of the smart contracts deployed. Refer to the "Adherence to Specification" section for more details.

Regarding testing, all tests passed, and the project implements code coverage metrics. It shows `76%` of branch coverage. We highly recommend improving the branch coverage to a minimum of `95%` and adding new tests to cover the proposed fixes.

Fix review: The Secured Finance team has either fixed, mitigated, or acknowledged all issues found within the report, and provided a new commit containing fixes for the issues found. For the mitigated or acknowledged issues, we recommend considering completely fixing them and taking them into account when configuring or making changes to the protocol configuration, to avoid unintended behavior or liquidations.

The auditing team found that the tests were improved during the fix review phase. The test suite was improved and the project shows an `87%` branch coverage. We recommend that the branch coverage be raised to at least `95%` and ideally as close to `100%` as possible.

ID	DESCRIPTION	SEVERITY	STATUS
SF-1	**Unaccounted Funds Due to Unprotected `LendingMarket.cleanUpOrders()`**	● High	Fixed
SF-2	**Issue removed: false positive**	● Undetermined	Fixed
SF-3	**Forced Repayment Is Possible with Currencies that Are Not Supported by the Platform**	● High	Fixed
SF-4	**Access Control Mechanism Based on a List of `acceptedContracts` Can Ultimately Lead to Draining Funds**	● High	Fixed
SF-5	**Anyone Can Pause the `ReserveFund` Liquidation Coverage**	● High	Fixed
SF-6	**Modifying System Parameters Can Lead To Dangerous Behavior**	● Medium	Acknowledged
SF-7	**The Mapping `TokenVault.tokenAddresses` Can Be Updated, Making Funds at Risk if the Owner of the Contract Is Compromised**	● Medium	Fixed
SF-8	**Currencies Can Be Unregistered From `TokenVault`, Which Could Lead to Funds Locked**	● Medium	Mitigated
SF-9	**Market Termination Is Impossible if a Token Has No Market or Is Paused**	● Medium	Fixed
SF-10	**The Roles in the System Can Be Temporarily or Indefinitely Lost**	● Medium	Fixed
SF-11	**The Modifier `onlyAcceptedContracts()` Can Be Bypassed After an Update of the Mapping of `AddressResolver`**	● Medium	Acknowledged
SF-12	**Auto-Rolling Price May Not Be Precise**	● Medium	Fixed
SF-13	**The Emergency Settlement Mechanism Can Be Negatively Impacted by Updating Price Feed Decimals**	● Medium	Fixed
SF-14	**The Emergency Termination Mechanism Can Be Negatively Impacted by Incorrect Prices Sent by the Oracle**	● Medium	Acknowledged
SF-15	**Precision Loss in Itayose Process Makes `FutureValueVault` Unstable**	● Medium	Fixed
SF-16	**Missing Input Validation**	● Low	Mitigated
SF-17	**Incomplete Checks when Updating a Price Feed in the `CurrencyController`**	● Low	Fixed
SF-18	**Possible Reentrancy in Liquidation Logic**	● Low	Acknowledged
SF-19	**Added Orderbooks Do Not Operate with Their Initial Information**	● Low	Acknowledged
SF-20	**Deletion of Tree Nodes Does Not Work as Intended**	● Low	Acknowledged
SF-21	**Haircut and Pricefeed Data Not Erased when a Currency Is Removed**	● Low	Acknowledged

ID	DESCRIPTION	SEVERITY	STATUS
SF-22	Upgradability	● Low ⓘ	Acknowledged
SF-23	Privileged Roles and Ownership	● Low ⓘ	Acknowledged
SF-24	Critical Roles Not Transferred Following a Two-Step Ownership Transfer Pattern	● Low ⓘ	Acknowledged
SF-25	Ownership Can Be Renounced	● Low ⓘ	Acknowledged
SF-26	The Libraries `FullMath`, `RoundingInt256`, and `RoundingUint256` Can Revert Due to Overflows	● Low ⓘ	Mitigated
SF-27	Partial Imports of Addresses in `AddressResolver` Are Possible	● Low ⓘ	Mitigated
SF-28	Genesis Value Balance Incorrectly Updated	● Low ⓘ	Fixed
SF-29	Users Can Cancel Orders Anytime	● Low ⓘ	Fixed
SF-30	`TokenVault` Is a Greedy Contract	● Low ⓘ	Fixed
SF-31	Protocol Does Not Support Non-Standard Tokens	● Low ⓘ	Acknowledged
SF-32	'Dead' Code	● Informational ⓘ	Fixed
SF-33	Ignoring Values Returned by Functions Is Not Always Explicitly Justified	● Informational ⓘ	Acknowledged
SF-34	Long Contracts Names	● Informational ⓘ	Acknowledged
SF-35	Unlocked Pragma	● Informational ⓘ	Fixed
SF-36	Wrong Formula In Function Comment	● Informational ⓘ	Fixed
SF-37	Undocumented Constants	● Informational ⓘ	Fixed
SF-38	Deprecated Function	● Informational ⓘ	Fixed
SF-39	Wrong Constant Used	● Informational ⓘ	Fixed
SF-40	Mismatch Between Code and Documentation when Inserting Nodes Into the Order Book	● Informational ⓘ	Fixed
SF-41	Events Emitted Even if No Update Is Performed	● Informational ⓘ	Mitigated
SF-42	Transactions Not Reverting Even if the State Is Not Modified	● Informational ⓘ	Acknowledged
SF-43	Possible DoS via High Gas Usage During Liquidations	● Undetermined ⓘ	Acknowledged
SF-44	The Purpose of Being Able to Remove a Price Feed Is Unclear	● Undetermined ⓘ	Fixed
SF-45	Risks Related to Errors Involving the Price Feeds of a Subset of Supported Currencies	● Undetermined ⓘ	Acknowledged

Assessment Breakdown

Quantstamp's objective was to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices.

Disclaimer
Only features that are contained within the repositories at the commit hashes specified on the front page of the report are within the scope of the audit and fix review. All features added in future revisions of the code are excluded from consideration in this report.

Possible issues we looked for included (but are not limited to):

- Transaction-ordering dependence
- Timestamp dependence
- Mishandled exceptions and call stack limits
- Unsafe external calls
- Integer overflow / underflow
- Number rounding errors
- Reentrancy and cross-function vulnerabilities
- Denial of service / logical oversights
- Access control
- Centralization of power
- Business logic contradicting the specification
- Code clones, functionality duplication
- Gas usage
- Arbitrary token minting

Methodology

1. Code review that includes the following
    1. Review of the specifications, sources, and instructions provided to Quantstamp to make sure we understand the size, scope, and functionality of the smart contract.
    2. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
    3. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Quantstamp describe.
2. Testing and automated analysis that includes the following:
    1. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
    2. Symbolic execution, which is analyzing a program to determine what inputs cause each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarity, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, and actionable recommendations to help you take steps to secure your smart contracts.

Scope

Files Included

Repository: `https://github.com/Secured-Finance/contracts/ @ 92f4e85370f2804438e08995ab9199ad137805d3`

Files:
- `contracts/protocol`
- `contracts/migration`

Files Excluded

Files:
- `contracts/external`

Findings

SF-1
Unaccounted Funds Due to Unprotected `LendingMarket.cleanUpOrders()`

● High ⓘ Fixed

Update
Access control added to `cleanUpOrders()` with a new modifier `onlyLendingMarketController`. Addressed in: `626fe5bcfa2ea080a8d70f2bd9fe2949963349b2` and `bd222dcae6c3d88d2975fc5860204e7df573cb4d`.

File(s) affected: `LendingMarket.sol`

Description: The Secured Finance protocol implements a lazy evaluation logic, where certain data structures are left unupdated after specific operations to mitigate gas costs. Notably, in the execution of an order, the array containing a user's active (unfilled) orders for a specific order book is not updated within the transaction. This array is only updated in the subsequent transaction before any user-related operations.

The critical function at the center of this issue is `LendingMarket.cleanUpOrders()`. This function calls `OrderActionLogic.cleanUpOrders()`, responsible for updating the unfilled orders of a user for a given order book, specifically `orderbook.activeLendOrderIds[user]` and `orderbook.activeBorrowOrderIds[user]`. This function edits these arrays and expects that the caller will update the removed funds. If not, these funds are not moved and will be lost. The function `FundManagementLogic._cleanUpOrders()` shows how the different vaults should be updated with the amounts returned by `LendingMarket.cleanUpOrders()`.

Importantly, this function lacks protection, allowing anyone to call it. While it successfully updates those arrays based on the order book trees, the funds filled of the removed orders from the array are not appropriately tracked or accounted for.

Recommendation: To address this vulnerability and prevent unaccounted funds, it is recommended to:

1. Protect `LendingMarket.cleanUpOrders()` to be callable only from accepted contracts.
2. Restrict the invocation of `LendingMarket.cleanUpOrders()` to the flow of `LendingMarketController.cleanUpFunds()` so that filled orders are accurately accounted for within the system.

SF-2 Issue removed: false positive

● Undetermined ⓘ Fixed

Description: Issue removed: false positive

SF-3 Forced Repayment Is Possible with Currencies that Are Not Supported by the Platform

● High ⓘ Fixed

Update

The liquidation logic validates that the collateral currency passed as argument is a collateral currency registered in `TokenVault`. Addressed in: `337113529e81f2f41ea2dcfbb73402b58efed5f0`.

File(s) affected: `CurrencyController.sol`, `LiquidationLogic.sol`, `TokenVault.sol`

Description: A liquidator may be able to repay a position either in a token that used to be accepted as collateral by the protocol or a token that used to be accepted for trading. When a liquidator calls `executeForcedRepayment()`, the function attempts forced repayment on behalf of a user if the user defaulted on a borrowing position. However, the function performs no validation on the supplied `_collateralCcy`, allowing repayment to take place even if the collateral currency is no longer accepted by the protocol, whether as collateral or as a trading token.

This is possible because the `TokenVault` is responsible for validating whether a currency is allowed as collateral. However, when `LiquidationLogic._transferCollateral()` calls `TokenVault.transferFrom()`, which is protected by the `onlyRegisteredCurrency` modifier, the check in `isRegisteredCurrency()` only validates that `Storage.slot().tokenAddresses[_ccy] != address(0)`. This will irreversibly be true for any currency that is added via `TokenVault.registerCurrency()` since the function call adds the passed address to the `tokenAddresses` mapping, however calling `TokenVault.updateCurrency()` will only remove the `_ccy` entry from `collateralCurrencies`.

Additionally, during the function's execution, it calls `CurrencyController.convert()` to get the `liquidationAmountInCollateralCcy`. The function call then lands on `CurrencyController._getAggregatedLastPrice()`, which will return `1` if the currency has no pricefeeds or the actual price of the currency in case it was previously accepted as collateral (see `Haircut and pricefeed data not erased when a currency is removed` for more context).

Recommendation: Check if the supplied currency is still accepted as collateral before carrying the repayment out.

SF-4 Access Control Mechanism Based on a List of `acceptedContracts` Can Ultimately Lead to Draining Funds

● High ⓘ Fixed

Update

The accepted contracts pattern was removed. Now functions are protected with a new modifier `onlyLendingMarketController`. Also `TokenVault` does not have privileged access to `LendingMarketController` functions anymore. Addressed in `52bc9d3a009856f51eee99924bda3a75a01498a5` and `bd222dcae6c3d88d2975fc5860204e7df573cb4d`.

File(s) affected: `TokenVault.sol`

Description: Several contracts use a list of names (`acceptedContracts`) as well as a modifier `onlyAcceptedContracts` to restrict some of their functions to only a subset of addresses. The contract `TokenVault` that "accepts" the contracts `LendingMarketController` and

`ReserveFund`. In parallel, the contract `ReserveFund` inherits the contract `MixinWallet`, which lets the owner of that contract execute arbitrary calls to arbitrary addresses with arbitrary payloads via the functions `MixinWallet.executeTransaction()` and `MixinWallet.executeTransactions()`. As a consequence, the owner of `ReserveFund` can execute the following restricted functions of the contract `TokenVault`:

- `depositFrom()` where any available amount of token allowed to be spent by the protocol can be transferred from users' wallets to the protocol;
- `addDepositAmount()` where the deposit balance of a registered currency of any user can be increased by any amount;
- `removeDepositAmount()` where the deposit balance of a registered currency of any user can be decreased by any available amount;
- `executeForcedReset()` where the deposit balance of a registered currency of any user can be set to 0;
- `transferFrom()` where the deposit balance of a registered currency of any user can be transferred to any other user;

After executing the exploit scenario described below, one address with a non-zero balance can withdraw funds from the system via the function `withdraw()`.

The root cause of this issue is the fact that the access control mechanism based on a list of `acceptedContracts` does not respect the Principle of Least Privilege. Several contracts requiring access to restricted functions of a third contract for separate use cases both obtain access to all restricted functions of that third contract.

Exploit Scenario:
1. Attacker gets access to the owner address of the contract `ReserveFund`.
2. Attacker executes batches of transactions via `MixinWallet.executeTransactions()` to perform the following actions: a. `TokenVault.depositFrom()` to transfer any amount allowed to be spent by the contract `TokenVault` directly from users' wallets. b. `TokenVault.addDepositAmount()` to set the deposit balance of Attacker to the exact total amount of funds owned by the contract `TokenVault`.
3. Attacker executes `TokenVault.withdraw()` to withdraw all tokens owned by the contract `TokenVault`.

Recommendation: Consider distinguishing the functions allowed to be executed by the contracts `LendingMarketController` and `ReserveFund`.

SF-5 Anyone Can Pause the `ReserveFund` Liquidation Coverage

● High ⓘ Fixed

Update

Functions `pause()` and `unpause()` are now protected by `onlyOperator`. Addressed in `af435086b82299f4f630e8554b875fdd3d19d3c9`.

File(s) affected: `ReserveFund.sol`

Description: `ReserveFund` contract implements a pausing system through a boolean variable `paused` and a public function `isPaused()` that returns it. Functions `pause()` and `unpause()` toggle `paused` to `true` and `false`.

Functions `pause()` and `unpause()` are both unprotected by any access modifiers. This allows an attacker to pause and unpause the `ReserveFund` contract as they please. Whether the contract is paused is checked by `LiquidationLogic.executeLiquidation()` and depending on the result, different calculations are made to cover insolvent positions, which may allow an attacker to force improper calculations while their positions are being liquidated by front-running the liquidation transaction to pause/unpause the contract.

Recommendation: `ReserveFund.pause()` and `ReserveFund.unpause()` must be protected to be called only by an administrative role of the Secured Finance team.

SF-6 Modifying System Parameters Can Lead To Dangerous Behavior

● Medium ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client. The client provided the following explanation:

> *We will notify users & community about system parameter updates before we execute them. Also, we will use multi-sig wallet as our contract owner, so nobody can update them alone.*

File(s) affected: `TokenVault.sol`

Description: Administrators of the Secured Finance protocol can modify a set of parameters that can lead to liquidations and market instability:
1. `TokenVaultStorage.slot().collateralCurrencies` set is queried when calculating the available collateral of users. This set can be modified by the owner of `TokenVault`, calling `TokenVault.updateCurrency()`. If this administrator removes a currency from the set, this currency will not be used in the collateral calculation, and it will be worth zero (in user positions and deposited funds), potentially leading to a cascade of user liquidations.
2. Updating haircut values can uncover the positions of some users.

3. If the contract `ReserveFund` is not paused and the liquidated user is insolvent, `ReserveFund` can cover in some conditions a portion of the amount transferred to the liquidator. If replicable, this process could lead to draining funds from the contract `ReserveFund`. This is expected to be protected by the Circuit Breaker and minimum debt price modules, and rapid liquidation executions. Protocol operators should be careful when updating the threshold and minimum debt price values. Still, adding more tests of that specific process would help to make sure it is not possible to execute this scenario, by considering several edge cases and extreme conditions.

Recommendation: Before removing a collateral currency from the system, updating the haircut rate of a currency, or modifying any other critical parameter, the Secured Finance team should consider the side effects of it, based on the users' positions at that point, and the collateral deposited in the system. This operation should be performed only when it is strictly necessary.

SF-7
The Mapping `TokenVault.tokenAddresses` Can Be Updated, Making Funds at Risk if the Owner of the Contract Is Compromised

● Medium ⓘ Fixed

Update
Now it is verified that the currency identifier is not registered to avoid changing its address. Addressed in: `7abd3d76a14272b581dfa95855ee93b7d00ede73`.

File(s) affected: `TokenVault.sol`

Description: Several functions in the contract `TokenVault` only accept to work for currencies registered in the mapping `tokenAddresses`, based on the modifier `onlyRegisteredCurrency()`. The owner of the contract can register new currencies via the function `registerCurrency()`. Only currencies with an existing identifier registered in the contract `CurrencyController` can be registered.

However, the fact that an address matches a given currency identifier is not checked. Also, the association between a currency identifier and an address can be updated by calling twice the function `registerCurrency()` for the same identifier by a different address.

If the owner of the contract gets compromised, the exploit scenario below becomes possible.

Exploit Scenario: Initially:

- `TokenVault.tokenAddresses[WETH_identifier] = WETH_address`.
- The deposit amount of Attacker in `WETH` is `0`.
- There is an amount of `A WETH` in `TokenVault`.

Steps:

1. Attacker gets access to the owner address of the contract `TokenVault`.
2. Attacker updates the address associated with `WETH_identifier` to the address of a random token `B` `TokB_address`.
3. Attacker deposits an amount `A` of token `B` via `TokenVault.deposit()`.
4. Attacker updates back the address associated with `WETH_identifier` to the address of `WETH_address`.
5. Attacker withdraws `A WETH` from the system via `TokenVault.withdraw()`.

Recommendation: Consider ensuring that the association between a currency identifier and its associated address cannot be changed once set. Also, an on-chain mechanism could be created to derive the currency identifier (`bytes32`) from the address to make sure that it is not possible to have two different addresses for the same currency identifier, and vice versa.

SF-8
Currencies Can Be Unregistered From `TokenVault`, Which Could Lead to Funds Locked

● Medium ⓘ Mitigated

Update
A zero address validation was added for the first point. However, it is still technically possible to have two different `bytes32` identifiers pointing to the same token address.

Update
Addressed in: `6f098ff77c70847165cce1efdcbdb5c72ae0c279`. The client provided the following explanation:

> *The currency identifier from the token address can be used, but another logic for ETH is also needed to do that. So to keep our protocol simple, we will not support this solution.*

File(s) affected: `TokenVault.sol`

Description: Several functions in the contract `TokenVault` only accept to work for currencies registered in the mapping `tokenAddresses`, based on the modifier `onlyRegisteredCurrency()`. The owner of the contract can register new currencies via the function `registerCurrency()`. Only currencies with an existing identifier registered in the contract `CurrencyController` can be registered.

However, it is possible to unregister a currency by setting it to the `address(0)`. This can lead to:

1. The owner can call this function with the wrong address, halting the deposit and withdrawal system of the protocol. All transfer calls to this wrong address will revert.
2. The address passed can be different from the one listed in `CurrencyController`.
3. Duplicated `CurrencyRegistered` events.

Exploit Scenario: Initially:

- `TokenVault.tokenAddresses[WETH_identifier] = WETH_address`.
- User has a deposit amount of `A WETH`.

Steps:

1. Owner updates the address associated with `WETH_identifier` to `address(0)`.
2. User cannot withdraw `WETH` anymore from the system via `TokenVault.withdraw()`, due to the modifier `onlyRegisteredCurrency`.

Recommendation:

1. `TokenVault.registerCurrency()` should validate `_tokenAddress` to be not zero, and should revert if `Storage.slot().tokenAddresses[_ccy]` is already set. Owner can add the currency as collateral in the specific use case for it, calling `TokenVault.updateCurrency()`.
2. Consider ensuring that the association between a currency identifier and its associated address cannot be changed once set. As mentioned in the previous finding, an on-chain mechanism could be created to derive the currency identifier (bytes32) from the token address.

SF-9
Market Termination Is Impossible if a Token Has No Market or Is Paused

● Medium ⓘ Fixed

Update

`pauseLendingMarket()` is not called in the emergency termination loop anymore. Addressed in: `db485121d37c7939555352acdfac51ac191f0f9d`.

The client provided the following explanation:

> *Market pausing was not needed for `executeEmergencyTermination()`, so it was removed.*

File(s) affected: `LendingMarketOperationLogic.sol`

Description: The `LendingMarketOperationLogic.executeEmergencyTermination()` function loops over the currencies in the protocol and invokes `pauseLendingMarkets()` on each of them, which looks like the following:

```
function pauseLendingMarkets(bytes32 _ccy) public {
    ILendingMarket market = ILendingMarket(Storage.slot().lendingMarkets[_ccy]);
    market.pauseMarket();
}
```

However, this function call will revert if a currency has no market or if the market is already paused (`_pause()` has `whenNotPaused` modifier), reverting the emergency termination trigger.

Recommendation: Check if a currency `_ccy` has a market that is not already paused before calling `pauseLendingMarkets()`.

SF-10
The Roles in the System Can Be Temporarily or Indefinitely Lost

● Medium ⓘ Fixed

Update

It is not possible anymore to lose the `DEFAULT_ADMIN_ROLE` role. However, the Secured Finance team acknowledged that the system can remain temporarily without an operator. Addressed in: `fda2f7389ce13248680fcf538bbf5ad7e1975f70`.

The client provided the following explanation:

> *Now, our protocol never loses the role `DEFAULT_ADMIN_ROLE` by the following update.*
> - *Remove `setRoleAdmin()`.*
> - *Override and disable `renounceRole()`.*
> - *Make `revokeRole()` not work for own wallet The protocol may lose the role `OPERATOR_ROLE`, but it can be assigned by users who have the role `DEFAULT_ADMIN_ROLE`.*

File(s) affected: `MixinAccessControl.sol`

Description:

1. Addresses with the role `OPERATOR_ROLE` can pause and unpause `LendingMarkets` if required. According to the logic in the contract `MixinAccessControl`, addresses with the role `DEFAULT_ADMIN_ROLE` can add or remove operators via the functions `addOperator()` and `removeOperator()`. However, it is possible to temporarily remain without any address having the role `OPERATOR_ROLE`, which would make it impossible to pause a `LendingMarket` in case of emergency.
2. Addresses with the role `DEFAULT_ADMIN_ROLE` can assign or revoke the roles `OPERATOR_ROLE` and `DEFAULT_ADMIN_ROLE`. However, it is possible to remain forever without any address having the role `DEFAULT_ADMIN_ROLE`, which would make it impossible to update the current assignments of Operators.

Recommendation: Consider making sure that the system cannot remain without an address with the role `OPERATOR_ROLE`. Same for the role `DEFAULT_ADMIN_ROLE`.

SF-11
The Modifier `onlyAcceptedContracts()` Can Be Bypassed After an Update of the Mapping of `AddressResolver`

• Medium ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client. The client provided the following explanation:

> *We use a multi-sig wallet as all contract's owners. In addition, this access control mechanism is updated and follows the Principle of Least Privilege by the update of SF-4.*

File(s) affected: `MixinAddressResolver.sol`

Description: The modifier `onlyAcceptedContracts()` is used by several contracts to restrict some of their operations to a list of addresses identified by their associated name registered in the mapping of the contract `AddressResolver`. This mapping can be updated by the owner of this contract. As a result, if the owner's address is compromised or if it becomes malicious, an authorized address corresponding to an accepted contract could be replaced by a malicious address which would get the right to execute restricted operations. The likelihood is low but the negative impact for the protocol would be high.

Recommendation: Consider using a multi-sig to make sure that compromising one address is not enough to update the mapping of the contract `AddressResolver`. This finding should be addressed as part of the finding describing the roles and privileges of the system.

SF-12 Auto-Rolling Price May Not Be Precise

• Medium ⓘ Fixed

Update

Addressed in: `61da5eb9667f5198d9a8419199e7ffde30e2ded5`. The client provided the following explanation:

> *The updated points are as follows:*
> - *Change the auto-roll price discovery mechanism to use the last block unit price instead of the last order unit price.*
> - *Change the storage data, `lastOrderBlockNumber`, in `OrderBookLib` to `lastOrderTimestamp` for the mechanism change above.*

File(s) affected: `LendingMarketOperation.sol`

Description: Each time an order is executed, its unit price is saved in the contract. This will be used to calculate the auto-roll price for the next market when the current market is rotated.

The function `updateOrderLogs()` is called when an order is executed (at the moment of updating the taker's balance). In a liquid and active market, it will save the present and future value of all orders in the last 6 hours of the market (observation period, starting 6 hours before maturity). This way, the volume-weighted average unit price can be calculated later.

However, to protect against the scenario of a market with no order executions in this observation period, the system also saves the `estimatedAutoRollUnitPrice`: the last executed price (converted to the new maturity). This price is recorded without verification: a small order of a few wei can set the next auto-rolling price.

The system also implements data structures to save the weighted average unit price on a block, more secure and precise as it performs additional checks like a minimum reliable amount (minimum amount traded in a block to avoid price manipulation).

Recommendation: Consider getting the unit price of the last reliable block to calculate the auto-roll price.

SF-13
The Emergency Settlement Mechanism Can Be Negatively Impacted by Updating Price Feed Decimals

• Medium ⓘ Fixed

Update
Now, when the market is terminated in case of emergency, `LendingMarketController` stores the current price and number of decimals currently stored in `CurrencyController`. As a result, a later update of decimals number in `CurrencyController` will not affect the price calculations done when the user does an emergency settlement. Addressed in: `b469f1a999368796bbd56c176a4995a905f44e9c`.

The client provided the following explanation:

> *The updated points are as follows:*
> - *Add logic to cache currency decimals values instead of the CurrencyContoroller updates*
> - *Add a check to prevent the update of collateral currency in TokenVault at emergency termination*

File(s) affected: `CurrencyController.sol`

Description: In case of an emergency, a termination mechanism can be executed by the owner of the contract `LendingMarketController` to cease the operations of the protocol. Once executed, this operation cannot be reverted. For each active currency, the last aggregated price returned by the oracle is stored in the mapping `marketTerminationPrices`. Values in this mapping have the number of decimals stored in the mapping `CurrencyController.decimalCaches[]`. Then, users can force a settlement of all lending and borrowing positions via the function `executeEmergencySettlement()`.

Once the protocol is terminated, it is possible to update the values in the mapping `CurrencyController.decimalCaches[]`, which would affect the prices used during the settlement process and obtained via the functions `FundManagementLogic._convertToBaseCurrencyAtMarketTerminationPrice()` and `FundManagementLogic._convertFromBaseCurrencyAtMarketTerminationPrice()`.

Recommendation: Consider adding a check to prevent the update of a price feed once the protocol has been terminated via the emergency termination process.

SF-14
The Emergency Termination Mechanism Can Be Negatively Impacted by Incorrect Prices Sent by the Oracle

● Medium ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> *We have mock price feed contracts which return the fixed price. These contracts inherit the interface of the Chainlink Price Feed contract. Therefore, we can use our mock price feed instead of the Chainlink Price Feed contract when something happens. We use these mock contracts to solve this issue by the following steps:*
> 1. *Pause all order books.*
> 2. *Replace price feeds that return incorrect prices with mock price feeds.*
> 3. *Execute emergency termination.*

File(s) affected: `LendingMarketOperationLogic.sol`

Description: In case of an emergency, a termination mechanism can be executed by the owner of the contract `LendingMarketController` to cease the operations of the protocol. Once executed, this operation cannot be reverted.

For each active currency, the last aggregated price returned by the oracle is stored in the mapping `marketTerminationPrices`. Then, users can force a settlement of all their lending and borrowing positions via the function `executeEmergencySettlement()`.

However, if the emergency is caused by incorrect prices returned by the oracle, these prices will be recorded as reference prices and it would negatively impact the settlement phase as the coverage calculation would provide incorrect values.

Recommendation: Consider adapting the emergency termination mechanism to mitigate/eliminate the impact of incorrect prices provided by the oracle.

SF-15
Precision Loss in Itayose Process Makes `FutureValueVault` Unstable

● Medium ⓘ Fixed

Update
Now the total supply of the Future Vault is split into borrowing and lending supply, and the total amount of the pending orders to be cleaned is checked before moving the residual amount to the Genesis Vault. Addressed in: `aeb7398e437b47acf7ca3e6e3538cf9d01c3fa6f`.

File(s) affected: `FutureValueVault.sol`, `LendingMarketOperationLogic.sol`

Description: During the Itayose call, we calculate the `totalSupply` of Future Value (FV) using the Present Value (PV) (total amount filled in Itayose pre-orders) and `openingPrice` calculated from them.

Following the scenario below:

If the offset pre-order has the following amount with an opening unit price of `9220`, a precision loss will happen.

```
A: 100
B: 200
```

After the Itayose process, the total supply will be `325 = (300 * 10000) / 9220`, calculated from the opening price and the total filled amount from the pre-orders.

But each PV of order becomes:

```
A: 108 = (100 * 10000) / 9220
B: 216 = (200 * 10000) / 9220
```

This total sum of orders' future value is `324`, so the precision loss is `1` (`totalSupply` was set to `325`). `isAllRemoved` will never be fulfilled in `FutureValueVault.reset()`, leading to never calling `GenesisValueVault.updateGenesisValueWithResidualAmount()`.

```
isAllRemoved =
    Storage.slot().removedLendingSupply[maturity] == Storage.slot().totalSupply[maturity] &&
    Storage.slot().removedBorrowingSupply[maturity] == Storage.slot().totalSupply[maturity];
```

Recommendation: Avoid precision error, so the total supply calculated in the Itayose process will match the sum of all filled pre-orders with the Itayose opening price. This way, when orders are cleaned (after maturity), `isAllRemoved` will have the possibility to be `true`.

SF-16 Missing Input Validation

• Low ⓘ Mitigated

Update

Addressed in: `ecd9512a6864881ea246eed6514e21065bd262a6` and `25774e72a8e802b0e1e65d75aeb192141bbffec7`. The client provided the following explanation:

> *Added new validations for only parameters that have a risk of breaking protocol.*

File(s) affected: `BeaconProxyController.sol`, `LendingMarket.sol`, `MixinLiquidationConfiguration.sol`

Related Issue(s): SWC-123

Description: It is important to validate inputs, even if they only come from trusted addresses, to avoid human error:

1. `BeaconProxyController`: verify the value of `LendingMarket.MINIMUM_RELIABLE_AMOUNT_IN_BASE_CURRENCY` when the contract becomes the implementation contract of the beacon.
2. `LendingMarketOperationLogic.createOrderBook()`: verify that the duration between `_preOpeningDate` and `_openingDate` is short enough to avoid locking tokens by placing pre-orders.
3. Fixed `MixinLiquidationConfiguration._updateLiquidationConfiguration()`: verify that the value of `_liquidationThresholdRate` is higher than `Constants.PCT_DIGIT` to prevent the under-collateralization of the protocol.
4. Fixed `LendingMarketController.initializeLendingMarket()`: verify the correctness of the values used as parameters.

Recommendation: We recommend adding the relevant checks.

SF-17 Incomplete Checks when Updating a Price Feed in the `CurrencyController`

• Low ⓘ Fixed

Update

Addressed in: `b5c21d6865e06e223faef7150739b3d9d8bbf330` and `77549829f7406112d856451058f2e1d7fe1fdb8a`.

File(s) affected: `CurrencyController.sol`

Description: The system gets prices for a given pair of tokens (A, B) (price of one token A in token B) by using functions of the contract `CurrencyController` that fetch data from Oracle price feeds implementing the interface `AggregatorV3Interface`. When no price feed exists to directly provide the price of A in the base currency used by the protocol, a sequence of price feeds is stored in a struct `CurrencyControllerStorage.PriceFeed` that has two attributes:

- `AggregatorV3Interface[] instances` representing the sequence of price feeds to call consecutively;
- `uint256 heartbeat` representing the update frequency of an oracle; In addition, the mapping `decimalsCaches` stores the sum of the decimals used by the instances of the price feed.

A price feed can be updated via the function `updatePriceFeed()`. However, some aspects are not checked during this update:

1. A single `heartbeat` value is provided to apply for all items of `_priceFeeds`. However, these price feeds can have different heartbeat values, leading to possible reverts or stale data if the expected unique heartbeat does not match with the rest of the price feeds' heartbeats. Add individual heartbeat limits for each price feed.
2. No check makes sure that the value of `_decimals` matches the sum of the decimals of each item of `_priceFeeds`
3. No check makes sure that `heartbeat` is respected for the last price returned by the price feed
4. No check makes sure that the last item of `_priceFeeds` returns a price in base currency. A check could at least make sure that the number of decimals matches the number of decimals of the base currency
5. The `_updatePriceFeed()` function calls `latestRoundData()` on the provided price feeds and uses the returned price information without checking the `updatedAt` against a time limit. Perform the same checks on the `updatedAt` result value as in `_getAggregatedLastPrice()`.

Recommendation: Consider implementing the mentioned validations.

SF-18 Possible Reentrancy in Liquidation Logic

• Low ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client as intended by design. The client provided the following explanation:

> *This issue mentions `executeLiquidation()` and `executeForcedRepayment()` in the `LiquidationLogic`, but the caller function, such as `executeLiquidationCall()` has the `isNotLocked` modifier, which has almost the same logic as `nonReentrant`. This `isNotLocked` is enough for the reentrancy guard.*

File(s) affected: `LiquidationLogic.sol`

Description: The `executeLiquidation()` and `executeForcedRepayment()` functions both perform external function calls on user-supplied addresses. This allows for the possibility of reentrancy vulnerabilities.

Recommendation: If external calls are unnecessary, remove the logic. Otherwise, consider using the `nonReentrant` modifier on the functions.

SF-19 Added Orderbooks Do Not Operate with Their Initial Information

• Low ⓘ Acknowledged

Update

Marked as "Acknowledged" by the client. The client provided the following explanation:

> This is intended behavior.

File(s) affected: `OrderBookLogic.sol`

Description: The `OrderBookLogic` contract contains two functions that invoke the `initialize()` function on orderbook instances, `createOrderBook()`, which is called when the protocol owner attempts to create a new orderbook in an existing market, and `executeAutoRoll()`, which auto-rolls the current maturity to the next one.

When the owner calls `createOrderBook()`, they specify a maturity, an opening date, and a pre-opening date, and the initialization function sets those values to the passed maturity's entry in the `isReady()` mapping. After a while, when `executeAutoRoll()` is called by `LendingMarketOperationLogic.rotateOrderBooks()`, the following parameters are passed to it, among others:

- `_newMaturity`, which is the result of `calculateNextMaturity()`. This will be used as the key to the `isReady` mapping.
- `_openingDate`, which is the maturity date of the newly opened orderbook. This will be set as the next orderbook's opening date.

The function also initializes the pre-opening date of the orderbook to `_openingDate - OrderBookLib.PRE_ORDER_BASE_PERIOD`. As a result, the opening date and pre-opening date are recalculated for every orderbook, which renders the initial setup useless.

Recommendation: Except for the first one, orderbooks could be created with their maturity only since their opening and pre-opening dates will be calculated as a function of the maturities of their preceding orderbooks.

SF-20 Deletion of Tree Nodes Does Not Work as Intended

Low Acknowledged

Update

Marked as "Acknowledged" by the client as intended by design. The client provided the following explanation:

> *`OrderStatisticsTreeLib` uses `Node.head` to check orders existence such as `orderIdExists()` without removing `orders` from the gas costs aspect, so this logic is as per our specifications.*

File(s) affected: `OrderStatisticsTreeLib.sol`

Description: The `remove()` function performs a `delete self.nodes[cursor]` call towards its execution's end. The `nodes` mapping maps `uint256` values to `Node` struct values, which in turn contain a mapping `orders` that keeps track of orders under such values.

Deletion in this fashion will result in the `Node` struct's instance having its values reset to their defaults, except for the `orders` mapping which will keep its contents intact. This may result in unwanted situations where a node is removed and its orders remain.

Recommendation: Consider the explicit clearing of each order in the mapping or add a boolean such as `isRemoved` that is marked `true` when a node is removed and check on that value later on.

SF-21 Haircut and Pricefeed Data Not Erased when a Currency Is Removed

Low Acknowledged

Update

Marked as "Acknowledged" by the client as intended by design. The client provided the following explanation:

> *When a currency is delisted, `removeCurrency()` is called, but order books are active until the maturity date. While the market is active, these haircut and price feeds are needed. This is why `removeCurrency()` removes only the currency.*

File(s) affected: `CurrencyController.sol`

Description: The `removeCurrency()` function only removes the supplied `_ccy` value from the `currencies` mapping, however the `haircuts`, `priceFeeds`, and `decimalsCaches` entries for the given currency are not cleared.

Recommendation: Consider clearing all the information related to a currency when it gets removed.

SF-22 Upgradability

Low Acknowledged

Update

Marked as "Acknowledged" by the client as intended by design. The client provided the following explanation:

> *We will communicate to users about them.*

Description: Most of the contracts within the Secured Finance protocol are upgradeable, implementing the EIP-1967 storage slots in transparent and beacon proxy patterns. While this is not a vulnerability, users should be aware that the behavior of the protocol could drastically change if the contracts are upgraded.

Furthermore, new vulnerabilities not present during the audit could be introduced in upgraded versions of the contract, including the transfer of funds or transfer tokens that were allowed to the protocol.

This audit does not guarantee the behavior of future contracts that may be upgraded.

Recommendation: The fact that the contract can be upgraded and reasons for future upgrades should be communicated to users beforehand.

SF-23 Privileged Roles and Ownership

Low Acknowledged

Update

Marked as "Acknowledged" by the client. Secured Finance has shared the following regarding role management:

1. Deployer (General wallet): A general wallet used for deploying implementation contracts. It is not allowed to upgrade our protocol and execute any operations of our protocol. This wallet is created by the core dev team, and its private key is stored in the GitHub Actions. Also, this wallet is used to manage Web3 SaaS such as Gelato and The Graph.

2. Multisig Wallet (Safe Wallet): A multi-sig wallet created by Safe, it is used to execute proposals such as upgrading smart contracts, adding new currencies and order books, and adjusting protocol parameters. There are 3 admins and 1 operator as an approver, and 3 out of 4 approvals are required to execute a proposal.
3. Admin (Hardware Wallet): A hardware wallet used for approving Multisig wallet proposals. There are multiple wallets of this type, managed by C-level members. This wallet can create proposals for the Multisig wallet.
4. Operator (Defender Relayer): A managed wallet of Defender Relayer, it is used for actions such as creating proposals for the Multisig wallet, signing a proposal as one of the Multisig wallet approvers, and executing pausing of LendingMarket & TokenVault. The private key is created and managed in a secure vault of Defender.

Description: These are the privileged operations identified during the audit:

AddressResolver

- Owner can:
  - Renounce ownership
  - Transfer ownership (maybe to uncontrolled address)
  - Update the mapping between contract names and the address of the associated contract used by the protocol

BeaconProxyController

- Owner can:
  - Renounce ownership
  - Transfer ownership (maybe to uncontrolled address)
  - Update the implementation contract to an arbitrary contract for the contracts:
    - LendingMarket
    - FutureValueVault
- AcceptedContracts can:
  - Deploy FutureValueVault contracts.
  - Deploy LendingMarket contracts.

CurrencyController

- Owner can:
  - Renounce ownership
  - Transfer ownership (maybe to uncontrolled address)
  - Add a new currency (or override the parameters of an existing one) with the following parameters: number of decimals, haircut value, list of price feeds and heartbeat value.
  - Remove a currency
  - Update the haircut value of an existing currency
  - Update or remove the price feeds list of an existing currency

FutureValueVault

- Accepted Contracts can:
  - Increase the balance of a user
  - Decrease the balance of a user
  - Transfer tokens from a user
  - Reset all amount of a user if there is an amount in a past maturity
  - Set the initial total supply
  - Force the reset of the future value of a user

GenesisValueVault

- Accepted Contracts can:
  - Initialize currency settings
  - Trigger the update of the initial compound factor
  - Execute an AutoRoll
  - Update genesis value with future value
  - Update genesis value with residual amount
  - Transfer tokens from a user
  - Cleanup the balance of a user per maturity
  - Force the reset of the genesis value of a user

LendingMarket

- Accepted Contracts can:
  - Create an order book
  - Execute an autoRoll
  - Cancel an order
  - Execute an order
  - Execute a preorder

    - Unwind a position
    - Execute the itayose call
    - Update the order fee rates
    - Update the Circuit breaker limit range
    - Pause the contract
    - Unpause the contract

LendingMarketController

- Owner can:
    - Renounce ownership
    - Transfer ownership (maybe to uncontrolled address)
    - Update the order fee rates of a given currency
    - Update the Circuit breaker limit range of a given currency
    - Initialize a lending market
    - Create an order book
    - Trigger an emergency termination to stop the protocol
    - Update the minimum debt unit price
- Operator can:
    - Pause a lending market
    - Unpause a lending market
    - Renounce its role
- DefaultAdmin can:
    - Grant DefaultAdmin role to an address
    - Revoke DefaultAdmin role from an address
    - Renounce its role
    - Grant Operator role to an address
    - Revoke Operator role from an address
    - Update the admin role of a role, which means which roles can manage (grant/revoke) other roles.

ProxyController

- Owner can:
    - Renounce ownership
    - Transfer ownership (maybe to uncontrolled address)
    - Update in a batch the admin of a set of proxies deployed by this contract to an arbitrary new admin.
    - Update the implementation contract to an arbitrary contract and provide initialization data for the contracts:
        - AddressResolver
        - BeaconProxyController
        - TokenVault
        - CurrencyController
        - GenesisValueVault
        - LendingMarketController
        - ReserveFund
- For each contract:
    - a UpgradeableProxy is deployed if none was deployed until now for this given contract.
    - If the new contract contains:
        - a self.destruct instruction, existing proxy contracts could self-destruct.
        - a function to transfer all funds to an arbitrary address, existing proxy contracts could be drained.

ReserveFund

- Owner can:
    - Renounce ownership
    - Transfer ownership (maybe to uncontrolled address)
    - Execute a transaction (or a batch) via arbitrary call to arbitrary addresses with arbitrary payloads. It includes sending to an arbitrary address tokens owned by the `ReserveFund` contract.
    - Deposit tokens to the token vault
    - Withdraw tokens held by ReserveFund from the token vault

TokenVault

- Owner can:
    - Renounce ownership
    - Transfer ownership (maybe to uncontrolled address)
    - Register a currency to be accepted as a collateral
    - Unregister a currency to be accepted as a collateral
- Accepted Contracts can:
    - Trigger a deposit of collateral from any user that approved that contract to do so
    - Add a deposit amount for a user
    - Remove a deposit amount for a user
    - Force the reset of the deposit amount of a user

    - Transfer tokens from a user
- Operator can:
    - Pause the token vault
    - Unpause the token vault
    - Renounce its role
- DefaultAdmin can:
    - Grant DefaultAdmin role to an address
    - Revoke DefaultAdmin role from an address
    - Renounce its role
    - Grand Operator role to an address
    - Revoke Operator role from an address
    - Update the admin role of a role, which means which roles can manage (grant/revoke) other roles.

UpgradeabilityProxy

- Admin can:
    - Upgrade the contract to an arbitrary new contract
    - Change the admin to another address

UpgradeableBeacon

- Owner can:
    - Renounce ownership
    - Transfer ownership (maybe to uncontrolled address)
    - Upgrade the address of the beacon implementation.

UpgradeabilityBeaconProxy

- Admin can:
    - Upgrade the implementation contract of the beacon to an arbitrary new contract
    - Change the admin to another address

Recommendation:
- It is advisable to recommend implementing segregation of duties, dividing roles based on the urgency and nature of tasks. This helps prevent a single point of failure and enhances overall security.
- Key management should follow the latest best practices in security.
- This centralization of power needs to be made clear to the users, especially depending on the level of privilege the contract allows to the owner.

SF-24 Critical Roles Not Transferred Following a Two-Step Ownership Transfer Pattern

● Low ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> *We use a multisig wallet for all owner addresses. Therefore it will be a two-step pattern such as:*
> 1. *Create a proposal*
> 2. *Approve the proposal*
> 3. *Execute the proposal*

File(s) affected: `Ownable.sol`, `ProxyController.sol`

Description: The owner of the contracts inheriting from OpenZeppelin's `Ownable` can call `transferOwnership()` to transfer the ownership to a new address. If an uncontrollable address is accidentally provided as the new owner address then the contract will no longer have an active owner, and functions with the `onlyOwner` modifier can no longer be executed.

The following critical roles are not transferred following a two-step pattern:

- the role `proxyAdmin` in the function `ProxyController.changeProxyAdmins()`;
- the role `owner` in the contracts inheriting the contract `Ownable`;

If the role is transferred to an address owned by no one, the role can be considered to be lost.

Recommendation: Consider enforcing a role transfer in two steps.

SF-25 Ownership Can Be Renounced

● Low ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> *We use a multisig wallet for all owner addresses. Therefore it will be a two-step pattern such as:*
> 1. *Create a proposal*
> 2. *Approve the proposal*
> 3. *Execute the proposal*

File(s) affected: `AddressResolver.sol` , `BeaconProxyController.sol` , `CurrencyController.sol` , `ProxyController.sol` , `TokenVault.sol` , `MixinLendingMarketConfiguration.sol` , `MixinLiquidationConfiguration.sol` , `MixinWallet.sol` , `Ownable.sol` , `UpgradeableBeacon.sol`

Description: If the owner renounces their ownership, all ownable contracts will be left without an owner. Consequently, any function guarded by the `onlyOwner` modifier will no longer be able to be executed.

Recommendation: Confirm that this is the intended behavior. If not, override and disable the `renounceOwnership()` function in the affected contracts. For extra security, consider using a two-step process when transferring the ownership of the contract (e.g. `Ownable2Step` from OpenZeppelin).

SF-26
The Libraries `FullMath` , `RoundingInt256` , and `RoundingUint256` Can Revert Due to Overflows

• Low ⓘ Mitigated

Update
Mitigated by using OpenZeppelin's Math library instead of FullMath. Points 2 and 3 were not addressed. Addressed in: `3746b0fd60cb87f57bdc24e94fa5917bd2ba0068` .

File(s) affected: `FullMath.sol` , `RoundingInt256.sol` , `RoundingUint256.sol`

Description:

1. The function `FullMath.mulDiv()` overflows when `a * b > type(uint256).max` . Note that an implementation of the same algorithm by Open Zeppelin uses an `unchecked` block for the whole function: https://github.com/OpenZeppelin/openzeppelin-contracts/blob/769071d47366addc20e160bce53a6dd480d5f89d/contracts/utils/math/Math.sol#L124.
2. The function `RoundingInt256.div()` overflows when `a x 10 > type(int256).max` and underflows when `a x 10 < type(int256).min` .
3. The function `RoundingUint256.div()` overflows when `a x 10 > type(uint256).max` .

However, we assessed that it was unlikely to use these functions with such values for `a` and `b` .

Recommendation:
1. Consider documenting the limitation of this function or adding an `unchecked` block as it is done in the implementation of Open Zeppelin.
2. Consider adding a custom error to simplify the troubleshooting.
3. Consider adding a custom error to simplify the troubleshooting.

SF-27 Partial Imports of Addresses in `AddressResolver` Are Possible

• Low ⓘ Mitigated

Update
The first recommended item has been fixed, with a cache array that removes all the previously updated names during the last update, before adding any new address and it will save the new cache array of names. However, the team did not add a mechanism to prevent the use of duplicates. Addressed in: `16ddaa32306ec5d0fe53965b0ebfc99b56c8a639` .

The client provided the following explanation:

> *Added the logic to prevent partial Imports.*

File(s) affected: `AddressResolver.sol`

Description: The function `importAddresses()` can be called by the owner of the contract to update the mapping of a list of addresses with new names. Such action will also result in the update of the storage list `addressCaches` . However, this function can be called several times and maintain previous mapping associations valid.

For instance:

1. Owner calls the function with `_names = ["A", "B"]` and `_addresses = [0x01,0x02]` . The mapping `addresses` is updated accordingly and `addressCaches` takes the value of `_addresses` .

2. Owner calls again the function with `_names = ["A"]` and `_addresses = [0x03]`. The mapping `addresses` is updated accordingly and `addressCaches` takes the value of `_addresses`.
3. After the operation:
   - `addressCaches` is `[0x03]`;
   - `addresses["A"] = 0x03` ';
   - `addresses["B"] = 0x02` '; This could lead to storage inconsistencies between what is stored in the mapping `addresses` and the list `addressCaches` and ultimately to broken assumptions about the system if the function `importAddresses()` is called with inconsistent parameters.

Also, duplicate items are possible in both lists (`_names` and `_addresses`). Having duplicates could result in unexpected mapping associations and duplicates in the list `addressCaches`.

Recommendation: Consider updating the behavior of the function `importAddresses()` to:
1. Verify that `addressCaches` always reflects all non-zero addresses currently associated with a name in the system.
2. Revert when duplicate items are found in at least one of the lists `_names` and `_addresses`.

SF-28 Genesis Value Balance Incorrectly Updated

• Low ⓘ Fixed

Update
Fixed by setting the balance of the reserve fund after calling `_updateTotalSupplies()`. Addressed in: `7df1b02e127463a21b479985ddea0b1efa1bbd08`.

File(s) affected: `GenesisValueVault.sol`

Description: In `GenesisValueVault._updateBalance()`, `_updateTotalSupply()` is called before updating the balance of the `ReserveFund` but it depends on the user balance passed as an argument, so there could be accounting issues.

Recommendation: Call `_updateTotalSupply()` before updating the balance of the `ReserveFund` contract, not after.

SF-29 Users Can Cancel Orders Anytime

• Low ⓘ Fixed

Update
A new modifier `ifNotItayosePeriod` was created to protect `LendingMarket.cancelOrder()`. Addressed in: `67e72eeed5023a411dcd5b203c54d5695ac9cc61`.

File(s) affected: `OrderActionLogic.sol`, `LendingMarket.sol`, `LendingMarketController.sol`

Description: Orderbooks are expected to be frozen after pre-order period, during the Itayose period:

> *"One hour before the launch, the orderbook will be frozen, and users will not be able to take any action on the orderbook. This includes placing, amending or canceling orders."*

However, this is not enforced anywhere in the code, and `LendingMarketController.cancelOrder()` can be called anytime as long as the maturity is valid and the caller is the maker of the existing order.

Recommendation: Verify that the market is not in the Itayose period before canceling pre-orders.

SF-30 `TokenVault` Is a Greedy Contract

• Low ⓘ Fixed

Update
Validation added in `_deposit()`. The transaction will revert if sending native tokens while indicating a non-native token address. Addressed in: `30f4c037da17652a4597814694c19b59328e2c70`.

File(s) affected: `TokenVault.sol`

Description: Calling the function `deposit()` for a currency `_ccy` that is not the contract representing the native token, with a non-zero `_amount` as well as a non-zero `msg.value` will result in a deposit of `_amount _ccy`, but the amount of native tokens `msg.value` will not be accounted by the system. This is caused by the first `if` condition in the function `_deposit()`.

Recommendation: Consider updating the condition in the function `_deposit()` to prevent the situation described above.

SF-31 Protocol Does Not Support Non-Standard Tokens

• Low ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> *We support only standard ERC20 tokens at this point.*

Description: Some ERC20 tokens can change the balance of accounts outside of transfer methods. These tokens are commonly referred to as rebasing tokens. Since the protocol only ensures that an expected amount of tokens is received after a transfer, tokens that change the balance through other methods will disrupt internal accounting.

Here are some tokens that can also affect the protocol operation:

- Tokens that can be paused
- Tokens with allowlist/blocklist
- Tokens that can be upgraded

Recommendation:

1. Markets should only be created using tokens that cannot modify the balance of a contract outside of a typical transfer call.
2. Currencies registered in the system should not be inflationary or deflationary or be subject to a transfer fee.
3. Before adding a new currency (ERC-20) to the system, extensive testing is highly recommended.

SF-32 'Dead' Code

● Informational ⓘ Fixed

Update
Recommendations were addressed in: `e8ffcfa8df0b28f6a8fcf08769e3922a8da6160b`.

File(s) affected: `LendingMarketController.sol`, `OrderBookLib.sol`, `FundManagementLogic.sol`, `OrderActionLogic.sol`, `OrderReaderLogic.sol`

Related Issue(s): SWC-131, SWC-135

Description: "Dead" code refers to code that is never executed and hence makes no impact on the final result of running a program. Dead code raises a concern, since either the code is unnecessary or the necessary code's results were ignored.

Here are some snippets of dead code found in the codebase:

1. `TokenVault` contract is an accepted contract in `LendingMarketController`. However, `LendingMarketController` does not implement any function restricted to `acceptedContracts()`, so `acceptedContracts()` can be refactored to return an empty array.
2. `OrderBookLib.calculateItayoseResult()`: if there are no matching orders, the function returns zero for all the variables (`L530`). `openingUnitPrice` is calculated before, but the value is not used, wasting gas in that calculation.
3. `FundManagementLogic.calculateTotalFundsInBaseCurrency.amounts[]` can be reduced to seven elements.
4. `OrderActionLogic._fillOrders()` updates `filledOrder.amount` with `_amount - vars.remainingAmount`. However, this will always return `filledOrder.amount`, as `vars.remainingAmount` was already subtracted.
5. `OrderReaderLogic.getTotalAmountFromBorrowOrders()` commented part in the second for loop.

More recommendations are discussed in the "Adherence to Best Practices" section.

Recommendation: Remove or refactor the abovementioned code statements.

SF-33 Ignoring Values Returned by Functions Is Not Always Explicitly Justified

● Informational ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. It will not be addressed now.

File(s) affected: `TokenVault.sol`

Description: The following functions are called but the calling functions ignore the values returned without explicit justification:

1. `LendingMarketController.cleanUpFunds()` in `TokenVault._withdraw()`;
2. `GenesisValueVault.executeForcedReset()` in `FundManagementLogic._resetFundsPerCurrency()`;
3. `LiquidationLogic._transferCollateral()` in `LiquidationLogic.executeLiquidation()`;
4. `LiquidationLogic._transferCollateral()` in `LiquidationLogic.executeForcedRepayment()`;
5. `LiquidationLogic._transferPositionsPerMaturity()` in `LiquidationLogic.executeLiquidation()`;
6. `TokenVault.transferFrom()` in `LiquidationLogic.executeForcedRepayment()`;

If not intended, it can result in unexpected behavior of the system. If intended, it makes it harder for an external observer to understand why returned values are ignored.

Recommendation: For each of the instances above, consider using the returned values or inserting an inline comment to explicitly explain why returned values are not used.

SF-34 Long Contracts Names

● Informational ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. It will not be addressed now.

File(s) affected: `Contracts.sol`

Description: In `Contracts.sol`, as the names of the contracts are used as `bytes32` keys in a mapping, using a string with a representation exceeding `bytes32` could break the assumption that the same key only refers to one contract name.

Recommendation: No issue was found in the current code, but it could happen in the future if a long contract name is used.

SF-35 Unlocked Pragma

● Informational ⓘ Fixed

Update
The Solidity version was fixed at `0.8.19`. Addressed in: `7b957ba46ffb0b746ed4064ccbf7489f2f543760`.

Related Issue(s): SWC-103

Description: Every Solidity file specifies in the header a version number of the format `pragma solidity (^)0.8.*`. The caret (`^`) before the version number implies an unlocked pragma, meaning that the compiler will use the specified version and above, hence the term "unlocked".

The pragma versions used throughout the codebase should not be unlocked and set to the same version to ensure that they are all compiled by the same compiler version.

Recommendation: For consistency and to prevent unexpected behavior in the future, we recommend removing the caret to lock the file onto a specific Solidity version (e.g. `pragma solidity 0.8.19`).

SF-36 Wrong Formula In Function Comment

● Informational ⓘ Fixed

Update
Addressed in: `6ce3df0c95363a991389c4d6e53069f28d0b9a32`.

File(s) affected: `DepositManagementLogic.sol`

Description: In `DepositManagementLogic.getWithdrawableCollateral()` the formula defined in the comment does not match the formula implemented in code to calculate `maxWithdraw`.

Recommendation: Align the comment and the implemented code to match what is intended.

SF-37 Undocumented Constants

● Informational ⓘ Fixed

Update
Addressed in: `a312cd741b823f10274899081cd05dd104bb9f86`.

File(s) affected: `CurrencyController.sol`, `LendingMarketOperationLogic.sol`

Description: Although the codebase makes use of hardcoded values declared in `Constants.sol`, there are some undocumented values in the codebase:

- `CurrencyController._updateHaircut()`: `10000`
- `LendingMarketOperationLogic.initializeLendingMarket()`: `36`

Recommendation: Define the mentioned hardcoded values as constants and document them.

SF-38 Deprecated Function

● Informational ⓘ Fixed

Update

`_setupRole()` was replaced with `_grantRole()`. Addressed in: `fda2f7389ce13248680fcf538bbf5ad7e1975f70`.

The client provided the following explanation:

> *Fixed it together with SF-10*

File(s) affected: `MixinAccessControl.sol`

Description: `MixinAccessControl._setupInitialRoles()` calls `_setupRole()` and `_grantRole()`.

Both functions have the same effect, as `_setupRole()` just calls `_grantRole()`. `_setupRole()` is being deprecated in `v5.0.0` of OpenZeppelin contracts, and its use is not recommended. See `_setupRole()` comment:

```
NOTE: This function is deprecated in favor of {_grantRole}.
```

Recommendation: Replace `_setupRole()` with `_grantRole()`.

SF-39 Wrong Constant Used

● Informational ⓘ Fixed

Update

Addressed in: `d22b8d93523f2c2bdf2aa5f00286f63d460f63b9`.

File(s) affected: `DepositManagementLogic.sol`

Description: `DepositManagementLogic.getWithdrawableCollateral()` uses `Constants.PRICE_DIGIT` to do related computations. However, as it is operating with rates, `Constants.PRICE_DIGIT` should not be used.

Recommendation: Even if both constants have the same value (`10000`), to keep consistency, use `Constants.PCT_DIGIT` instead of `Constants.PRICE_DIGIT`.

SF-40 Mismatch Between Code and Documentation when Inserting Nodes Into the Order Book

● Informational ⓘ Fixed

Update

Addressed in: `a118279611451227927c61433ceb7687fa312bd4`.

File(s) affected: `OrderStatisticsTreeLib.sol`

Description: In `OrderStatisticsTreeLib.insert()` there are 4 ways to pass the while loop:

- case 1: `tree.root == EMPTY`, which means that the tree is empty;
- case 2: we can reach a node of value “value” from `root`, and we exit the function with a return statement;
- case 3: we cannot reach a node of value “value” from `root`. Here, we can have two situations:
  - 3.a: `value < cursor`;
  - 3.b: `value > cursor`; Then, we get from storage the node `nValue`, which is a node of value “value”. `nValue` can be:
- case 4: empty;
- case 5: non-empty, meaning it has been dropped previously; We should update `nValue` in case 4 and case 5, but doing it in case 4 would be redundant because it already has default values. It is the purpose of the if statement, to only do these 4 operations if we are in case 5, to save gas.

However, the line `(self.nodes[cursor].left != value && self.nodes[cursor].right != value)` is a tautology and does not only capture case 4 but rather captures cases 4 and 5, because subcases `self.nodes[cursor].left == value` and `self.nodes[cursor].right == value` would have been captured in the while loop, which is case 2.

Recommendation: Consider replacing the if statement with a statement that only captures the case where `nValue` is non-empty (case 5).

SF-41 Events Emitted Even if No Update Is Performed

● Informational ⓘ Mitigated

Update

Mitigated as only point 2 was fixed by implementing `_executeTransaction()` and emitting the event inside it. Addressed in: `ffef3207b79ef67b59853f7b0a4559bb74f15832`.

The client provided the following explanation:

> *Fixed only no.2.*

File(s) affected: `MixinAddressResolver.sol`, `MixinWallet.sol`

Description:
1. The function `MixinAddressResolver.buildCache()` can be called by anyone to update the addresses associated with the required contract names used by the contract. An event `CacheUpdated` is emitted for each required contract, whether it is updated or not. This could result in more events emitted than required.
2. The function `MixinWallet.executeTransactions()` can be called by the owner of the contract. It results in the emission of an event `TransactionsExecuted` even if no transaction has been executed.

Recommendation:
1. Consider adding a check to only emit events when the execution results in a mapping update.
2. Consider reverting the transaction if the lists are empty.

SF-42
Transactions Not Reverting Even if the State Is Not Modified

● Informational ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. It will not be addressed now.

File(s) affected: `CurrencyController.sol`, `ProxyController.sol`, `ReserveFund.sol`

Description: The following functions let a transaction execute normally even if it does not result in a state modification, resulting in gas spent when it was not needed:
1. In the function `ProxyController.changeProxyAdmins()` if the list `destinations` is empty
2. In the functions `ReserveFund.pause()` and `ReserveFund.unpause()`
3. In the function `CurrencyController.removeCurrency()` if the removed currency does not exist

Recommendation: Consider making sure that a transaction involving one of the functions listed above reverts if it does not result in a state modification.

SF-43
Possible DoS via High Gas Usage During Liquidations

● Undetermined ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> *In `TokenVault.getLiquidationAmount()`, `getTotalCollateralAmount()` is the gas-intensive function. To check the actual gas costs of that function and `FundManagementLogic.cleanUpFunds()`, we have performance tests. As a result, it was confirmed those gas costs were within a realistic size because of the limitation of active order numbers. In addition, we have a plan to add a limitation of used currency numbers in the future.*

File(s) affected: `LiquidationLogic.sol`

Description: The `executeLiquidation()` function performs a heavy logic to liquidate a position. The function calls other lengthy and gas-intensive functions such as `FundManagementLogic.cleanUpFunds()` and `TokenVault.getLiquidationAmount()`, which may result in failed liquidations due to insufficient gas usage.

Recommendation:
1. Perform a gas analysis of the functionality.
2. Try to break the functionality down into smaller actions that may be carried out by a liquidator, possibly in different transactions. Another option is to attempt to cache function execution results and use them in the liquidation logic if they are not outdated.

SF-44
The Purpose of Being Able to Remove a Price Feed Is Unclear

● Undetermined ⓘ Fixed

Update
The logic for removing price feeds was deleted. Addressed in: `136e5deee27bbb8c68b1b8de1b92c1be964e0ccc`.

File(s) affected: `CurrencyController.sol`

Description: The owner of the contract `CurrencyController` can remove the price feed of a given supported currency via the function `removePriceFeed()`. If the currency is still supported by the system, getting prices for that currency will not revert but will rather return the value of `1` wei. As a result, this use case could negatively impact the computation of position coverage.

Recommendation: Consider assessing if this functionality is needed in the protocol. If not, consider removing it.

SF-45
Risks Related to Errors Involving the Price Feeds of a Subset of Supported Currencies

● Undetermined ⓘ Acknowledged

Update
Marked as "Acknowledged" by the client. The client provided the following explanation:

> *Each order book is totally separated. Therefore, even if some currency is not working because of the error StalePriceFeed, other currency doesn't get affected by that. When users have positions in multiple currencies, they can execute no action, but this is an expected behavior to protect the protocol's and user's funds.*

File(s) affected: `CurrencyController.sol`

Description: The price feeds associated with a currency `C` can temporarily not update their price. In this case, this will be detected via the function `CurrencyController._getAggregatedLastPrice()` which will raise an error `StalePriceFeed`. This would make all flows involving getting a price for `C` revert. However, any other flow will still work, including the ones involving getting prices of other currencies.

Recommendation: Consider assessing if it is acceptable to keep the protocol live if the price feeds of a subset of supported currencies start reverting.

Definitions

- **High severity** – High-severity issues usually put a large number of users' sensitive information at risk, or are reasonably likely to lead to catastrophic impact for client's reputation or serious financial implications for client and users.
- **Medium severity** – Medium-severity issues tend to put a subset of users' sensitive information at risk, would be detrimental for the client's reputation if exploited, or are reasonably likely to lead to moderate financial impact.
- **Low severity** – The risk is relatively small and could not be exploited on a recurring basis, or is a risk that the client has indicated is low impact in view of the client's business circumstances.
- **Informational** – The issue does not post an immediate risk, but is relevant to security best practices or Defence in Depth.
- **Undetermined** – The impact of the issue is uncertain.
- **Fixed** – Adjusted program implementation, requirements or constraints to eliminate the risk.
- **Mitigated** – Implemented actions to minimize the impact or likelihood of the risk.
- **Acknowledged** – The issue remains in the code but is a result of an intentional business or design decision. As such, it is supposed to be addressed outside the programmatic means, such as: 1) comments, documentation, README, FAQ; 2) business processes; 3) analyses showing that the issue shall have no negative consequences in practice (e.g., gas analysis, deployment settings).

Code Documentation

1. Consider improving the readability of the code documentation by systematically using the NatSpec format. The purpose of the function as well as the expected format of parameters and what it returns should clearly be described.
2. In `GenesisValueVault.sol`, consider renaming the name of the parameter `_currentMaturity` to make sure we consider this value as a duration, instead of an absolute timestamp.
3. `TokenVault.getWithdrawableCollateral()` comment states: `@return Maximum amount of ETH that can be withdrawn`. This is incorrect, as this function returns a `uint256` referring to the base currency of the protocol (`USDC` at this point).

Adherence to Best Practices

1. Multiple functions operate over array lengths without caching them to local variables. It is recommended to cache this value in a local `uint256` variable before the loop instead of getting its length at each loop iteration, to save gas.

In `LendingMarketStorage.sol`:

1. The line `using OrderStatisticsTreeLib for OrderStatisticsTreeLib.Tree` can be removed because no `OrderStatisticsTreeLib.Tree` object is used.

In `OrderBookLib.sol`:

1. The first line of the function `_removeOrderIdFromOrders()` can revert due to an underflow if the list `orders` is empty.

In `DepositManagementLogic.sol`:

1. The body of the function `getUsedCurrencies()` could be replaced by `return Storage.slot().usedCurrencies[_user].values();`.
2. The comment in the function `getWithdrawableCollateral()` does not match the behavior of the code.

In `OrderReaderLogic.sol`:

1. The line `maturity = orderBook.userCurrentMaturities[_user];` is found twice in this function.

In `OrderActionLogic.sol`:

1. In the function `unwindPosition()`, in the else block, `conditions.orderExists` is always `true` due to the line above `if (!conditions.orderExists) revert EmptyOrderBook();`.

In `TransferHelper.sol`:

1. The error message in the function `safeTransfer()` is incorrect.

In `FundManagementLogic.sol`:

1. In the function `calculateTotalFundsInBaseCurrency()`, the array `amount` is initialized with a size of `8` but only `7` items are used.

In `FutureValueVault.sol`:

1. In the function `executeForcedReset()`, the line `Storage.slot().balances[_orderBookId][_user] -= removedAmount;` could be replaced with `Storage.slot().balances[_orderBookId][_user] = 0;`.

In `OrderStatisticsTreeLib.sol`:

1. In function `_removeOrders()`, the returned value amount is defined but not used.
2. In functions `dropLeft()` and `dropRight()`, in the last condition `droppedAmount >= totalNodeAmount`, `droppedAmount` cannot be strictly higher than `totalNodeAmount` due to the line above `uint256 totalNodeAmount = droppedAmount + exceededAmount;`.
3. In function `insertOrder()` will revert with the message `Insufficient value` if `value` is greater than `10000` (`Constants.PRICE_DIGIT`). Modify the revert message to `Value too high` or similar.

In the contracts `[...]Storage.sol`:

1. When hardcoding the storage slot of a `struct`, the standard ERC-1967 recommends adding a `-1` offset so the preimage of the hash cannot be known, further reducing the chances of a possible attack.

Adherence to Specification

1. Liquidators are not whitelisted, contrary to what is written in the doc: https://docs.secured.finance/technical-overview/liquidators.
2. The public documentation does not cover the haircut system for non-collateral currencies. This is a core part of collateral calculation and users should be informed about it.

Appendix

File Signatures

The following are the SHA-256 hashes of the reviewed files. A file with a different SHA-256 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different SHA-256 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review.

Contracts

- `31b...ce1 ./protocol/BeaconProxyController.sol`

- `99e...af1 ./protocol/FutureValueVault.sol`
- `ff5...583 ./protocol/GenesisValueVault.sol`
- `936...8bf ./protocol/CurrencyController.sol`
- `40e...399 ./protocol/TokenVault.sol`
- `306...b28 ./protocol/AddressResolver.sol`
- `bb6...e39 ./protocol/ReserveFund.sol`
- `aa7...7e7 ./protocol/LendingMarketController.sol`
- `429...1fb ./protocol/ProxyController.sol`
- `4c6...abd ./protocol/LendingMarket.sol`
- `5d7...cb4 ./protocol/interfaces/INativeToken.sol`
- `a32...14c ./protocol/interfaces/ILiquidationReceiver.sol`
- `851...659 ./protocol/interfaces/IAddressResolver.sol`
- `19f...712 ./protocol/interfaces/IBeaconProxyController.sol`
- `c20...4a3 ./protocol/interfaces/IProxyController.sol`
- `536...0a6 ./protocol/interfaces/ILendingMarketController.sol`
- `081...6c5 ./protocol/interfaces/ILendingMarket.sol`
- `53d...378 ./protocol/interfaces/IFutureValueVault.sol`
- `f38...7e8 ./protocol/interfaces/ITokenVault.sol`
- `f07...654 ./protocol/interfaces/IGenesisValueVault.sol`
- `354...b19 ./protocol/interfaces/ICurrencyController.sol`
- `02c...604 ./protocol/interfaces/IReserveFund.sol`
- `194...e38 ./protocol/utils/Proxyable.sol`
- `918...f18 ./protocol/utils/Pausable.sol`
- `60d...e82 ./protocol/utils/Ownable.sol`
- `ef5...e21 ./protocol/utils/UpgradeableBeacon.sol`
- `07e...eea ./protocol/utils/UpgradeabilityProxy.sol`
- `68c...b17 ./protocol/utils/LockAndMsgSender.sol`
- `b40...fb2 ./protocol/utils/UpgradeabilityBeaconProxy.sol`
- `8b0...efb ./protocol/utils/AccessControl.sol`
- `677...8bb ./protocol/libraries/QuickSort.sol`
- `0e8...bfe ./protocol/libraries/OrderBookLib.sol`
- `0df...b1b ./protocol/libraries/AddressResolverLib.sol`
- `ee3...a77 ./protocol/libraries/Constants.sol`
- `4a9...57c ./protocol/libraries/TransferHelper.sol`
- `3da...4f2 ./protocol/libraries/Contracts.sol`
- `36c...ce2 ./protocol/libraries/OrderStatisticsTreeLib.sol`
- `e02...761 ./protocol/libraries/BokkyPooBahsDateTimeLibrary.sol`
- `927...844 ./protocol/libraries/logics/OrderActionLogic.sol`
- `c69...b56 ./protocol/libraries/logics/LendingMarketUserLogic.sol`
- `89a...eff ./protocol/libraries/logics/DepositManagementLogic.sol`
- `7dd...f3e ./protocol/libraries/logics/LendingMarketOperationLogic.sol`
- `7a5...842 ./protocol/libraries/logics/FundManagementLogic.sol`
- `749...1e2 ./protocol/libraries/logics/OrderReaderLogic.sol`
- `dc6...09d ./protocol/libraries/logics/LiquidationLogic.sol`
- `cb5...157 ./protocol/libraries/logics/OrderBookLogic.sol`
- `f5e...062 ./protocol/libraries/math/RoundingUint256.sol`
- `9c6...059 ./protocol/libraries/math/RoundingInt256.sol`
- `5b3...45a ./protocol/libraries/math/FullMath.sol`
- `c31...006 ./protocol/mixins/MixinAccessControl.sol`
- `d1c...ea2 ./protocol/mixins/MixinAddressResolver.sol`
- `793...e0d ./protocol/mixins/MixinWallet.sol`
- `3ac...ae3 ./protocol/mixins/MixinLendingMarketConfiguration.sol`
- `258...c7b ./protocol/mixins/MixinLiquidationConfiguration.sol`
- `a15...937 ./protocol/types/ProtocolTypes.sol`
- `322...eb1 ./protocol/storages/CurrencyControllerStorage.sol`

- `99f...622 ./protocol/storages/ReserveFundStorage.sol`
- `6e4...796 ./protocol/storages/BeaconProxyControllerStorage.sol`
- `b26...d29 ./protocol/storages/AddressResolverStorage.sol`
- `24a...387 ./protocol/storages/GenesisValueVaultStorage.sol`
- `006...8ff ./protocol/storages/FutureValueVaultStorage.sol`
- `bbf...7e2 ./protocol/storages/TokenVaultStorage.sol`
- `a3e...1e9 ./protocol/storages/LendingMarketControllerStorage.sol`
- `13a...b52 ./protocol/storages/LendingMarketStorage.sol`
- `9b1...3a7 ./protocol/storages/utils/AccessControlStorage.sol`
- `d65...d76 ./protocol/storages/utils/OwnableStorage.sol`
- `be9...54d ./protocol/storages/utils/PausableStorage.sol`
- `d15...031 ./protocol/storages/libraries/TransferHelperStorage.sol`
- `5d1...9bc ./protocol/storages/mixins/MixinAddressResolverStorage.sol`

Tests

- `d26...976 ./test/performance/auto-rolls.test.ts`
- `ba3...c4d ./test/performance/order-book.test.ts`
- `08b...150 ./test/common/currencies.ts`
- `016...a65 ./test/common/format.ts`
- `7c1...f97 ./test/common/orders.ts`
- `383...571 ./test/common/constants.ts`
- `4bf...339 ./test/common/signers.ts`
- `3b2...19c ./test/common/deployment.ts`
- `093...803 ./test/integration/calculations.test.ts`
- `754...9ad ./test/integration/auto-rolls.test.ts`
- `96c...c71 ./test/integration/order-book.test.ts`
- `2c7...2aa ./test/integration/emergency-terminations.test.ts`
- `159...6ce ./test/integration/liquidations.test.ts`
- `1ef...55a ./test/integration/deposit.test.ts`
- `b64...e23 ./test/integration/itayose.test.ts`
- `cb3...7f1 ./test/unit/currency-contoroller.test.ts`
- `a23...a01 ./test/unit/reserve-fund.test.ts`
- `ed0...829 ./test/unit/token-vault.test.ts`
- `8f6...2e7 ./test/unit/proxy-controller.test.ts`
- `a1f...f66 ./test/unit/time-library.test.ts`
- `5c6...66e ./test/unit/order-statistics-tree/insert-delete.test.ts`
- `33f...76d ./test/unit/order-statistics-tree/drop.test.ts`
- `012...558 ./test/unit/order-statistics-tree/data/lending-orders.ts`
- `66a...08d ./test/unit/order-statistics-tree/data/borrowing-orders.ts`
- `dbe...4ce ./test/unit/order-statistics-tree/data/steps.ts`
- `d0d...d9a ./test/unit/libraries/lending-market-operation-logic.test.ts`
- `f6b...7b9 ./test/unit/lending-market/calculations.test.ts`
- `744...a89 ./test/unit/lending-market/orders.test.ts`
- `fe1...4c6 ./test/unit/lending-market/utils.ts`
- `2b9...86d ./test/unit/lending-market/circuit-breakers.test.ts`
- `508...96b ./test/unit/lending-market/initialization.test.ts`
- `40e...ad4 ./test/unit/lending-market/itayose.test.ts`
- `7a9...ec4 ./test/unit/lending-market-controller/calculations.test.ts`
- `4e5...644 ./test/unit/lending-market-controller/terminations.test.ts`
- `b79...2d5 ./test/unit/lending-market-controller/orders.test.ts`
- `e3e...706 ./test/unit/lending-market-controller/rotations.test.ts`
- `26c...957 ./test/unit/lending-market-controller/utils.ts`
- `b22...b33 ./test/unit/lending-market-controller/liquidations.test.ts`
- `c2c...6c1 ./test/unit/lending-market-controller/itayose.test.ts`
- `798...78b ./test/unit/lending-market-controller/operations.test.ts`

Toolset

The notes below outline the setup and steps performed in the process of this audit.

Setup

Tool Setup:
- Slither v0.9.3

Steps taken to run the tools:
1. Install the Slither tool: `pip3 install slither-analyzer`
2. Run Slither from the project directory: `slither .`

Automated Analysis

Slither

Slither was used to get a static analysis of the repository. All the issues and recommendations are discussed in this report or classified as false positives.

```
INFO:Slither analyzed (353 contracts with 85 detectors), 3592 result(s) found
```

Test Suite Results

All tests passed. We recommend adding tests covering auto-rolls in markets opened by the Itayose method.

```
  Integration Test: Auto-rolls
    Execute auto-roll with orders on the single market
      ✓ Fill an order
      ✓ Execute auto-roll (1st time)
      ✓ Execute auto-roll (2nd time)
    Execute auto-roll with orders on the multiple markets
      ✓ Fill an order on the closest maturity market
      ✓ Fill an order on the second closest maturity market
      ✓ Check total PVs
      ✓ Execute auto-roll
      ✓ Clean orders
    Execute auto-rolls more times than the number of markets using the past auto-roll price
      ✓ Fill an order
      ✓ Execute auto-roll (1st time)
      ✓ Execute auto-roll (2nd time)
      ✓ Execute auto-roll (3rd time)
      ✓ Execute auto-roll (4th time)
      ✓ Execute auto-roll (5th time)
      ✓ Execute auto-roll (6th time)
      ✓ Execute auto-roll (7th time)
      ✓ Execute auto-roll (8th time)
      ✓ Execute auto-roll (9th time)
      ✓ Execute auto-roll (10th time)
    Execute auto-roll with many orders, Check the FV and GV
      ✓ Fill an order
      ✓ Check future values
      ✓ Execute auto-roll, Check genesis values
    Execute auto-roll well past maturity
      ✓ Fill an order
      ✓ Advance time
      ✓ Fail to create an order due to market closure
      ✓ Execute auto-roll

  Integration Test: Calculations
    Order Estimations
      Estimate a borrowing order result to be filled
        ✓ Deposit ETH
        ✓ Place a lending order on the ETH market
        ✓ Estimate a borrowing order result
      Estimate a lending order result to be filled
        ✓ Place a borrowing order on the ETH market
        ✓ Deposit ETH
        ✓ Estimate a lending order result
      Estimate a borrowing order result to be placed
        ✓ Deposit ETH
        ✓ Estimate a borrowing order result
      Estimate a lending order result to be placed
        ✓ Deposit ETH
        ✓ Estimate a lending order result
    Borrowable Amount Calculations
      Calculate the borrowable amount with deposit
        ✓ Deposit ETH
        ✓ Calculate the borrowable amount in ETH
        ✓ Calculate the borrowable amount in WFIL
      Calculate the borrowable amount with borrowing position (Haircut: 0)
        ✓ Deposit ETH
        ✓ Fill an order to get a borrowing position
        ✓ Calculate the borrowable amount in ETH(1)
        ✓ Calculate the borrowable amount in WFIL(1)
        ✓ Fill an order to partially use the borrowing position
        ✓ Calculate the borrowable amount in ETH(2)
        ✓ Calculate the borrowable amount in WFIL(2)
        ✓ Fill an order to use the whole borrowing position
        ✓ Calculate the borrowable amount in ETH(3)
        ✓ Calculate the borrowable amount in WFIL(3)
      Calculate the borrowable amount with borrowing position (Haircut: 5000)
        ✓ Deposit ETH
        ✓ Fill an order to get a borrowing position
        ✓ Calculate the borrowable amount in ETH(1)
        ✓ Calculate the borrowable amount in WFIL(1)
        ✓ Fill an order to partially use the borrowing position
        ✓ Calculate the borrowable amount in ETH(2)
        ✓ Calculate the borrowable amount in WFIL(2)
        ✓ Fill an order to use the whole borrowing position
        ✓ Calculate the borrowable amount in ETH(3)
        ✓ Calculate the borrowable amount in WFIL(3)
      Calculate the borrowable amount with borrowing position (Haircut: 9600)
        ✓ Deposit ETH
        ✓ Fill an order to get a borrowing position
        ✓ Calculate the borrowable amount in ETH(1)
        ✓ Calculate the borrowable amount in WFIL(1)
        ✓ Fill an order to partially use the borrowing position
        ✓ Calculate the borrowable amount in ETH(2)
        ✓ Calculate the borrowable amount in WFIL(2)
        ✓ Fill an order to use the whole borrowing position
        ✓ Calculate the borrowable amount in ETH(3)
        ✓ Calculate the borrowable amount in WFIL(3)
      Calculate the borrowable amount with deposit & borrowing position
        ✓ Deposit ETH
        ✓ Fill an order to get a borrowing position
        ✓ Calculate the borrowable amount in ETH
        ✓ Calculate the borrowable amount in WFIL

  Integration Test: Deposit
    Deposit ETH, Withdraw all collateral
      ✓ Deposit ETH
      ✓ Withdraw all collateral
    Deposit WBTC, Withdraw all collateral
      ✓ Deposit WBTC
      ✓ Withdraw all collateral
    Deposit ETH twice, Withdraw all collateral
      ✓ Deposit ETH
      ✓ Withdraw partially
      ✓ Withdraw with over amount input
    Deposit multiple currency, Withdraw all collateral
      ✓ Deposit ETH (Non-ERC20 collateral currency)
      ✓ Deposit FIL (ERC20 non-collateral currency)
      ✓ Withdraw ETH with over amount input
      ✓ Deposit USDC (ERC20 collateral currency)
      ✓ Deposit WBTC (ERC20 collateral currency)
```

```
      ✓ Withdraw FIL (ERC20 non-collateral currency) with over amount input
    Deposit by multiple users
      ✓ Deposit FIL
      ✓ Withdraw by one user
      ✓ Withdraw from empty deposit
    Deposit new currency as collateral
      ✓ Register new currency as collateral
      ✓ Deposit TestToken
      ✓ Place an order
    Fill an borrowing order, Withdraw collateral
      ✓ Fill an order(WBTC)
      ✓ Fill an order(WFIL)
      ✓ Withdraw by borrower
      ✓ Withdraw by lender(empty deposit)
    Fill an lending order, Withdraw collateral
      ✓ Fill an order
      ✓ Withdraw by borrower
      ✓ Withdraw by lender(empty deposit)
    Fill orders on multiple markets, Withdraw collateral
      ✓ Fill an order on the FIL market
      ✓ Fill an order on the ETH market
      ✓ Withdraw by Alice
    Place orders, Withdraw collateral
      ✓ Deposit ETH
      ✓ Place orders
      ✓ Check withdrawable amount
      ✓ Withdraw ETH
    Withdraw non-collateral currencies while a active lending order exists
      ✓ Place lending orders for non-collateral currencies, WFIL and WBTC
      ✓ Withdraw all WFIL and WBTC deposit of bob and get deposit amount again
    Deposit and withdraw wFIL using MixinWallet
      ✓ Deposit wFIL on ReserveFund contract
      ✓ Withdraw all wFIL deposit on ReserveFund contract
      ✓ Deposit wFIL on ReserveFund contract using wallet transactions
      ✓ Withdraw all wFIL deposit on ReserveFund contract using wallet transaction
      ✓ Deposit wFIL on Liquidator contract
      ✓ Withdraw all wFIL deposit on Liquidator contract
      ✓ Deposit wFIL on Liquidator contract using wallet transactions
      ✓ Withdraw all wFIL deposit on liquidator contract using wallet transaction
    Fill an orders under min debt unit price, Withdraw collateral
      ✓ Fill an order with amount with over the min debt unit price
      ✓ Fill an order with amount with under the min debt unit price
      ✓ Check the withdrawable collateral amount of borrower
      ✓ Withdraw by borrower

  Integration Test: Emergency terminations
    Execute emergency termination & redemption
      Including only healthy users
        ✓ Fill an order on the ETH market with depositing ETH
        ✓ Fill an order on the FIL market with depositing USDC
        ✓ Execute emergency termination
        ✓ Execute forced redemption
        ✓ Withdraw all collateral
      Including an auto-rolled position
        ✓ Fill an order on the ETH market with depositing ETH
        ✓ Execute auto-roll
        ✓ Execute emergency termination
        ✓ Execute forced redemption
      Including a liquidation user
        ✓ Fill an order on the ETH market with depositing ETH
        ✓ Fill an order on the FIL market with depositing USDC
        ✓ Update a price feed to change the wFIL price
        ✓ Execute emergency termination
        ✓ Execute forced redemption
      Including an insolvent user
        ✓ Fill an order on the ETH market with depositing ETH
        ✓ Fill an order on the FIL market with depositing USDC
        ✓ Fill an order for a huge amount to store fees in the reserve funds
        ✓ Update a price feed to change the wFIL price
        ✓ Execute emergency termination
        ✓ Execute forced redemption
        ✓ Withdraw all collateral
  Integration Test: Itayose
    Execute Itayose on the single market without pre-order
      ✓ Fill an order
      ✓ Execute auto-roll
      ✓ Check the expected result before Itayose execution
      ✓ Execute Itayose without pre-order
    Execute Itayose with pre-order
      ✓ Fill an order
      ✓ Crate pre-orders
      ✓ Execute auto-roll
      ✓ Check the expected result before Itayose execution
      ✓ Execute Itayose with pre-order
    Execute Itayose with pre-order and execute clearing order process
      ✓ Crate pre-orders
      ✓ Check if clearing order process takes the opening price into accounts
    Execute Itayose with pre-order in same amount and execute clearing order process
      ✓ Crate pre-orders
      ✓ Check if clearing order process takes the opening price into accounts

  Integration Test: Liquidations
    Liquidations on FIL(non-collateral currency) market by ETH
      Increase FIL exchange rate, Execute liquidation once, Manage reserve funds
        ✓ Create orders
        ✓ Withdraw
┌─────────┬──────────┬────────────────┬─────────────────────────┬───────────────┬─────────────────────────
┬────────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │         PV(WFIL)        │ Deposit(WFIL) │      Deposit(ETH)
│ Deposit(USDC) │
├─────────┼──────────┼────────────────┼─────────────────────────┼───────────────┼─────────────────────────
┼────────────────┤
│ Before  │  '8368'  │   1916697600   │ '-200000000000000000000' │      '0'      │ '1000000000000000000'
│      '0'      │
│  After  │  '7575'  │   1916697600   │ '-99999999999999999999'  │      '0'      │ '552307134710000000'
│      '0'      │
└─────────┴──────────┴────────────────┴─────────────────────────┴───────────────┴─────────────────────────
┴────────────────┘
        ✓ Execute liquidation
      Increase FIL exchange rate, Execute liquidation twice
        ✓ Create orders
        ✓ Withdraw
┌─────────┬──────────┬────────────────┬─────────────────────────┬───────────────┬─────────────────────────
┬────────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │         PV(WFIL)        │ Deposit(WFIL) │      Deposit(ETH)
│ Deposit(USDC) │
├─────────┼──────────┼────────────────┼─────────────────────────┼───────────────┼─────────────────────────
┼────────────────┤
│ Before  │  '8748'  │   1924560000   │ '-200000000000000000000' │      '0'      │ '1000000000000000000'
│      '0'      │
│ After1  │  '8222'  │   1924560000   │ '-99999999999999999999'  │      '0'      │ '531957459015000000'
│      '0'      │
│ After2  │  '7340'  │   1924560000   │ '-50000000000000000000'  │      '0'      │ '297936188522500000'
│      '0'      │
└─────────┴──────────┴────────────────┴─────────────────────────┴───────────────┴─────────────────────────
┴────────────────┘
        ✓ Execute liquidation twice
      Execute auto-roll a borrowing position, Execute liquidation after auto-roll
        ✓ Create orders
        ✓ Withdraw
        ✓ Execute auto-roll
┌─────────┬──────────┬────────────────┬─────────────────────────┬───────────────┬─────────────────────────
┬────────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │         PV(WFIL)        │ Deposit(WFIL) │      Deposit(ETH)
│ Deposit(USDC) │
├─────────┼──────────┼────────────────┼─────────────────────────┼───────────────┼─────────────────────────
┼────────────────┤
│ Before  │  '8340'  │   1940284800   │ '-219273561643835616511' │      '0'      │ '1000000000000000000'
│      '0'      │
│  After  │  '7530'  │   1940284800   │ '-109636780822474227243' │      '0'      │ '553785413208300000'
│      '0'      │
└─────────┴──────────┴────────────────┴─────────────────────────┴───────────────┴─────────────────────────
┴────────────────┘
```

```
        ✓ Execute liquidation
      Liquidate partially due to insufficient collateral
        ✓ Create orders
        ✓ Withdraw
┌─────────┬──────────────────────────────────────────────────────────────────────────────────┬───────────
─────┬──────────────────────────┬───────────────┬───────────────────────┬───────────────┐
│ (index) │                                     Coverage                                     │
Maturity(WFIL) │         PV(WFIL)          │ Deposit(WFIL) │     Deposit(ETH)      │ Deposit(USDC) │
├─────────┼──────────────────────────────────────────────────────────────────────────────────┼───────────
─────┼──────────────────────────┼───────────────┼───────────────────────┼───────────────┤
│ Before  │                                     '22822'                                      │
1948147200   │ '-200000000000000000000' │      '0'      │ '1000000000000000000' │      '0'      │
│  After  │ '115792089237316195423570985008687907853269984665640564039457584007913129639935' │
1948147200   │ '-107453832845476795425' │      '0'      │          '0'          │      '0'      │
└─────────┴──────────────────────────────────────────────────────────────────────────────────┴───────────
─────┴──────────────────────────┴───────────────┴───────────────────────┴───────────────┘
        ✓ Execute liquidation
        ✓ Withdraw funds on Liquidator contract
      Liquidate partially due to insufficient collateral without the reserve fund after auto-roll
        ✓ Create orders
        ✓ Withdraw
        ✓ Execute auto-roll
┌─────────┬──────────────────────────────────────────────────────────────────────────────────┬───────────
─────┬──────────────────────────┬───────────────┬───────────────────────┬───────────────┐
│ (index) │                                     Coverage                                     │
Maturity(WFIL) │         PV(WFIL)          │ Deposit(WFIL) │     Deposit(ETH)      │ Deposit(USDC) │
├─────────┼──────────────────────────────────────────────────────────────────────────────────┼───────────
─────┼──────────────────────────┼───────────────┼───────────────────────┼───────────────┤
│ Before  │                                     '23829'                                      │
1963872000   │ '-208831963470319634763' │      '0'      │ '1000000000000000000' │      '0'      │
│  After  │ '115792089237316195423570985008687907853269984665640564039457584007913129639935' │
1963872000   │ '-125370549938800531471' │      '0'      │          '0'          │      '0'      │
└─────────┴──────────────────────────────────────────────────────────────────────────────────┴───────────
─────┴──────────────────────────┴───────────────┴───────────────────────┴───────────────┘
        ✓ Execute liquidation
      Liquidate a borrowing position using the user's deposits and lending positions
        ✓ Create orders on the USDC market
        ✓ Create orders on the FIL market
        ✓ Withdraw
┌──────────────┬──────────┬────────────────┬────────────────┬──────────────────────────┬─────────────┬───
────────────┬──────────────┬────────────────┐
│   (index)    │ Coverage │ Maturity(WFIL) │ Maturity(USDC) │         PV(WFIL)         │  PV(USDC)   │
Deposit(WFIL) │ Deposit(ETH) │ Deposit(USDC) │
├──────────────┼──────────┼────────────────┼────────────────┼──────────────────────────┼─────────────┼───
────────────┼──────────────┼────────────────┤
│ User(Before) │  '9398'  │   1971734400   │   1971734400   │ '-131451726312142587685' │ '997435247' │
'0'      │     '0'      │      '0'       │
│ User(After)  │  '7861'  │   1971734400   │   1971734400   │ '-65725863100426027158'  │ '596214050' │
'0'      │     '0'      │      '0'       │
└──────────────┴──────────┴────────────────┴────────────────┴──────────────────────────┴─────────────┴───
────────────┴──────────────┴────────────────┘
┌────────────┬──────────┬────────────────┬────────────────┬──────────┬──────────┬────────────────────────┬
──────────────┬────────────────┐
│  (index)   │ Coverage │ Maturity(WFIL) │ Maturity(USDC) │ PV(WFIL) │ PV(USDC) │     Deposit(WFIL)      │
Deposit(ETH) │ Deposit(USDC) │
├────────────┼──────────┼────────────────┼────────────────┼──────────┼──────────┼────────────────────────┼
──────────────┼────────────────┤
│ Liquidator │   '0'    │   1971734400   │   1971734400   │   '0'    │   '0'    │ '2950389949110578134' │
'0'      │     '0'      │
└────────────┴──────────┴────────────────┴────────────────┴──────────┴──────────┴────────────────────────┴
──────────────┴────────────────┘
        ✓ Execute liquidation
      Liquidate a borrowing position using the user's lending positions after two auto-rolls
        ✓ Create orders on the USDC market
        ✓ Create orders on the FIL market
        ✓ Execute auto-roll twice
        ✓ Withdraw
        ✓ Create orders for unwinding
┌──────────────┬──────────┬────────────────┬────────────────┬──────────────────────────┬──────────────┬───
────────────┬──────────────┬────────────────┐
│   (index)    │ Coverage │ Maturity(WFIL) │ Maturity(USDC) │         PV(WFIL)         │   PV(USDC)   │
Deposit(WFIL) │ Deposit(ETH) │ Deposit(USDC) │
```

```
├───────────────┼──────────┼────────────────┼────────────────┼──────────────────────────┼──────────────┼─
──────────────┼──────────────┼──────────────┤
│ User(Before) │  '8834'  │   1995321600   │   1995321600   │ '-1372314006165244185 89' │ '1033909158' │
'0'      │    '0'     │      '0'      │
│ User(After)  │  '7103'  │   1995321600   │   1995321600   │ '-68615700384420561765' │ '642971194'  │
'0'      │    '0'     │      '0'      │
└───────────────┴──────────┴────────────────┴────────────────┴──────────────────────────┴──────────────┴─
──────────────┴──────────────┴──────────────┘

┌────────────────────┬──────────┬────────────────┬────────────────┬──────────────────────────┬────────────
─┬──────────────┬──────────────┬──────────────┐
│      (index)       │ Coverage │ Maturity(WFIL) │ Maturity(USDC) │         PV(WFIL)         │  PV(USDC)
│ Deposit(WFIL) │ Deposit(ETH) │ Deposit(USDC) │
├────────────────────┼──────────┼────────────────┼────────────────┼──────────────────────────┼────────────
─┼──────────────┼──────────────┼──────────────┤
│ Liquidator(Before) │   '0'    │   1995321600   │   1995321600   │           '0'            │     '0'
│      '0'      │     '0'      │ '999928215'  │
│ Liquidator(After)  │  '2795'  │   1995321600   │   1995321600   │ '-68615700232103856824' │ '383630712'
│      '0'      │     '0'      │ '999928215'  │
└────────────────────┴──────────┴────────────────┴────────────────┴──────────────────────────┴────────────
─┴──────────────┴──────────────┴──────────────┘
        ✓ Execute liquidation
      Liquidate a borrowing position using the user's multiple lending positions
        ✓ Create orders on the multiple USDC markets
        ✓ Create orders on the FIL market
        ✓ Withdraw
┌───────────────┬──────────┬────────────────┬────────────────┬────────────────┬──────────────────────────
──┬─────────────┬─────────────┬───────────────┬──────────────┬───────────────┐
│   (index)     │ Coverage │ Maturity(WFIL) │ Maturity(USDC-1) │ Maturity(USDC-2) │          PV(WFIL)
│ PV(USDC-1)  │ PV(USDC-2)  │ Deposit(WFIL) │ Deposit(ETH) │ Deposit(USDC) │
├───────────────┼──────────┼────────────────┼────────────────┼────────────────┼──────────────────────────
──┼─────────────┼─────────────┼───────────────┼──────────────┼───────────────┤
│ User(Before) │  '9415'  │   2003184000   │   2003184000   │   2011651200   │
'-131451726312142587685' │ '332478416' │ '663167006' │      '0'      │      '0'     │      '0'      │
│ User(After)  │  '7885'  │   2003184000   │   2003184000   │   2011651200   │
'-65725863100426027158'  │     '0'     │ '594424225' │      '0'      │      '0'     │      '0'      │
└───────────────┴──────────┴────────────────┴────────────────┴────────────────┴──────────────────────────
──┴─────────────┴─────────────┴───────────────┴──────────────┴───────────────┘

┌────────────┬──────────┬────────────────┬──────────────────┬──────────────────┬──────────┬────────────┬─
────────────┬─────────────────────────┬──────────────┬───────────────┐
│  (index)   │ Coverage │ Maturity(WFIL) │ Maturity(USDC-1) │ Maturity(USDC-2) │ PV(WFIL) │ PV(USDC-1) │
PV(USDC-2) │      Deposit(WFIL)      │ Deposit(ETH) │ Deposit(USDC) │
├────────────┼──────────┼────────────────┼──────────────────┼──────────────────┼──────────┼────────────┼─
────────────┼─────────────────────────┼──────────────┼───────────────┤
│ Liquidator │   '0'    │   2003184000   │    2003184000    │    2011651200    │   '0'    │    '0'     │
'0'     │ '2921787722795650692' │     '0'      │      '0'      │
└────────────┴──────────┴────────────────┴──────────────────┴──────────────────┴──────────┴────────────┴─
────────────┴─────────────────────────┴──────────────┴───────────────┘
        ✓ Execute liquidation
    Liquidations on multiple market
┌─────────┬──────────┬────────────────┬────────────────┬──────────────────────────┬──────────────┬───────
────────┬───────────────────────┬───────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │ Maturity(USDC) │         PV(WFIL)         │   PV(USDC)   │
Deposit(WFIL) │     Deposit(ETH)      │ Deposit(USDC) │
├─────────┼──────────┼────────────────┼────────────────┼──────────────────────────┼──────────────┼───────
────────┼───────────────────────┼───────────────┤
│ Before1 │  '7775'  │   2019513600   │   2019513600   │ '-200000000000000000000' │ '-600000000' │
'0'      │ '175000000000000000' │      '0'      │
│ Before2 │  '8210'  │   2019513600   │   2019513600   │ '-200000000000000000000' │ '-600000000' │
'0'      │ '175000000000000000' │      '0'      │
│  After  │  '7321'  │   2019513600   │   2019513600   │ '-28293910964595707418' │ '-600000000' │
'0'      │ '98128409012000000'  │      '0'      │
└─────────┴──────────┴────────────────┴────────────────┴──────────────────────────┴──────────────┴───────
────────┴───────────────────────┴───────────────┘
      ✓ Take orders from both FIL & USDC markets, Liquidate the larger position
┌─────────┬──────────┬────────────────┬────────────────┬──────────────────────────┬──────────────┬───────
────────┬───────────────────────┬───────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │ Maturity(USDC) │         PV(WFIL)         │   PV(USDC)   │
Deposit(WFIL) │     Deposit(ETH)      │ Deposit(USDC) │
├─────────┼──────────┼────────────────┼────────────────┼──────────────────────────┼──────────────┼───────
────────┼───────────────────────┼───────────────┤
│ Before1 │  '7775'  │   2027376000   │   2027376000   │ '-200000000000000000000' │ '-600000000' │
'0'      │ '175000000000000000' │      '0'      │
```

```
│ Before2 │  '8210'  │   2027376000    │   2027376000    │ '-20000000000000000000' │ '-600000000' │
'0'      │ '1750000000000000000' │      '0'      │
│  After  │  '7552'  │   2027376000    │   2027376000    │ '-20000000000000000000' │     '0'      │
'0'      │ '1107953910820000000' │      '0'      │
└─────────┴──────────┴─────────────────┴─────────────────┴─────────────────────────┴──────────────┴─────────
────────┴─────────────────────┴───────────────┘
      ✓ Take orders from both FIL & USDC markets, Liquidate the smaller position
    Delisting
      Repay and redeem positions
        ✓ Create orders
        ✓ Delist a currency
┌─────────┬──────────┬─────────────────┬──────────────────────────┬─────────────────────────┬──────────────
──────────┬────────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │         PV(WFIL)         │      Deposit(WFIL)      │
Deposit(ETH)       │ Deposit(USDC) │
├─────────┼──────────┼─────────────────┼──────────────────────────┼─────────────────────────┼──────────────
──────────┼────────────────┤
│ Before  │  '3803'  │   2035238400    │ '-20000000000000000000' │ '20000000000000000000' │
'2000000000000000000' │      '0'      │
│  After  │   '0'    │   2035238400    │           '0'            │           '0'           │
'2000000000000000000' │      '0'      │
└─────────┴──────────┴─────────────────┴──────────────────────────┴─────────────────────────┴──────────────
──────────┴────────────────┘
        ✓ Execute repayment & redemption
      Force a repayment of a borrowing position
        ✓ Create orders
        ✓ Withdraw
┌─────────┬──────────┬─────────────────┬──────────────────────────┬───────────────┬──────────────────────
┬────────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │         PV(WFIL)         │ Deposit(WFIL) │     Deposit(ETH)
│ Deposit(USDC) │
├─────────┼──────────┼─────────────────┼──────────────────────────┼───────────────┼──────────────────────
┼────────────────┤
│ Before  │  '3803'  │   2043100800    │ '-20000000000000000000' │      '0'      │ '2000000000000000000'
│      '0'      │
│  After  │   '0'    │   2043100800    │           '0'            │      '0'      │ '1152096846041666667'
│      '0'      │
└─────────┴──────────┴─────────────────┴──────────────────────────┴───────────────┴──────────────────────
┴────────────────┘
        ✓ Execute forced repayment
      Force a repayment of a insolvent borrowing position
        ✓ Create orders
        ✓ Withdraw
┌─────────┬──────────────────────────────────────────────────────────────────────────────────────┬─────────
─────┬─────────────────────────┬───────────────┬─────────────────────┬────────────────┐
│ (index) │                                       Coverage                                       │
Maturity(WFIL) │         PV(WFIL)         │ Deposit(WFIL) │     Deposit(ETH)      │ Deposit(USDC) │
├─────────┼──────────────────────────────────────────────────────────────────────────────────────┼─────────
─────┼─────────────────────────┼───────────────┼─────────────────────┼────────────────┤
│ Before  │                                       '11411'                                        │
2050963200    │ '-20000000000000000000' │      '0'      │ '2000000000000000000' │      '0'      │
│  After  │ '115792089237316195423570985008687907853269984665640564039457584007913129639935' │
2050963200    │ '-39754086019483321679' │      '0'      │          '0'          │      '0'      │
└─────────┴──────────────────────────────────────────────────────────────────────────────────────┴─────────
─────┴─────────────────────────┴───────────────┴─────────────────────┴────────────────┘
        ✓ Execute forced repayment
      Force a repayment of a borrowing position after auto-roll
        ✓ Create orders
        ✓ Withdraw
        ✓ Execute auto-roll
┌─────────┬──────────┬─────────────────┬──────────────────────────┬───────────────┬──────────────────────
┬────────────────┐
│ (index) │ Coverage │ Maturity(WFIL) │         PV(WFIL)         │ Deposit(WFIL) │     Deposit(ETH)
│ Deposit(USDC) │
├─────────┼──────────┼─────────────────┼──────────────────────────┼───────────────┼──────────────────────
┼────────────────┤
│ Before  │  '3574'  │   2066688000    │ '-187948767123287671212' │      '0'      │ '2000000000000000000'
│      '0'      │
│  After  │   '0'    │   2066688000    │           '0'            │      '0'      │ '1203188237867920378'
│      '0'      │
└─────────┴──────────┴─────────────────┴──────────────────────────┴───────────────┴──────────────────────
┴────────────────┘
```

```
        ✓ Execute forced repayment
      Force a repayment of a insolvent borrowing position after auto-roll
        ✓ Create orders
        ✓ Withdraw
        ✓ Execute auto-roll
┌─────────┬───────────────────────────────────────────────────────────────────────────────┬─────────────
─────┬────────────────────────┬───────────────┬────────────────────────┬───────────────┐
│ (index) │                                    Coverage                                   │
Maturity(WFIL) │         PV(WFIL)         │ Deposit(WFIL) │     Deposit(ETH)       │ Deposit(USDC) │
├─────────┼───────────────────────────────────────────────────────────────────────────────┼─────────────
─────┼────────────────────────┼───────────────┼────────────────────────┼───────────────┤
│ Before  │                                    '11914'                                    │
2082412800   │ '-20883196347031963466 7' │      '0'      │ '2000000000000000000' │      '0'      │
│  After  │ '115792089237316195423570985008687907853269984665640564039457584007913129639935' │
2082412800   │ '-4858604948980295636 1'  │      '0'      │          '0'           │      '0'      │
└─────────┴───────────────────────────────────────────────────────────────────────────────┴─────────────
─────┴────────────────────────┴───────────────┴────────────────────────┴───────────────┘
        ✓ Execute forced repayment

  Integration Test: Order Book
    Market orders
      Add orders using the same currency as the collateral, Fill the order, Unwind the ETH borrowing
order
        ✓ Create orders
        ✓ Deposit ETH
        ✓ Fill an order on the ETH market
        ✓ Check collateral
        ✓ Unwind all positions
      Add orders using the different currency as the collateral, Fill the order, Unwind the non-ETH
borrowing order
        ✓ Deposit ETH
        ✓ Fill an order on the FIL market
        ✓ Check collateral
        ✓ Unwind all positions
      Fill the order, Unwind the lending order
        ✓ Deposit ETH
        ✓ Fill an order on the FIL market
        ✓ Check lending position
        ✓ Unwind a lending position
      Fill orders in multiple markets, Unwind partially
        ✓ Deposit ETH
        ✓ Fill an order on the FIL market
        ✓ Fill an order on the ETH market
        ✓ Check collateral
        ✓ Unwind positions partially
      Fill multiple orders on different order sides in multiple markets
        ✓ Deposit ETH
        ✓ Fill an order on the ETH market(1)
        ✓ Fill an order on the ETH market(2)
      Fill orders, Trigger circuit breakers by one order
        ✓ Fill an order to determine the market unit price
        ✓ Deposit ETH
        ✓ Fill orders on the FIL market
      Fill orders, Trigger circuit breakers by multiple orders
        ✓ Deposit ETH
        ✓ Fill an order to determine the market unit price
        ✓ Fill orders on the FIL market
      Unwind lending position used as collateral
        ✓ Deposit ETH
        ✓ Fill an order on the FIL market(1)
        ✓ Fill an order on the FIL market(2)
        ✓ Fail to unwind positions due to insufficient collateral
    Limit orders
      Fill a borrowing order with the same amount
        ✓ Create users
        ✓ Fill an order
        ✓ Check orders
      Fill a borrowing order with less amount
        ✓ Create users
        ✓ Fill an order
        ✓ Check orders
      Fill a borrowing order with greater amount
```

```
        ✓ Create users
        ✓ Fill an order
        ✓ Check orders
      Fill a lending order with the same amount
        ✓ Create users
        ✓ Fill an order
        ✓ Check orders
      Fill a lending order with less amount
        ✓ Create users
        ✓ Fill an order
        ✓ Check orders
      Fill a lending order with greater amount
        ✓ Create users
        ✓ Fill an order
        ✓ Check orders
    Order Cancellation
      Place a borrowing order, Cancel orders
        ✓ Deposit ETH
        ✓ Place a borrowing order on the FIL market
        ✓ Cancel an order
      Place a lending order by a user who has a deposit, Cancel orders
        ✓ Deposit ETH
        ✓ Place a lending order on the FIL market
        ✓ Cancel an order

  Performance Test: Auto-rolls
    Execute auto-rolls for 200 years
      ✓ Fill an order
      ✓ Execute auto-roll (1st time)
      ✓ Execute auto-roll (2nd time)
      ✓ Execute auto-roll (3rd time)
      ✓ Execute auto-roll (4th time)
      ✓ Execute auto-roll (5th time)
      ✓ Execute auto-roll (6th time)
      ✓ Execute auto-roll (7th time)
      ✓ Execute auto-roll (8th time)
      ✓ Execute auto-roll (9th time)
      ✓ Execute auto-roll (10th time)
      ✓ Execute auto-roll (11th time)
      ✓ Execute auto-roll (12th time)
      ✓ Execute auto-roll (13th time)
      ✓ Execute auto-roll (14th time)
      ✓ Execute auto-roll (15th time)
      ✓ Execute auto-roll (16th time)
      ✓ Execute auto-roll (17th time)
      ✓ Execute auto-roll (18th time)
      ✓ Execute auto-roll (19th time)
      ✓ Execute auto-roll (20th time)
      ✓ Execute auto-roll (21st time)
      ✓ Execute auto-roll (22nd time)
      ✓ Execute auto-roll (23rd time)
      ✓ Execute auto-roll (24th time)
      ✓ Execute auto-roll (25th time)
      ✓ Execute auto-roll (26th time)
      ✓ Execute auto-roll (27th time)
      ✓ Execute auto-roll (28th time)
      ✓ Execute auto-roll (29th time)
      ✓ Execute auto-roll (30th time)
      ✓ Execute auto-roll (31st time)
      ✓ Execute auto-roll (32nd time)
      ✓ Execute auto-roll (33rd time)
      ✓ Execute auto-roll (34th time)
      ✓ Execute auto-roll (35th time)
      ✓ Execute auto-roll (36th time)
      ✓ Execute auto-roll (37th time)
      ✓ Execute auto-roll (38th time)
      ✓ Execute auto-roll (39th time)
      ✓ Execute auto-roll (40th time)
      ✓ Execute auto-roll (41st time)
      ✓ Execute auto-roll (42nd time)
      ✓ Execute auto-roll (43rd time)
      ✓ Execute auto-roll (44th time)
      ✓ Execute auto-roll (45th time)
      ✓ Execute auto-roll (46th time)
      ✓ Execute auto-roll (47th time)
      ✓ Execute auto-roll (48th time)
      ✓ Execute auto-roll (49th time)
      ✓ Execute auto-roll (50th time)
      ✓ Execute auto-roll (51st time)
      ✓ Execute auto-roll (52nd time)
      ✓ Execute auto-roll (53rd time)
      ✓ Execute auto-roll (54th time)
      ✓ Execute auto-roll (55th time)
      ✓ Execute auto-roll (56th time)
      ✓ Execute auto-roll (57th time)
      ✓ Execute auto-roll (58th time)
      ✓ Execute auto-roll (59th time)
      ✓ Execute auto-roll (60th time)
      ✓ Execute auto-roll (61st time)
      ✓ Execute auto-roll (62nd time)
      ✓ Execute auto-roll (63rd time)
      ✓ Execute auto-roll (64th time)
      ✓ Execute auto-roll (65th time)
      ✓ Execute auto-roll (66th time)
      ✓ Execute auto-roll (67th time)
      ✓ Execute auto-roll (68th time)
      ✓ Execute auto-roll (69th time)
      ✓ Execute auto-roll (70th time)
      ✓ Execute auto-roll (71st time)
      ✓ Execute auto-roll (72nd time)
      ✓ Execute auto-roll (73rd time)
      ✓ Execute auto-roll (74th time)
      ✓ Execute auto-roll (75th time)
      ✓ Execute auto-roll (76th time)
      ✓ Execute auto-roll (77th time)
      ✓ Execute auto-roll (78th time)
      ✓ Execute auto-roll (79th time)
      ✓ Execute auto-roll (80th time)
      ✓ Execute auto-roll (81st time)
      ✓ Execute auto-roll (82nd time)
      ✓ Execute auto-roll (83rd time)
      ✓ Execute auto-roll (84th time)
      ✓ Execute auto-roll (85th time)
      ✓ Execute auto-roll (86th time)
      ✓ Execute auto-roll (87th time)
      ✓ Execute auto-roll (88th time)
      ✓ Execute auto-roll (89th time)
      ✓ Execute auto-roll (90th time)
      ✓ Execute auto-roll (91st time)
      ✓ Execute auto-roll (92nd time)
      ✓ Execute auto-roll (93rd time)
      ✓ Execute auto-roll (94th time)
      ✓ Execute auto-roll (95th time)
      ✓ Execute auto-roll (96th time)
      ✓ Execute auto-roll (97th time)
      ✓ Execute auto-roll (98th time)
      ✓ Execute auto-roll (99th time)
      ✓ Execute auto-roll (100th time)
      ✓ Execute auto-roll (101st time)
      ✓ Execute auto-roll (102nd time)
      ✓ Execute auto-roll (103rd time)
      ✓ Execute auto-roll (104th time)
      ✓ Execute auto-roll (105th time)
      ✓ Execute auto-roll (106th time)
      ✓ Execute auto-roll (107th time)
      ✓ Execute auto-roll (108th time)
      ✓ Execute auto-roll (109th time)
      ✓ Execute auto-roll (110th time)
      ✓ Execute auto-roll (111th time)
      ✓ Execute auto-roll (112th time)
      ✓ Execute auto-roll (113th time)
      ✓ Execute auto-roll (114th time)
      ✓ Execute auto-roll (115th time)
      ✓ Execute auto-roll (116th time)
```

```
✓ Execute auto-roll (117th time)
✓ Execute auto-roll (118th time)
✓ Execute auto-roll (119th time)
✓ Execute auto-roll (120th time)
✓ Execute auto-roll (121st time)
✓ Execute auto-roll (122nd time)
✓ Execute auto-roll (123rd time)
✓ Execute auto-roll (124th time)
✓ Execute auto-roll (125th time)
✓ Execute auto-roll (126th time)
✓ Execute auto-roll (127th time)
✓ Execute auto-roll (128th time)
✓ Execute auto-roll (129th time)
✓ Execute auto-roll (130th time)
✓ Execute auto-roll (131st time)
✓ Execute auto-roll (132nd time)
✓ Execute auto-roll (133rd time)
✓ Execute auto-roll (134th time)
✓ Execute auto-roll (135th time)
✓ Execute auto-roll (136th time)
✓ Execute auto-roll (137th time)
✓ Execute auto-roll (138th time)
✓ Execute auto-roll (139th time)
✓ Execute auto-roll (140th time)
✓ Execute auto-roll (141st time)
✓ Execute auto-roll (142nd time)
✓ Execute auto-roll (143rd time)
✓ Execute auto-roll (144th time)
✓ Execute auto-roll (145th time)
✓ Execute auto-roll (146th time)
✓ Execute auto-roll (147th time)
✓ Execute auto-roll (148th time)
✓ Execute auto-roll (149th time)
✓ Execute auto-roll (150th time)
✓ Execute auto-roll (151st time)
✓ Execute auto-roll (152nd time)
✓ Execute auto-roll (153rd time)
✓ Execute auto-roll (154th time)
✓ Execute auto-roll (155th time)
✓ Execute auto-roll (156th time)
✓ Execute auto-roll (157th time)
✓ Execute auto-roll (158th time)
✓ Execute auto-roll (159th time)
✓ Execute auto-roll (160th time)
✓ Execute auto-roll (161st time)
✓ Execute auto-roll (162nd time)
✓ Execute auto-roll (163rd time)
✓ Execute auto-roll (164th time)
✓ Execute auto-roll (165th time)
✓ Execute auto-roll (166th time)
✓ Execute auto-roll (167th time)
✓ Execute auto-roll (168th time)
✓ Execute auto-roll (169th time)
✓ Execute auto-roll (170th time)
✓ Execute auto-roll (171st time)
✓ Execute auto-roll (172nd time)
✓ Execute auto-roll (173rd time)
✓ Execute auto-roll (174th time)
✓ Execute auto-roll (175th time)
✓ Execute auto-roll (176th time)
✓ Execute auto-roll (177th time)
✓ Execute auto-roll (178th time)
✓ Execute auto-roll (179th time)
✓ Execute auto-roll (180th time)
✓ Execute auto-roll (181st time)
✓ Execute auto-roll (182nd time)
✓ Execute auto-roll (183rd time)
✓ Execute auto-roll (184th time)
✓ Execute auto-roll (185th time)
✓ Execute auto-roll (186th time)
✓ Execute auto-roll (187th time)
✓ Execute auto-roll (188th time)
✓ Execute auto-roll (189th time)
✓ Execute auto-roll (190th time)
✓ Execute auto-roll (191st time)
✓ Execute auto-roll (192nd time)
✓ Execute auto-roll (193rd time)
✓ Execute auto-roll (194th time)
✓ Execute auto-roll (195th time)
✓ Execute auto-roll (196th time)
✓ Execute auto-roll (197th time)
✓ Execute auto-roll (198th time)
✓ Execute auto-roll (199th time)
✓ Execute auto-roll (200th time)
✓ Execute auto-roll (201st time)
✓ Execute auto-roll (202nd time)
✓ Execute auto-roll (203rd time)
✓ Execute auto-roll (204th time)
✓ Execute auto-roll (205th time)
✓ Execute auto-roll (206th time)
✓ Execute auto-roll (207th time)
✓ Execute auto-roll (208th time)
✓ Execute auto-roll (209th time)
✓ Execute auto-roll (210th time)
✓ Execute auto-roll (211th time)
✓ Execute auto-roll (212th time)
✓ Execute auto-roll (213th time)
✓ Execute auto-roll (214th time)
✓ Execute auto-roll (215th time)
✓ Execute auto-roll (216th time)
✓ Execute auto-roll (217th time)
✓ Execute auto-roll (218th time)
✓ Execute auto-roll (219th time)
✓ Execute auto-roll (220th time)
✓ Execute auto-roll (221st time)
✓ Execute auto-roll (222nd time)
✓ Execute auto-roll (223rd time)
✓ Execute auto-roll (224th time)
✓ Execute auto-roll (225th time)
✓ Execute auto-roll (226th time)
✓ Execute auto-roll (227th time)
✓ Execute auto-roll (228th time)
✓ Execute auto-roll (229th time)
✓ Execute auto-roll (230th time)
✓ Execute auto-roll (231st time)
✓ Execute auto-roll (232nd time)
✓ Execute auto-roll (233rd time)
✓ Execute auto-roll (234th time)
✓ Execute auto-roll (235th time)
✓ Execute auto-roll (236th time)
✓ Execute auto-roll (237th time)
✓ Execute auto-roll (238th time)
✓ Execute auto-roll (239th time)
✓ Execute auto-roll (240th time)
✓ Execute auto-roll (241st time)
✓ Execute auto-roll (242nd time)
✓ Execute auto-roll (243rd time)
✓ Execute auto-roll (244th time)
✓ Execute auto-roll (245th time)
✓ Execute auto-roll (246th time)
✓ Execute auto-roll (247th time)
✓ Execute auto-roll (248th time)
✓ Execute auto-roll (249th time)
✓ Execute auto-roll (250th time)
✓ Execute auto-roll (251st time)
✓ Execute auto-roll (252nd time)
✓ Execute auto-roll (253rd time)
✓ Execute auto-roll (254th time)
✓ Execute auto-roll (255th time)
✓ Execute auto-roll (256th time)
✓ Execute auto-roll (257th time)
✓ Execute auto-roll (258th time)
✓ Execute auto-roll (259th time)
✓ Execute auto-roll (260th time)
```

```
    ✓ Execute auto-roll (261st time)
    ✓ Execute auto-roll (262nd time)
    ✓ Execute auto-roll (263rd time)
    ✓ Execute auto-roll (264th time)
    ✓ Execute auto-roll (265th time)
    ✓ Execute auto-roll (266th time)
    ✓ Execute auto-roll (267th time)
    ✓ Execute auto-roll (268th time)
    ✓ Execute auto-roll (269th time)
    ✓ Execute auto-roll (270th time)
    ✓ Execute auto-roll (271st time)
    ✓ Execute auto-roll (272nd time)
    ✓ Execute auto-roll (273rd time)
    ✓ Execute auto-roll (274th time)
    ✓ Execute auto-roll (275th time)
    ✓ Execute auto-roll (276th time)
    ✓ Execute auto-roll (277th time)
    ✓ Execute auto-roll (278th time)
    ✓ Execute auto-roll (279th time)
    ✓ Execute auto-roll (280th time)
    ✓ Execute auto-roll (281st time)
    ✓ Execute auto-roll (282nd time)
    ✓ Execute auto-roll (283rd time)
    ✓ Execute auto-roll (284th time)
    ✓ Execute auto-roll (285th time)
    ✓ Execute auto-roll (286th time)
    ✓ Execute auto-roll (287th time)
    ✓ Execute auto-roll (288th time)
    ✓ Execute auto-roll (289th time)
    ✓ Execute auto-roll (290th time)
    ✓ Execute auto-roll (291st time)
    ✓ Execute auto-roll (292nd time)
    ✓ Execute auto-roll (293rd time)
    ✓ Execute auto-roll (294th time)
    ✓ Execute auto-roll (295th time)
    ✓ Execute auto-roll (296th time)
    ✓ Execute auto-roll (297th time)
    ✓ Execute auto-roll (298th time)
    ✓ Execute auto-roll (299th time)
    ✓ Execute auto-roll (300th time)
    ✓ Execute auto-roll (301st time)
    ✓ Execute auto-roll (302nd time)
    ✓ Execute auto-roll (303rd time)
    ✓ Execute auto-roll (304th time)
    ✓ Execute auto-roll (305th time)
    ✓ Execute auto-roll (306th time)
    ✓ Execute auto-roll (307th time)
    ✓ Execute auto-roll (308th time)
    ✓ Execute auto-roll (309th time)
    ✓ Execute auto-roll (310th time)
    ✓ Execute auto-roll (311th time)
    ✓ Execute auto-roll (312th time)
    ✓ Execute auto-roll (313th time)
    ✓ Execute auto-roll (314th time)
    ✓ Execute auto-roll (315th time)
    ✓ Execute auto-roll (316th time)
    ✓ Execute auto-roll (317th time)
    ✓ Execute auto-roll (318th time)
    ✓ Execute auto-roll (319th time)
    ✓ Execute auto-roll (320th time)
    ✓ Execute auto-roll (321st time)
    ✓ Execute auto-roll (322nd time)
    ✓ Execute auto-roll (323rd time)
    ✓ Execute auto-roll (324th time)
    ✓ Execute auto-roll (325th time)
    ✓ Execute auto-roll (326th time)
    ✓ Execute auto-roll (327th time)
    ✓ Execute auto-roll (328th time)
    ✓ Execute auto-roll (329th time)
    ✓ Execute auto-roll (330th time)
    ✓ Execute auto-roll (331st time)
    ✓ Execute auto-roll (332nd time)
```

```
✓ Execute auto-roll (333rd time)
✓ Execute auto-roll (334th time)
✓ Execute auto-roll (335th time)
✓ Execute auto-roll (336th time)
✓ Execute auto-roll (337th time)
✓ Execute auto-roll (338th time)
✓ Execute auto-roll (339th time)
✓ Execute auto-roll (340th time)
✓ Execute auto-roll (341st time)
✓ Execute auto-roll (342nd time)
✓ Execute auto-roll (343rd time)
✓ Execute auto-roll (344th time)
✓ Execute auto-roll (345th time)
✓ Execute auto-roll (346th time)
✓ Execute auto-roll (347th time)
✓ Execute auto-roll (348th time)
✓ Execute auto-roll (349th time)
✓ Execute auto-roll (350th time)
✓ Execute auto-roll (351st time)
✓ Execute auto-roll (352nd time)
✓ Execute auto-roll (353rd time)
✓ Execute auto-roll (354th time)
✓ Execute auto-roll (355th time)
✓ Execute auto-roll (356th time)
✓ Execute auto-roll (357th time)
✓ Execute auto-roll (358th time)
✓ Execute auto-roll (359th time)
✓ Execute auto-roll (360th time)
✓ Execute auto-roll (361st time)
✓ Execute auto-roll (362nd time)
✓ Execute auto-roll (363rd time)
✓ Execute auto-roll (364th time)
✓ Execute auto-roll (365th time)
✓ Execute auto-roll (366th time)
✓ Execute auto-roll (367th time)
✓ Execute auto-roll (368th time)
✓ Execute auto-roll (369th time)
✓ Execute auto-roll (370th time)
✓ Execute auto-roll (371st time)
✓ Execute auto-roll (372nd time)
✓ Execute auto-roll (373rd time)
✓ Execute auto-roll (374th time)
✓ Execute auto-roll (375th time)
✓ Execute auto-roll (376th time)
✓ Execute auto-roll (377th time)
✓ Execute auto-roll (378th time)
✓ Execute auto-roll (379th time)
✓ Execute auto-roll (380th time)
✓ Execute auto-roll (381st time)
✓ Execute auto-roll (382nd time)
✓ Execute auto-roll (383rd time)
✓ Execute auto-roll (384th time)
✓ Execute auto-roll (385th time)
✓ Execute auto-roll (386th time)
✓ Execute auto-roll (387th time)
✓ Execute auto-roll (388th time)
✓ Execute auto-roll (389th time)
✓ Execute auto-roll (390th time)
✓ Execute auto-roll (391st time)
✓ Execute auto-roll (392nd time)
✓ Execute auto-roll (393rd time)
✓ Execute auto-roll (394th time)
✓ Execute auto-roll (395th time)
✓ Execute auto-roll (396th time)
✓ Execute auto-roll (397th time)
✓ Execute auto-roll (398th time)
✓ Execute auto-roll (399th time)
✓ Execute auto-roll (400th time)
✓ Execute auto-roll (401st time)
✓ Execute auto-roll (402nd time)
✓ Execute auto-roll (403rd time)
✓ Execute auto-roll (404th time)
✓ Execute auto-roll (405th time)
✓ Execute auto-roll (406th time)
✓ Execute auto-roll (407th time)
✓ Execute auto-roll (408th time)
✓ Execute auto-roll (409th time)
✓ Execute auto-roll (410th time)
✓ Execute auto-roll (411th time)
✓ Execute auto-roll (412th time)
✓ Execute auto-roll (413th time)
✓ Execute auto-roll (414th time)
✓ Execute auto-roll (415th time)
✓ Execute auto-roll (416th time)
✓ Execute auto-roll (417th time)
✓ Execute auto-roll (418th time)
✓ Execute auto-roll (419th time)
✓ Execute auto-roll (420th time)
✓ Execute auto-roll (421st time)
✓ Execute auto-roll (422nd time)
✓ Execute auto-roll (423rd time)
✓ Execute auto-roll (424th time)
✓ Execute auto-roll (425th time)
✓ Execute auto-roll (426th time)
✓ Execute auto-roll (427th time)
✓ Execute auto-roll (428th time)
✓ Execute auto-roll (429th time)
✓ Execute auto-roll (430th time)
✓ Execute auto-roll (431st time)
✓ Execute auto-roll (432nd time)
✓ Execute auto-roll (433rd time)
✓ Execute auto-roll (434th time)
✓ Execute auto-roll (435th time)
✓ Execute auto-roll (436th time)
✓ Execute auto-roll (437th time)
✓ Execute auto-roll (438th time)
✓ Execute auto-roll (439th time)
✓ Execute auto-roll (440th time)
✓ Execute auto-roll (441st time)
✓ Execute auto-roll (442nd time)
✓ Execute auto-roll (443rd time)
✓ Execute auto-roll (444th time)
✓ Execute auto-roll (445th time)
✓ Execute auto-roll (446th time)
✓ Execute auto-roll (447th time)
✓ Execute auto-roll (448th time)
✓ Execute auto-roll (449th time)
✓ Execute auto-roll (450th time)
✓ Execute auto-roll (451st time)
✓ Execute auto-roll (452nd time)
✓ Execute auto-roll (453rd time)
✓ Execute auto-roll (454th time)
✓ Execute auto-roll (455th time)
✓ Execute auto-roll (456th time)
✓ Execute auto-roll (457th time)
✓ Execute auto-roll (458th time)
✓ Execute auto-roll (459th time)
✓ Execute auto-roll (460th time)
✓ Execute auto-roll (461st time)
✓ Execute auto-roll (462nd time)
✓ Execute auto-roll (463rd time)
✓ Execute auto-roll (464th time)
✓ Execute auto-roll (465th time)
✓ Execute auto-roll (466th time)
✓ Execute auto-roll (467th time)
✓ Execute auto-roll (468th time)
✓ Execute auto-roll (469th time)
✓ Execute auto-roll (470th time)
✓ Execute auto-roll (471st time)
✓ Execute auto-roll (472nd time)
✓ Execute auto-roll (473rd time)
✓ Execute auto-roll (474th time)
✓ Execute auto-roll (475th time)
✓ Execute auto-roll (476th time)
✓ Execute auto-roll (477th time)
✓ Execute auto-roll (478th time)
✓ Execute auto-roll (479th time)
✓ Execute auto-roll (480th time)
✓ Execute auto-roll (481st time)
✓ Execute auto-roll (482nd time)
✓ Execute auto-roll (483rd time)
✓ Execute auto-roll (484th time)
✓ Execute auto-roll (485th time)
✓ Execute auto-roll (486th time)
✓ Execute auto-roll (487th time)
✓ Execute auto-roll (488th time)
✓ Execute auto-roll (489th time)
✓ Execute auto-roll (490th time)
✓ Execute auto-roll (491st time)
✓ Execute auto-roll (492nd time)
✓ Execute auto-roll (493rd time)
✓ Execute auto-roll (494th time)
✓ Execute auto-roll (495th time)
✓ Execute auto-roll (496th time)
✓ Execute auto-roll (497th time)
✓ Execute auto-roll (498th time)
✓ Execute auto-roll (499th time)
✓ Execute auto-roll (500th time)
✓ Execute auto-roll (501st time)
✓ Execute auto-roll (502nd time)
✓ Execute auto-roll (503rd time)
✓ Execute auto-roll (504th time)
✓ Execute auto-roll (505th time)
✓ Execute auto-roll (506th time)
✓ Execute auto-roll (507th time)
✓ Execute auto-roll (508th time)
✓ Execute auto-roll (509th time)
✓ Execute auto-roll (510th time)
✓ Execute auto-roll (511th time)
✓ Execute auto-roll (512th time)
✓ Execute auto-roll (513th time)
✓ Execute auto-roll (514th time)
✓ Execute auto-roll (515th time)
✓ Execute auto-roll (516th time)
✓ Execute auto-roll (517th time)
✓ Execute auto-roll (518th time)
✓ Execute auto-roll (519th time)
✓ Execute auto-roll (520th time)
✓ Execute auto-roll (521st time)
✓ Execute auto-roll (522nd time)
✓ Execute auto-roll (523rd time)
✓ Execute auto-roll (524th time)
✓ Execute auto-roll (525th time)
✓ Execute auto-roll (526th time)
✓ Execute auto-roll (527th time)
✓ Execute auto-roll (528th time)
✓ Execute auto-roll (529th time)
✓ Execute auto-roll (530th time)
✓ Execute auto-roll (531st time)
✓ Execute auto-roll (532nd time)
✓ Execute auto-roll (533rd time)
✓ Execute auto-roll (534th time)
✓ Execute auto-roll (535th time)
✓ Execute auto-roll (536th time)
✓ Execute auto-roll (537th time)
✓ Execute auto-roll (538th time)
✓ Execute auto-roll (539th time)
✓ Execute auto-roll (540th time)
✓ Execute auto-roll (541st time)
✓ Execute auto-roll (542nd time)
✓ Execute auto-roll (543rd time)
✓ Execute auto-roll (544th time)
✓ Execute auto-roll (545th time)
✓ Execute auto-roll (546th time)
✓ Execute auto-roll (547th time)
✓ Execute auto-roll (548th time)
✓ Execute auto-roll (549th time)
✓ Execute auto-roll (550th time)
✓ Execute auto-roll (551st time)
✓ Execute auto-roll (552nd time)
✓ Execute auto-roll (553rd time)
✓ Execute auto-roll (554th time)
✓ Execute auto-roll (555th time)
✓ Execute auto-roll (556th time)
✓ Execute auto-roll (557th time)
✓ Execute auto-roll (558th time)
✓ Execute auto-roll (559th time)
✓ Execute auto-roll (560th time)
✓ Execute auto-roll (561st time)
✓ Execute auto-roll (562nd time)
✓ Execute auto-roll (563rd time)
✓ Execute auto-roll (564th time)
✓ Execute auto-roll (565th time)
✓ Execute auto-roll (566th time)
✓ Execute auto-roll (567th time)
✓ Execute auto-roll (568th time)
✓ Execute auto-roll (569th time)
✓ Execute auto-roll (570th time)
✓ Execute auto-roll (571st time)
✓ Execute auto-roll (572nd time)
✓ Execute auto-roll (573rd time)
✓ Execute auto-roll (574th time)
✓ Execute auto-roll (575th time)
✓ Execute auto-roll (576th time)
✓ Execute auto-roll (577th time)
✓ Execute auto-roll (578th time)
✓ Execute auto-roll (579th time)
✓ Execute auto-roll (580th time)
✓ Execute auto-roll (581st time)
✓ Execute auto-roll (582nd time)
✓ Execute auto-roll (583rd time)
✓ Execute auto-roll (584th time)
✓ Execute auto-roll (585th time)
✓ Execute auto-roll (586th time)
✓ Execute auto-roll (587th time)
✓ Execute auto-roll (588th time)
✓ Execute auto-roll (589th time)
✓ Execute auto-roll (590th time)
✓ Execute auto-roll (591st time)
✓ Execute auto-roll (592nd time)
✓ Execute auto-roll (593rd time)
✓ Execute auto-roll (594th time)
✓ Execute auto-roll (595th time)
✓ Execute auto-roll (596th time)
✓ Execute auto-roll (597th time)
✓ Execute auto-roll (598th time)
✓ Execute auto-roll (599th time)
✓ Execute auto-roll (600th time)
✓ Execute auto-roll (601st time)
✓ Execute auto-roll (602nd time)
✓ Execute auto-roll (603rd time)
✓ Execute auto-roll (604th time)
✓ Execute auto-roll (605th time)
✓ Execute auto-roll (606th time)
✓ Execute auto-roll (607th time)
✓ Execute auto-roll (608th time)
✓ Execute auto-roll (609th time)
✓ Execute auto-roll (610th time)
✓ Execute auto-roll (611th time)
✓ Execute auto-roll (612th time)
✓ Execute auto-roll (613th time)
✓ Execute auto-roll (614th time)
✓ Execute auto-roll (615th time)
✓ Execute auto-roll (616th time)
✓ Execute auto-roll (617th time)
✓ Execute auto-roll (618th time)
✓ Execute auto-roll (619th time)
✓ Execute auto-roll (620th time)
✓ Execute auto-roll (621st time)
✓ Execute auto-roll (622nd time)
✓ Execute auto-roll (623rd time)
✓ Execute auto-roll (624th time)
✓ Execute auto-roll (625th time)
✓ Execute auto-roll (626th time)
✓ Execute auto-roll (627th time)
✓ Execute auto-roll (628th time)
✓ Execute auto-roll (629th time)
✓ Execute auto-roll (630th time)
✓ Execute auto-roll (631st time)
✓ Execute auto-roll (632nd time)
✓ Execute auto-roll (633rd time)
✓ Execute auto-roll (634th time)
✓ Execute auto-roll (635th time)
✓ Execute auto-roll (636th time)
✓ Execute auto-roll (637th time)
✓ Execute auto-roll (638th time)
✓ Execute auto-roll (639th time)
✓ Execute auto-roll (640th time)
✓ Execute auto-roll (641st time)
✓ Execute auto-roll (642nd time)
✓ Execute auto-roll (643rd time)
✓ Execute auto-roll (644th time)
✓ Execute auto-roll (645th time)
✓ Execute auto-roll (646th time)
✓ Execute auto-roll (647th time)
✓ Execute auto-roll (648th time)
✓ Execute auto-roll (649th time)
✓ Execute auto-roll (650th time)
✓ Execute auto-roll (651st time)
✓ Execute auto-roll (652nd time)
✓ Execute auto-roll (653rd time)
✓ Execute auto-roll (654th time)
✓ Execute auto-roll (655th time)
✓ Execute auto-roll (656th time)
✓ Execute auto-roll (657th time)
✓ Execute auto-roll (658th time)
✓ Execute auto-roll (659th time)
✓ Execute auto-roll (660th time)
✓ Execute auto-roll (661st time)
✓ Execute auto-roll (662nd time)
✓ Execute auto-roll (663rd time)
✓ Execute auto-roll (664th time)
✓ Execute auto-roll (665th time)
✓ Execute auto-roll (666th time)
✓ Execute auto-roll (667th time)
✓ Execute auto-roll (668th time)
✓ Execute auto-roll (669th time)
✓ Execute auto-roll (670th time)
✓ Execute auto-roll (671st time)
✓ Execute auto-roll (672nd time)
✓ Execute auto-roll (673rd time)
✓ Execute auto-roll (674th time)
✓ Execute auto-roll (675th time)
✓ Execute auto-roll (676th time)
✓ Execute auto-roll (677th time)
✓ Execute auto-roll (678th time)
✓ Execute auto-roll (679th time)
✓ Execute auto-roll (680th time)
✓ Execute auto-roll (681st time)
✓ Execute auto-roll (682nd time)
✓ Execute auto-roll (683rd time)
✓ Execute auto-roll (684th time)
✓ Execute auto-roll (685th time)
✓ Execute auto-roll (686th time)
✓ Execute auto-roll (687th time)
✓ Execute auto-roll (688th time)
✓ Execute auto-roll (689th time)
✓ Execute auto-roll (690th time)
✓ Execute auto-roll (691st time)
✓ Execute auto-roll (692nd time)
✓ Execute auto-roll (693rd time)
✓ Execute auto-roll (694th time)
✓ Execute auto-roll (695th time)
✓ Execute auto-roll (696th time)
✓ Execute auto-roll (697th time)
✓ Execute auto-roll (698th time)
✓ Execute auto-roll (699th time)
✓ Execute auto-roll (700th time)
✓ Execute auto-roll (701st time)
✓ Execute auto-roll (702nd time)
✓ Execute auto-roll (703rd time)
✓ Execute auto-roll (704th time)
✓ Execute auto-roll (705th time)
✓ Execute auto-roll (706th time)
✓ Execute auto-roll (707th time)
✓ Execute auto-roll (708th time)
✓ Execute auto-roll (709th time)
✓ Execute auto-roll (710th time)
✓ Execute auto-roll (711th time)
✓ Execute auto-roll (712th time)
✓ Execute auto-roll (713th time)
✓ Execute auto-roll (714th time)
✓ Execute auto-roll (715th time)
✓ Execute auto-roll (716th time)
✓ Execute auto-roll (717th time)
✓ Execute auto-roll (718th time)
✓ Execute auto-roll (719th time)
✓ Execute auto-roll (720th time)
✓ Execute auto-roll (721st time)
✓ Execute auto-roll (722nd time)
✓ Execute auto-roll (723rd time)
✓ Execute auto-roll (724th time)
✓ Execute auto-roll (725th time)
✓ Execute auto-roll (726th time)
✓ Execute auto-roll (727th time)
✓ Execute auto-roll (728th time)
✓ Execute auto-roll (729th time)
✓ Execute auto-roll (730th time)
✓ Execute auto-roll (731st time)
✓ Execute auto-roll (732nd time)
✓ Execute auto-roll (733rd time)
✓ Execute auto-roll (734th time)
✓ Execute auto-roll (735th time)
✓ Execute auto-roll (736th time)
✓ Execute auto-roll (737th time)
✓ Execute auto-roll (738th time)
✓ Execute auto-roll (739th time)
✓ Execute auto-roll (740th time)
✓ Execute auto-roll (741st time)
✓ Execute auto-roll (742nd time)
✓ Execute auto-roll (743rd time)
✓ Execute auto-roll (744th time)
✓ Execute auto-roll (745th time)
✓ Execute auto-roll (746th time)
✓ Execute auto-roll (747th time)
✓ Execute auto-roll (748th time)
✓ Execute auto-roll (749th time)
✓ Execute auto-roll (750th time)
✓ Execute auto-roll (751st time)
✓ Execute auto-roll (752nd time)
✓ Execute auto-roll (753rd time)
✓ Execute auto-roll (754th time)
✓ Execute auto-roll (755th time)
✓ Execute auto-roll (756th time)
✓ Execute auto-roll (757th time)
✓ Execute auto-roll (758th time)
✓ Execute auto-roll (759th time)
✓ Execute auto-roll (760th time)
✓ Execute auto-roll (761st time)
✓ Execute auto-roll (762nd time)
✓ Execute auto-roll (763rd time)
✓ Execute auto-roll (764th time)
```

```
        ✓ Execute auto-roll (765th time)
        ✓ Execute auto-roll (766th time)
        ✓ Execute auto-roll (767th time)
        ✓ Execute auto-roll (768th time)
        ✓ Execute auto-roll (769th time)
        ✓ Execute auto-roll (770th time)
        ✓ Execute auto-roll (771st time)
        ✓ Execute auto-roll (772nd time)
        ✓ Execute auto-roll (773rd time)
        ✓ Execute auto-roll (774th time)
        ✓ Execute auto-roll (775th time)
        ✓ Execute auto-roll (776th time)
        ✓ Execute auto-roll (777th time)
        ✓ Execute auto-roll (778th time)
        ✓ Execute auto-roll (779th time)
        ✓ Execute auto-roll (780th time)
        ✓ Execute auto-roll (781st time)
        ✓ Execute auto-roll (782nd time)
        ✓ Execute auto-roll (783rd time)
        ✓ Execute auto-roll (784th time)
        ✓ Execute auto-roll (785th time)
        ✓ Execute auto-roll (786th time)
        ✓ Execute auto-roll (787th time)
        ✓ Execute auto-roll (788th time)
        ✓ Execute auto-roll (789th time)
        ✓ Execute auto-roll (790th time)
        ✓ Execute auto-roll (791st time)
        ✓ Execute auto-roll (792nd time)
        ✓ Execute auto-roll (793rd time)
        ✓ Execute auto-roll (794th time)
        ✓ Execute auto-roll (795th time)
        ✓ Execute auto-roll (796th time)
        ✓ Execute auto-roll (797th time)
        ✓ Execute auto-roll (798th time)
        ✓ Execute auto-roll (799th time)
        ✓ Execute auto-roll (800th time)
        ✓ Fill an order

    Performance Test: Order Book
      Fill orders without the order cleaning
        ETH market
          ✓ 1 orders
          ✓ 10 orders
          ✓ 100 orders
        USDC market
          ✓ 1 orders
          ✓ 10 orders
          ✓ 100 orders
        Show results
┌─────────┬──────────────┬───────────────┐
│ (index) │ GasCosts(ETH) │ GasCosts(USDC) │
├─────────┼──────────────┼───────────────┤
│    1    │    850530    │     945474    │
│   10    │    544793    │     672717    │
│   100   │    1492448   │    1618519    │
└─────────┴──────────────┴───────────────┘
          ✓ Gas Costs
      Place an order with the order cleaning
        USDC market
          ✓ 1 markets
          ✓ 2 markets
          ✓ 8 markets
        Show results
┌─────────┬──────────┐
│ (index) │ GasCosts │
├─────────┼──────────┤
│    1    │  533081  │
│    2    │  710113  │
│    8    │  875126  │
└─────────┴──────────┘
          ✓ Gas Costs
      Place an order with active orders
        USDC market
          ✓ Deposit
          ✓ Active orders: 0
          ✓ Active orders: 1
          ✓ Active orders: 2
          ✓ Active orders: 3
          ✓ Active orders: 4
          ✓ Active orders: 5
          ✓ Active orders: 6
          ✓ Active orders: 7
          ✓ Active orders: 8
          ✓ Active orders: 9
          ✓ Active orders: 10
          ✓ Active orders: 11
          ✓ Active orders: 12
          ✓ Active orders: 13
          ✓ Active orders: 14
          ✓ Active orders: 15
          ✓ Active orders: 16
          ✓ Active orders: 17
          ✓ Active orders: 18
          ✓ Active orders: 19
        Show results
┌─────────┬────────────────┐
│ (index) │ GasCosts(USDC) │
├─────────┼────────────────┤
│    0    │     787930     │
│    1    │     624829     │
│    2    │     716248     │
│    3    │     683599     │
│    4    │     746064     │
│    5    │     753872     │
│    6    │     801412     │
│    7    │     836635     │
│    8    │     851557     │
│    9    │     852595     │
│   10    │     926702     │
│   11    │     927478     │
│   12    │     963843     │
│   13    │     966771     │
│   14    │    1027822     │
│   15    │    1063618     │
│   16    │    1088277     │
│   17    │    1135624     │
│   18    │    1139250     │
│   19    │    1154338     │
└─────────┴────────────────┘
          ✓ Gas Costs

    AddressResolver
      Ownership
        ✓ Transfer ownership
        ✓ Renounce ownership
        ✓ Fail to renounce ownership due to execution by non-owner
        ✓ Fail to transfer ownership due execution by non-owner
        ✓ Fail to transfer ownership due to zero address
      Initialization
        ✓ Fail to call initialization due to duplicate execution
        ✓ Fail to call initialization due to execution by non-proxy contract
      Address importing
        ✓ Import empty array
        ✓ Import an address
        ✓ Import multiple addresses
        ✓ Import an addresses multiple times with different contract
        ✓ Fail to import an addresses due to unmatched inputs
        ✓ Fail to import an addresses due to execution by non-owner
      Imported address check
        ✓ Get an empty address
        ✓ Get a imported address
        ✓ Get multiple imported addresses
        ✓ Fail to get a imported address due to non-exist contract
```

```
BeaconProxyController
  Initialization
    ✓ Check if the contract addresses are cached in the resolver
    ✓ Fail to call initialization due to duplicate execution
    ✓ Fail to call initialization due to execution by non-proxy contract
  Get data
    ✓ Get the required contracts
    ✓ Fail to get the beacon proxy address due to empty address
  FutureValueVault implementation
    ✓ Set an implementation contract and deploy the contract
    ✓ Set an implementation contract twice
    ✓ Fail to set an implementation contract due to execution by non-owner
    ✓ Fail to deploy the contract due to execution by non-accepted contract
    ✓ Fail to deploy the contract due to non-existence of beacon proxy contract
  LendingMarket implementation
    ✓ Set an implementation contract and deploy the contract
    ✓ Set an implementation contract twice
    ✓ Fail to set an implementation contract due to execution by non-owner
    ✓ Fail to deploy the contract due to execution by non-accepted contract
    ✓ Fail to deploy the contract due to non-existence of beacon proxy contract
  Change Admin
    ✓ Successfully change admins of a beacon proxy contract
    ✓ Fail to change admins of a beacon proxy contract due to execution by non-owner

CurrencyController
  Initialization
    ✓ Add a currency except for ETH as a supported currency
    ✓ Fail to add a currency due to the invalid price
    ✓ Fail to add a currency due to the invalid decimals
    ✓ Fail to add a currency due to the invalid price feed
    ✓ Fail to add a currency due to execution by non-owner
    ✓ Fail to call initialization due to duplicate execution
    ✓ Fail to call initialization due to execution by non-proxy contract
  Update
    ✓ Update a currency support
    ✓ Update a haircut
    ✓ Update a price feed
    ✓ Update multiple price feeds
    ✓ Update multiple data using multicall
    ✓ Fail to update the haircut due to overflow
    ✓ Fail to update the haircut due to invalid currency
    ✓ Fail to update the price feed due to invalid currency
    ✓ Fail to remove the currency due to execution by non-owner
    ✓ Fail to update the haircut due to execution by non-owner
    ✓ Fail to update the price feed due to execution by non-owner
    ✓ Fail to update the price feed due to stale price feed
  Convert
    ✓ Get the converted amount(int256) in the base currency
    ✓ Get the converted amount(uint256) in the base currency
    ✓ Get the array of converted amounts(uint256[]) in the base currency
    ✓ Get the converted amount(uint256) in the selected currency
    ✓ Get the converted amount(uint256) in the selected currency
    ✓ Get the converted amount in the selected currency from another selected currency
    ✓ Get the converted amounts in the selected currency from another selected currency
    ✓ Get the converted amounts in the selected currency from another selected currency from 0
    ✓ Fail to get the converted amount due to stale price feed

FutureValueVault
  Initialization
    ✓ Fail to call initialization due to duplicate execution
    ✓ Fail to call initialization due to execution by non-beacon proxy contract
  Update balance
    ✓ Increase user balance
    ✓ Decrease user balance
    ✓ Fail to increase balance due to execution by non-accepted contract
    ✓ Fail to decrease balance due to execution by non-accepted contract
    ✓ Fail to increase balance due to invalid user address
    ✓ Fail to decrease balance due to invalid user address
  Transfer balance
    ✓ Transfer balance to another user
    ✓ Fail to transfer balance because sender has balance in the past maturity
    ✓ Fail to transfer balance because receiver has balance in the past maturity
    ✓ Fail to transfer balance due to execution by non-accepted contract
  Reset balance
    ✓ Force reset a user's empty balance with amount
    ✓ Force reset a user's balance with amount
    ✓ Force reset a user's empty balance without amount
    ✓ Force reset a user's balance with amount
    ✓ Fail to force reset a user's balance due to lending amount mismatch
    ✓ Fail to force reset a user's balance due to borrowing amount mismatch
    ✓ Fail to force reset a user's balance with amount due to execution by non-accepted contract
    ✓ Fail to force reset a user's balance without amount due to execution by non-accepted contract

GenesisValueVault
  Initialize
    ✓ Initialize contract settings
    ✓ Fail to call initialization due to duplicate execution
    ✓ Fail to call initialization due to execution by non-proxy contract
    ✓ Fail to initialize the currency setting due to execution by non-accepted contract
    ✓ Fail to update the initial compound factor due to execution by non-accepted contract
    ✓ Fail to initialize the currency setting due to zero compound factor
    ✓ Fail to initialize the currency setting due to the initialized
    ✓ Fail to update the initial compound factor due to the finalized
  Balance
    Calculate balance
      ✓ Convert balance to selected maturity from another maturity
      ✓ Convert 0 to selected maturity from another maturity
      ✓ Convert balance to selected maturity from same maturity
      ✓ Calculate amount in FV from positive amount in GV
      ✓ Calculate amount in FV from negative amount in GV
      ✓ Fail to calculate amount in FV from amount in GV due to no compound factor
      ✓ Fail to calculate amount in GV from amount in FV due to no compound factor
    Update balance
      ✓ Update the genesis value
      ✓ Update the genesis value after auto-rolls
      ✓ Update the genesis value with residual amount
      ✓ Update the genesis value from a positive amount to a negative amount
      ✓ Update the genesis value from a negative amount to a positive amount
      ✓ Fail to update the genesis value due to execution by non-accepted contract
      ✓ Fail to update the genesis value with residual amount due to execution by non-accepted contract
    Transfer balance
      ✓ Transfer balance to another user
      ✓ Fail to transfer balance to another user due to execution by non-accepted contract
    Reset balance
      ✓ Force reset a user's empty balance with amount
      ✓ Force reset a user's empty balance with small amount
      ✓ Force reset a user's empty balance without amount
      ✓ Fail to force reset a user's balance due to lending amount mismatch
      ✓ Fail to force reset a user's balance due to borrowing amount mismatch
      ✓ Fail to force reset a user's balance without amount due to execution by non-accepted contract
      ✓ Fail to force reset a user's balance with amount due to execution by non-accepted contract
    Clean up balance
      ✓ Clean up a user balance
      ✓ Clean up a user balance with fluctuation
      ✓ Fail to clean up a user balance due to execution by non-accepted contract
  Auto-roll
    ✓ Execute auto-roll
    ✓ Calculate the balance fluctuation of auto-rolls by on the current maturity
    ✓ Calculate the balance fluctuation of auto-rolls by on the future maturity
    ✓ Calculate the balance fluctuation of auto-rolls by on the past maturity
    ✓ Calculate the balance fluctuation of auto-rolls with invalid maturity
    ✓ Fail to execute auto-roll due to duplicate execution
    ✓ Fail to execute auto-roll due to execution by non-accepted contract
    ✓ Fail to execute auto-roll due to invalid order fee rate
    ✓ Fail to execute auto-roll due to zero unit price
    ✓ Fail to execute auto-roll due to invalid maturity

LendingMarket - Auto-rolls
  ✓ Execute an auto-roll
  ✓ Fail to execute an auto-roll due to invalid caller
  ✓ Fail to execute an auto-roll due to invalid caller

LendingMarket - Calculations
  ✓ Calculate the filled amount from one lending order
```

```
    ✓ Calculate the filled amount from one borrowing order
    ✓ Calculate the filled amount from multiple lending order
    ✓ Calculate the filled amount from multiple borrowing order
    ✓ Calculate the blocked order amount by the circuit breaker

  LendingMarket - Circuit Breakers
    Get circuit breaker thresholds
      ✓ Get circuit breaker thresholds without the last block price
      ✓ Get circuit breaker thresholds with the last block price
    Borrow orders
      ✓ Fill an order partially until the circuit breaker threshold using the market order
      ✓ Fill an order partially until the circuit breaker threshold using the limit order
      ✓ Execute multiple transactions to fill orders in one block with the circuit breaker triggered
      ✓ Fill an order in different blocks after the circuit breaker has been triggered
      ✓ Fill an order in the same block after the circuit breaker has been triggered
      ✓ Fail to place a second market order in the same block due to no filled amount
      ✓ Fail to place a second limit order in the same block due to over the circuit breaker threshold
      ✓ Fill an order within the circuit breaker minimum rage
      ✓ Fill an order outside the circuit breaker minimum rage
      ✓ Fill an order within the circuit breaker that has reached min unit price
    Lend orders
      ✓ Fill an order partially until the circuit breaker threshold using the market order
      ✓ Fill an order partially until the circuit breaker threshold using the limit order
      ✓ Execute multiple transactions to fill orders in one block with the circuit breaker triggered
      ✓ Fill an order in different blocks after the circuit breaker has been triggered
      ✓ Fill an order in the same block after the circuit breaker has been triggered
      ✓ Fail to place a second market order in the same block due to no filled amount
      ✓ Fail to place a second limit order in the same block due to over the circuit breaker threshold
      ✓ Fill an order within the circuit breaker minimum rage
      ✓ Fill an order outside the circuit breaker minimum rage
      ✓ Fill an order within the circuit breaker that has reached max unit price
    Unwind positions
      ✓ Unwind a position partially until the circuit breaker threshold
      ✓ Unwind no position due to circuit breaker

  LendingMarket - Initialization
    ✓ Deploy Lending Market
    ✓ Create an order book
    ✓ Fail to deploy Lending Market with circuit breaker range more than equal to 10000
    ✓ Fail to create an order book due to invalid caller

  LendingMarket - Itayose
    ✓ Execute Itayose call(Case 1)
    ✓ Execute Itayose call(Case 2)
    ✓ Execute Itayose call(Case 3)
    ✓ Execute Itayose call(Case 4)
    ✓ Execute Itayose call(Case 5)
    ✓ Execute Itayose call(Case 6)
    ✓ Execute Itayose call(Case 7)
    ✓ Execute Itayose call(Case 8)
    ✓ Execute Itayose call without pre-orders
    ✓ Fail to create a pre-order due to an existing order with a past maturity
    ✓ Fail to create a pre-order due to not in the pre-order period
    ✓ Fail to cancel a pre-order due to in the Itayose period
    ✓ Fail to execute the Itayose call due to not in the Itayose period
    ✓ Fail to execute the Itayose call due to invalid caller

  LendingMarket - Operations
    ✓ Pause and unpause the lending market
    ✓ Fail to update the order fee rate due to invalid caller
    ✓ Fail to update the circuit breaker limit range due to invalid caller

  LendingMarket - Orders
    Check the block unit price
      ✓ Check with a single order
      ✓ Check with multiple orders in the same block
      ✓ Check with multiple orders in the different block
      ✓ Check with 5 orders in the different block
      ✓ Check with over 5 orders in the different block
      ✓ Check with unwinding
      ✓ Check with an order less than the reliable amount
      ✓ Check with an order equal to the reliable amount
    Execute orders
      ✓ Fail to create a order due to the matured order book
      ✓ Fail to unwind the position due to the matured order book
      ✓ Fail to create an order due to invalid caller
      ✓ Fail to cancel the order due to invalid caller
      ✓ Fail to unwind the position due to invalid caller
    Execute pre-orders
      ✓ Fail to create a lending pre-order due to opposite order existing
      ✓ Fail to create a borrowing pre-order due to opposite order existing
      ✓ Fail to create a pre-order due to invalid caller
    Clean up orders
      ✓ Clean up a lending order
      ✓ Clean up a borrowing order
      ✓ Fail to clean up orders due to invalid caller

  LendingMarketController - Calculations
    Total Funds Calculations
      ✓ Calculate total funds without positions
      ✓ Calculate total funds with positions
      ✓ Calculate total funds with additional lent amount exceeded deposit amount
    Order Estimations
      ✓ Get an borrowing order estimation from one lending order on the order book
      ✓ Get an lending order estimation from one borrowing order on the order book
      ✓ Get an order estimation from multiple order on the order book
      ✓ Get an order estimation blocked by the circuit breaker
      ✓ Get an borrowing order estimation on the empty order book
      ✓ Get an lending order estimation on the empty order book

  LendingMarketController - Itayose
    ✓ Get Itayose estimation with no pre-orders
    ✓ Execute Itayose call on the initial markets, the opening price become the same as the lending order
    ✓ Execute Itayose call on the initial markets, the opening price become the same as the borrowing
order
    ✓ Fill a borrowing pre-order whose unit price is lower than the opening price after Itayose call
    ✓ Fill a lending pre-order whose unit price is higher than the opening price after Itayose call.
    ✓ Execute Itayose call after auto-rolling
    ✓ Fill orders that are not filled with Itayose call and not the same as the opening unit price
    ✓ Fill orders that are not filled with Itayose call and the same as the opening unit price
    ✓ Filled pre-order should be returned as inactive orders with opening unit price
    ✓ Fail to create an order due to too many orders
    ✓ Fail to create an pre-order due to invalid maturity
    ✓ Fail to create an pre-order and deposit token due to invalid maturity

  LendingMarketController - Liquidations
    External liquidator
      ✓ Fail to execute liquidation call due to non-operator
      ✓ Fail to execute liquidation call due to invalid maturity
      ✓ Fail to execute liquidation call due to non-collateral currency selected
      ✓ Fail to execute forced repayment due to non-operator
      ✓ Fail to execute forced repayment due to invalid maturity
      ✓ Fail to execute forced repayment due to non-collateral currency selected
      ✓ Fail to execute operations for collateral due to non lending market controller
      ✓ Fail to execute operations for debt due to non lending market controller
    Liquidations
      ✓ Liquidate less than 50% borrowing position in case the one position doesn't cover liquidation
amount
      ✓ Liquidate 50% borrowing position in case the one position cover liquidation amount
      ✓ Liquidate borrowing position using zero-coupon bonds
      ✓ Liquidate insolvent user using the reserve fund
      ✓ Liquidate insolvent user without using the reserve fund
      ✓ Liquidate borrowing position after auto-roll
      ✓ Fail to liquidate a borrowing position due to no debt
      ✓ Fail to liquidate a borrowing position due to no liquidation amount
      ✓ Fail to liquidate a borrowing position due to insufficient collateral
    Delisting
      ✓ Execute repayment & redemption
      ✓ Execute repayment & redemption after auto-roll
      ✓ Force repayment of overdue borrowing positions
      ✓ Force a insolvent user to repay
      ✓ Force a insolvent user to repay after auto roll
      ✓ Fail to repay due to active market
      ✓ Fail to repay due to active currency
      ✓ Fail to redeem due to active market
      ✓ Fail to redeem due to under repayment period
      ✓ Fail to repay due to invalid maturity
      ✓ Fail to redeem due to invalid maturity

  LendingMarketController - Operations
    Operations
      Order books
        ✓ Get the lending market detail with empty order book
        ✓ Get the lending market detail with non-empty order book
        ✓ Get the multiple lending market details
      Pause/Unpause
        ✓ Pause lending markets
        ✓ Change the operator
      Protocol updates
        ✓ Get the min debt unit price
        ✓ Update the min debt unit price
        ✓ Get the current min debt unit price
        ✓ Update the order fee rate
        ✓ Update the circuit breaker limit range
        ✓ Update beacon proxy implementations and calculate Genesis value
        ✓ Fail to update the order fee rate due to execution by non-owner
        ✓ Fail to update the circuit breaker limit range due to execution by non-owner
        ✓ Fail to update the min debt unit price due to execution by non-owner
        ✓ Fail to update the min debt unit price due to invalid value
      Calculation
┌──────────────┬───────┬─────┬───────┬─────────────┬────────────────────┬──────────────────────┐
│   (index)    │ Alice │ Bob │ Carol │ ReserveFund │ TotalLendingSupply │ TotalBorrowingSupply │
├──────────────┼───────┼─────┼───────┼─────────────┼────────────────────┼──────────────────────┤
│ GenesisValue │  '0'  │ '0' │  '0'  │     '0'     │        '0'         │         '0'          │
└──────────────┴───────┴─────┴───────┴─────────────┴────────────────────┴──────────────────────┘
┌──────────────┬────────────────────────────────────────┬────────────────────────────────────────┬──────
─┬──────────────────────────────────────┬──────────────────────────────────────┬──────────────────────┐
│   (index)    │                 Alice                  │                  Bob                   │ Carol
│           ReserveFund                │          TotalLendingSupply          │ TotalBorrowingSupply │
├──────────────┼────────────────────────────────────────┼────────────────────────────────────────┼──────
─┼──────────────────────────────────────┼──────────────────────────────────────┼──────────────────────┤
│ GenesisValue │ '125000000000000000000000000000000000' │ '-125938045183079147431428308349654885' │  '0'
│ '31164185375443900000000000000000' │ '31164185375443900000000000000000' │         '0'          │
└──────────────┴────────────────────────────────────────┴────────────────────────────────────────┴──────
─┴──────────────────────────────────────┴──────────────────────────────────────┴──────────────────────┘
┌──────────────┬────────────────────────────────────────┬────────────────────────────────────────┬──────
─┬──────────────────────────────────────┬────────────────────────────────────────┬────────────────────
────────────────┐
│   (index)    │                 Alice                  │                  Bob                   │ Carol
│           ReserveFund                │          TotalLendingSupply            │
TotalBorrowingSupply          │
├──────────────┼────────────────────────────────────────┼────────────────────────────────────────┼──────
─┼──────────────────────────────────────┼────────────────────────────────────────┼────────────────────
────────────────┤
│ GenesisValue │ '125000000000000000000000000000000000' │ '-125938045183079147431428308349654885' │  '0'
│ '938045183079147431428308349654885' │ '125938045183079147431428308349654885' │
'125938045183079147431428308349654885' │
└──────────────┴────────────────────────────────────────┴────────────────────────────────────────┴──────
─┴──────────────────────────────────────┴────────────────────────────────────────┴────────────────────
────────────────┘
┌──────────────┬────────────────────────────────────────┬────────────────────────────────────────┬──────
─┬──────────────────────────────────────┬────────────────────────────────────────┬────────────────────
────────────────┐
│   (index)    │                 Alice                  │                  Bob                   │ Carol
│           ReserveFund                │          TotalLendingSupply            │
TotalBorrowingSupply          │
├──────────────┼────────────────────────────────────────┼────────────────────────────────────────┼──────
─┼──────────────────────────────────────┼────────────────────────────────────────┼────────────────────
────────────────┤
│ GenesisValue │ '125000000000000000000000000000000000' │ '-125938045183079147431428308349654885' │  '0'
│ '938045183079147431428308349654885' │ '125938045183079147431428308349654885' │
'125938045183079147431428308349654885' │
└──────────────┴────────────────────────────────────────┴────────────────────────────────────────┴──────
─┴──────────────────────────────────────┴────────────────────────────────────────┴────────────────────
────────────────┘
┌──────────────┬────────────────────────────────────────┬────────────────────────────────────────┬──────
```

```
─┬──────────────────────────────────────────┬──────────────────────────────────────────┬──────
────────────────┐
│   (index)    │               Alice                  │                 Bob                  │ Carol
│           ReserveFund             │           TotalLendingSupply           │
TotalBorrowingSupply          │
├──────────────┼──────────────────────────────────────┼──────────────────────────────────────┼──────
─┼──────────────────────────────────────────┼──────────────────────────────────────────┼──────
────────────────┤
│ GenesisValue │ '12500000000000000000000000000000000' │ '-126567579754842184087485128954433938' │  '0'
│ '938045183079147431428308349654885' │ '125938045183079147431428308349654885' │
'125938045183079147431428308349654885' │
└──────────────┴──────────────────────────────────────┴──────────────────────────────────────┴──────
─┴──────────────────────────────────────────┴──────────────────────────────────────────┴──────
────────────────┘

┌──────────────┬──────────────────────────────────────┬──────────────────────────────────────┬──────
─┬──────────────────────────────────────────┬──────────────────────────────────────────┬──────
─────────────────┐
│   (index)    │               Alice                  │                 Bob                  │ Carol
│           ReserveFund              │           TotalLendingSupply           │
TotalBorrowingSupply          │
├──────────────┼──────────────────────────────────────┼──────────────────────────────────────┼──────
─┼──────────────────────────────────────────┼──────────────────────────────────────────┼──────
─────────────────┤
│ GenesisValue │ '12500000000000000000000000000000000' │ '-127200261221385656143159406050467025' │  '0'
│ '2200261221385656143159406050467025' │ '127200261221385656143159406050467025' │
'127200261221385656143159406050467025' │
└──────────────┴──────────────────────────────────────┴──────────────────────────────────────┴──────
─┴──────────────────────────────────────────┴──────────────────────────────────────────┴──────
─────────────────┘

┌──────────────┬──────────────────────────────────────┬──────────────────────────────────────┬──────
─┬──────────────────────────────────────────┬──────────────────────────────────────────┬──────
─────────────────┐
│   (index)    │               Alice                  │                 Bob                  │ Carol
│           ReserveFund              │           TotalLendingSupply           │
TotalBorrowingSupply          │
├──────────────┼──────────────────────────────────────┼──────────────────────────────────────┼──────
─┼──────────────────────────────────────────┼──────────────────────────────────────────┼──────
─────────────────┤
│ GenesisValue │ '12500000000000000000000000000000000' │ '-127200261221385656143159406050467025' │  '0'
│ '2200261221385656143159406050467025' │ '127200261221385656143159406050467025' │
'127200261221385656143159406050467025' │
└──────────────┴──────────────────────────────────────┴──────────────────────────────────────┴──────
─┴──────────────────────────────────────────┴──────────────────────────────────────────┴──────
─────────────────┘
        ✓ Calculate the genesis value per maturity
        ✓ Calculate the total funds from inactive lending order list
        ✓ Calculate the total funds from inactive borrowing order list
        ✓ Calculate the total funds with open orders less than min debt unit price
        ✓ Calculate the total funds with position less than min debt unit price

  LendingMarketController - Orders
    Initialization
      ✓ Initialize the lending market
      ✓ Fail to initialize the lending market due to invalid currency
      ✓ Fail to initialize the lending market due to execution by non-owner
      ✓ Get genesisDate
      ✓ Get beacon proxy implementations
      ✓ Fail to get beacon proxy implementations
      ✓ Create a order book
┌─────────┬───────────────────────────────┬────────────────────┐
│ (index) │           Maturity            │ Maturity(Unixtime) │
├─────────┼───────────────────────────────┼────────────────────┤
│    0    │   'March 25, 2242 1:00 AM'    │    '8590665600'    │
│    1    │    'June 24, 2242 2:00 AM'    │    '8598528000'    │
│    2    │ 'September 30, 2242 2:00 AM'  │    '8606995200'    │
│    3    │  'December 30, 2242 1:00 AM'  │    '8614857600'    │
│    4    │   'March 31, 2243 2:00 AM'    │    '8622720000'    │
│    5    │    'June 30, 2243 2:00 AM'    │    '8630582400'    │
│    6    │ 'September 29, 2243 2:00 AM'  │    '8638444800'    │
│    7    │  'December 29, 2243 1:00 AM'  │    '8646307200'    │
│    8    │   'March 29, 2244 1:00 AM'    │    '8654169600'    │
└─────────┴───────────────────────────────┴────────────────────┘
```

```
      ✓ Create multiple lending markets
      ✓ Fail to create a order book because market is not initialized
      ✓ Fail to create a order book because currency does not exist
      ✓ Fail to create a order book due to invalid pre-opening date
      ✓ Fail to create a order book due to execution by non-owner
    Orders
      ✓ Get a market currency data
      ✓ Add orders and check rates
┌─────────────────────────┬──────────────────────┬─────┬───────────────────────┬───────────────────┐
│         (index)         │        Alice         │ Bob │         Carol         │    ReserveFund    │
├─────────────────────────┼──────────────────────┼─────┼───────────────────────┼───────────────────┤
│    TotalPresentValue    │ '100000000000000000' │ '0' │ '-100230133751902588' │ '230133751902588' │
│ FutureValue(8590665600) │ '114678899082568807' │ '0' │ '-114942813935668105' │ '263914853099298' │
│ FutureValue(8598528000) │         '0'          │ '0' │          '0'          │        '0'        │
│       GenesisValue      │         '0'          │ '0' │          '0'          │        '0'        │
└─────────────────────────┴──────────────────────┴─────┴───────────────────────┴───────────────────┘
┌─────────────────────────┬──────────────────────┬──────────────────────┬───────────────────────┬────────
───────────┐
│         (index)         │        Alice         │         Bob          │         Carol         │
ReserveFund    │
├─────────────────────────┼──────────────────────┼──────────────────────┼───────────────────────┼────────
───────────┤
│    TotalPresentValue    │ '140000000000000000' │ '-40099726002029427' │ '-100230133751902588' │
'329859753932015' │
│ FutureValue(8590665600) │ '114678899082568807' │         '0'          │ '-114942813935668105' │
'263914853099298' │
│ FutureValue(8598528000) │ '45871559633027523'  │ '-45985924314253930' │          '0'          │
'114364681226407' │
│       GenesisValue      │         '0'          │         '0'          │          '0'          │
'0'        │
└─────────────────────────┴──────────────────────┴──────────────────────┴───────────────────────┴────────
───────────┘
┌─────────────────────────┬───────────────────────────────────────┬──────────────────────┬──────────────
───────────────────────────┬───────────────────────────────────────┐
│         (index)         │                 Alice                 │         Bob          │
Carol                     │              ReserveFund              │
├─────────────────────────┼───────────────────────────────────────┼──────────────────────┼──────────────
───────────────────────────┼───────────────────────────────────────┤
│    TotalPresentValue    │          '154442736772314281'         │ '-40099726002029427' │
'-115205885789457075'         │          '363097365943557'            │
│ FutureValue(8590665600) │                  '0'                  │         '0'          │
'0'                       │                  '0'                  │
│ FutureValue(8598528000) │          '177113230243479680'         │ '-45985924314253930' │
'-132116841501670958'         │          '416396061861877'            │
│       GenesisValue      │ '114678899082568807000000000000000000' │         '0'          │
'-115443622922544748673034195543409371' │ '263914853099298000000000000000000' │
└─────────────────────────┴───────────────────────────────────────┴──────────────────────┴──────────────
───────────────────────────┴───────────────────────────────────────┘
┌─────────────────────────┬───────────────────────────────────────┬──────────────────────┬──────────────
───────────────────────────┬───────────────────────────────────────┐
│         (index)         │                 Alice                 │         Bob          │
Carol                     │              ReserveFund              │
├─────────────────────────┼───────────────────────────────────────┼──────────────────────┼──────────────
───────────────────────────┼───────────────────────────────────────┤
│    TotalPresentValue    │          '154442736772314281'         │ '-40099726002029427' │
'-115205885789457075'         │          '363097365943557'            │
│ FutureValue(8590665600) │                  '0'                  │         '0'          │
'0'                       │                  '0'                  │
│ FutureValue(8598528000) │          '177113230243479680'         │ '-45985924314253930' │
'-132116841501670958'         │          '416396061861877'            │
│       GenesisValue      │ '114678899082568807000000000000000000' │         '0'          │
'-115443622922544748673034195543409371' │ '263914853099298000000000000000000' │
└─────────────────────────┴───────────────────────────────────────┴──────────────────────┴──────────────
───────────────────────────┴───────────────────────────────────────┘
┌─────────────────────────┬───────────────────────────────────────┬──────────────────────┬──────────────
───────────────────────────┬───────────────────────────────────────┐
│         (index)         │                 Alice                 │         Bob          │
Carol                     │              ReserveFund              │
├─────────────────────────┼───────────────────────────────────────┼──────────────────────┼──────────────
───────────────────────────┼───────────────────────────────────────┤
│    TotalPresentValue    │          '154442736772314281'         │ '-40099726002029427' │
'-115205885789457075'         │          '862875019172221'            │
```

```
│ FutureValue(8590665600) │                '0'                    │        '0'          │
'0'                     │                '0'                    │
│ FutureValue(8598528000) │        '177113230243479680'           │ '-45985924314253930' │
'-132116841501670958'             │          '989535572445208'            │
│      GenesisValue       │ '114678899082568807000000000000000000' │        '0'          │
'-115443622922544748673034195543409371' │ '764723839975941673034195543409371' │
└─────────────────────────┴───────────────────────────────────────┴─────────────────────┴───────────────
──────────────────────────┴──────────────────────────────────────┘

┌─────────────────────────┬───────────────────────────────────────┬─────────────────────┬───────────────
──────────────────────────┬──────────────────────────────────────┐
│         (index)         │                 Alice                 │         Bob         │
Carol                     │              ReserveFund              │
├─────────────────────────┼───────────────────────────────────────┼─────────────────────┼───────────────
──────────────────────────┼──────────────────────────────────────┤
│    TotalPresentValue    │          '154442736772314281'         │ '-40099726002029427' │
'-115205885789457075'             │          '862875019172221'            │
│ FutureValue(8590665600) │                '0'                    │        '0'          │
'0'                     │                '0'                    │
│ FutureValue(8598528000) │        '177113230243479680'           │ '-45985924314253930' │
'-132116841501670958'             │          '989535572445208'            │
│      GenesisValue       │ '114678899082568807000000000000000000' │        '0'          │
'-115443622922544748673034195543409371' │ '764723839975941673034195543409371' │
└─────────────────────────┴───────────────────────────────────────┴─────────────────────┴───────────────
──────────────────────────┴──────────────────────────────────────┘
      ✓ Add orders and rotate markets
      ✓ Deposit and add an order
      ✓ Deposit and add an order(payable)
      ✓ Add multiple orders using multicall
      ✓ Get an order
      ✓ Cancel an order
      ✓ Get an active order from one market
      ✓ Get active orders from multiple markets
      ✓ Get active orders from multiple currencies
      ✓ Get active orders and inactive orders
      ✓ Get an empty order list
      ✓ Get an active position from one market
      ✓ Get active positions of a user who has both side position
      ✓ Get active positions from multiple markets
      ✓ Get active positions from multiple currencies
      ✓ Get an active position after auto-rolls
      ✓ Get an empty position list of a user who has an open order
      ✓ Get an empty position list of a user who has no open order
┌─────────────────────┬──────────────────────┬──────────────────────┬─────────┬─────────┐
│       (index)       │        Total         │       Market0        │ Market1 │ Market2 │
├─────────────────────┼──────────────────────┼──────────────────────┼─────────┼─────────┤
│ PresentValue(Alice) │ '-50000000000000000' │ '-50000000000000000' │   '0'   │   '0'   │
└─────────────────────┴──────────────────────┴──────────────────────┴─────────┴─────────┘
┌─────────────────────┬───────────────────────┬──────────────────────┬──────────────────────┬─────────┐
│       (index)       │         Total         │       Market0        │       Market1        │ Market2 │
├─────────────────────┼───────────────────────┼──────────────────────┼──────────────────────┼─────────┤
│ PresentValue(Alice) │ '-100000000000000000' │ '-50000000000000000' │ '-50000000000000000' │   '0'   │
└─────────────────────┴───────────────────────┴──────────────────────┴──────────────────────┴─────────┘
┌─────────────────────┬──────────────────────┬──────────────────────┬──────────────────────┬──────────────
────────┐
│       (index)       │        Total         │       Market0        │       Market1        │
Market2        │
├─────────────────────┼──────────────────────┼──────────────────────┼──────────────────────┼──────────────
────────┤
│ PresentValue(Alice) │ '-20000000000000000' │ '-50000000000000000' │ '-50000000000000000' │
'80000000000000000' │
└─────────────────────┴──────────────────────┴──────────────────────┴──────────────────────┴──────────────
────────┘
      ✓ Fill lending orders and check the total present value
      ✓ Calculate the funds of users who have a large lending position and a small borrowing position
      ✓ Calculate the funds of users who have a small lending position and a large borrowing position
      ✓ Fill lending orders and partially fill own order.
      ✓ Fill lending orders including own order
      ✓ Fill borrowing orders including own order
      ✓ Fill lending orders including another user's order for unwinding
      ✓ Fill multiple lending orders without partially filled orders
    Limit Order
      ✓ Fill all lending orders at one rate
```

```
        ✓ Fill all borrowing orders at one rate
        ✓ Fill orders partially at one rate
        ✓ Fill orders at one rate with a partial amount with limit rate
        ✓ Fill orders at one rate with a over amount with limit rate
        ✓ Fill an own order
        ✓ Fill multiple lending order at different rates with limit rate
        ✓ Fill multiple borrowing order at different rates with limit rate
        ✓ Fill multiple lending order at different rates with limit rate
        ✓ Fill multiple borrowing order at different rates with limit rate
        ✓ Fill an order partially out of the orders held
        ✓ Fill multiple orders partially out of the orders held
        ✓ Fill 100 orders in same rate
        ✓ Fill 100 orders in different rate
      Market Order
        ✓ Fail to place a borrow market order
        ✓ Fail to place a lend market order
      Unwinding
        ✓ Unwind a lending order
        ✓ Unwind a borrowing order
        ✓ Unwind a order at the order book that don't has enough orders
        ✓ Unwind a order ta the order book that don't has any orders
        ✓ Fail to execute unwinding due to insufficient collateral
        ✓ Fail to execute unwinding due to no future values user has
        ✓ Fail to execute unwinding due to invalid maturity
      Order Book
        ✓ Get all borrow orders
        ✓ Get all lend orders
        ✓ Get all borrow orders in multiple calls
        ✓ Get all lend orders in multiple calls
        ✓ Get borrow orders starting from a non-existent unit price
        ✓ Get lend orders starting from a non-existent unit price
        ✓ Get borrow orders starting from the minimum unit price
        ✓ Get borrow orders starting from the maximum unit price
      Failure
        ✓ Fail to create an order due to insufficient collateral
        ✓ Fail to create an order due to too many orders
        ✓ Fail to create an order due to invalid maturity
        ✓ Fail to create an order and deposit token due to invalid maturity
        ✓ Fail to cancel an order due to invalid maturity
        ✓ Fail to rotate lending markets due to pre-maturity
        ✓ Fail to cancel an order due to execution by non-maker
        ✓ Fail to cancel an order due to invalid order

  LendingMarketController - Rotations
    General order books
      ✓ Rotate markets multiple times under condition without lending position
┌─────────────────────┬────────────────────────────────────────┬──────────────────────────────────────
─────┐
│       (index)       │                 Alice                  │                 Bob
│
├─────────────────────┼────────────────────────────────────────┼──────────────────────────────────────
─────┤
│ GenesisValue(8654169600) │ '125000000000000000000000000000000000' │
'-1258682359512735806790557891352011173' │
│ GenesisValue(8662032000) │ '-19858475640791108334888479365227212' │
'18180555494858496832214392666778823'  │
│ GenesisValue(8669894400) │ '147789548787143393928693567331037336' │
'-151390837638659039989013351002805532' │
│ GenesisValue(8677756800) │ '-46594946634607274925438208515571129' │
'40851150277504967131987035690056658'  │
└─────────────────────┴────────────────────────────────────────┴──────────────────────────────────────
─────┘
      ✓ Rotate markets multiple times under condition where users have lending positions that are offset
after the auto-rolls every time
      ✓ Rotate markets using the unit price average(only one order) during the observation period
      ✓ Rotate markets using the unit price average(multiple orders) during the observation period
      ✓ Rotate markets using the estimated auto-roll price
      ✓ Rotate markets using the past auto-roll price as an order is filled on dates too old
      ✓ Rotate markets using the past auto-roll price as no orders are filled
      ✓ Rotate markets including one market that has orders adjusted by with the residual amount.
      ✓ Fail to rotate order books due to no currency
      ✓ Fail to rotate order books due to no order book
    Pre-open order books
      ✓ Rotate markets including one market that has pre-orders adjusted by with the residual amount.
      ✓ Rotate markets including one market that has pre-orders partially filled and  adjusted by with
the residual amount.

  LendingMarketController - Terminations
    Terminations
      ✓ Get the termination status
      ✓ Execute an emergency termination without an order
      ✓ Execute an emergency termination with orders of single market
      ✓ Execute an emergency termination with orders of multiple markets
      ✓ Execute an emergency termination with orders after auto-rolls
      ✓ Execute an emergency termination with paused markets
      ✓ Fail to redeem due to a insolvent user
      ✓ Fail to redeem due to 2nd execution
      ✓ Fail to execute the emergency termination due to execution by non-owner
      ✓ Fail to initialize the lending market due to the market being already initialized
      ✓ Fail to execute the emergency settlement due to no markets terminated

  LendingMarketOperationLogic
    Testing calculateNextMaturity()
      ✓ Get the last Friday after 3 months
      ✓ Get the date 1 week later

  OrderStatisticsTree - drop values
    Dropping
      Lending market orders
        Drop nodes from the tree by one action
          1 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 50000000
            ✓ Drop all nodes: Target amount is 100000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          2 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 100000000
            ✓ Drop 1 node: Target amount is 300000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 350000000
            ✓ Drop all nodes: Target amount is 400000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          3 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 300000000
            ✓ Drop 1 node: Target amount is 500000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 600000000
            ✓ Drop 2 nodes: Target amount is 800000000
            ✓ Drop all nodes: Target amount is 900000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          3 nodes with multiple orders in the tree
            ✓ Fill 1 node partially, Remove 4 order with a unfilled amount: Target amount is 350000000
            ✓ Fill 1 node partially, Remove 4 order without a unfilled amount: Target amount is 400000000
            ✓ Drop 1 node: Target amount is 500000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 580000000
            ✓ Drop 1 node, Fill 1 node partially, Remove 2 order with a unfilled amount: Target amount is
620000000
            ✓ Drop 1 node, Fill 1 node partially, Remove 2 order without a unfilled amount: Target amount
is 700000000
            ✓ Drop 2 nodes: Target amount is 800000000
            ✓ Drop 2 nodes, Fill 1 node partially: Target amount is 820000000
            ✓ Drop 2 nodes, Fill 1 node partially, Remove 1 order with a unfilled amount: Target amount
is 830000000
            ✓ Drop 2 nodes, Fill 1 node partially, Remove 1 order without a unfilled amount: Target
amount is 850000000
            ✓ Drop all nodes: Target amount is 900000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          Many nodes in the tree
            ✓ Fill 1 node partially: Target amount is 50000000
            ✓ Drop 1 node: Target amount is 100000000
            ✓ Drop multiple nodes less than the root: Target amount is 400000000
            ✓ Drop multiple nodes less than the root, Fill root node partially: Target amount is
6000000000
            ✓ Drop multiple nodes less than or equal to the root: Target amount is 6600000000
            ✓ Drop multiple nodes across the root: Target amount is 7000000000
            ✓ Drop all nodes: Target amount is 7500000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 10000000000
        Drop nodes from the tree by multiple actions
          1 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 50000000
            ✓ Drop all nodes: Target amount is 100000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          2 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 100000000
            ✓ Drop 1 node: Target amount is 300000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 350000000
            ✓ Drop all nodes: Target amount is 400000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          3 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 300000000
            ✓ Drop 1 node: Target amount is 500000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 600000000
            ✓ Drop 2 nodes: Target amount is 800000000
            ✓ Drop all nodes: Target amount is 900000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          3 nodes with multiple orders in the tree
            ✓ Fill 1 node partially, Remove 4 order with a unfilled amount: Target amount is 350000000
            ✓ Fill 1 node partially, Remove 4 order without a unfilled amount: Target amount is 400000000
            ✓ Drop 1 node: Target amount is 500000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 580000000
            ✓ Drop 1 node, Fill 1 node partially, Remove 2 order with a unfilled amount: Target amount is
620000000
            ✓ Drop 1 node, Fill 1 node partially, Remove 2 order without a unfilled amount: Target amount
is 700000000
            ✓ Drop 2 nodes: Target amount is 800000000
            ✓ Drop 2 nodes, Fill 1 node partially: Target amount is 820000000
            ✓ Drop 2 nodes, Fill 1 node partially, Remove 1 order with a unfilled amount: Target amount
is 830000000
            ✓ Drop 2 nodes, Fill 1 node partially, Remove 1 order without a unfilled amount: Target
amount is 850000000
            ✓ Drop all nodes: Target amount is 900000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          Many nodes in the tree
            ✓ Fill 1 node partially: Target amount is 50000000
            ✓ Drop 1 node: Target amount is 100000000
            ✓ Drop multiple nodes less than the root: Target amount is 400000000
            ✓ Drop multiple nodes less than the root, Fill root node partially: Target amount is
6000000000
            ✓ Drop multiple nodes less than or equal to the root: Target amount is 6600000000
            ✓ Drop multiple nodes across the root: Target amount is 7000000000
            ✓ Drop all nodes: Target amount is 7500000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 10000000000
        Drop nodes from the tree by repeated inserting and dropping
          1 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 50000000
            ✓ Drop all nodes: Target amount is 100000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          2 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 100000000
            ✓ Drop 1 node: Target amount is 300000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 350000000
            ✓ Drop all nodes: Target amount is 400000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          3 nodes in the tree
            ✓ Fill 1 node partially: Target amount is 300000000
            ✓ Drop 1 node: Target amount is 500000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 600000000
            ✓ Drop 2 nodes: Target amount is 800000000
            ✓ Drop all nodes: Target amount is 900000000
            ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
          3 nodes with multiple orders in the tree
            ✓ Fill 1 node partially, Remove 4 order with a unfilled amount: Target amount is 350000000
            ✓ Fill 1 node partially, Remove 4 order without a unfilled amount: Target amount is 400000000
            ✓ Drop 1 node: Target amount is 500000000
            ✓ Drop 1 node, Fill 1 node partially: Target amount is 580000000
            ✓ Drop 1 node, Fill 1 node partially, Remove 2 order with a unfilled amount: Target amount is
620000000
            ✓ Drop 1 node, Fill 1 node partially, Remove 2 order without a unfilled amount: Target amount
is 700000000
```

```
          ✓ Drop 2 nodes: Target amount is 800000000
          ✓ Drop 2 nodes, Fill 1 node partially: Target amount is 820000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 1 order with a unfilled amount: Target amount
is 830000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 1 order without a unfilled amount: Target
amount is 850000000
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        Many nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop multiple nodes less than the root: Target amount is 400000000
          ✓ Drop multiple nodes less than the root, Fill root node partially: Target amount is
6000000000
          ✓ Drop multiple nodes less than or equal to the root: Target amount is 6600000000
          ✓ Drop multiple nodes across the root: Target amount is 7000000000
          ✓ Drop all nodes: Target amount is 7500000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 10000000000
    Lending limit orders
      Drop nodes from the tree
        1 nodes in the tree
          ✓ Drop all nodes: Target amount is 100000000, Limit value 9800
          ✓ Drop all nodes by limitValue: Target amount is 200000000, Limit value 9800
        2 nodes in the tree
          ✓ Drop 1 node: Target amount is 300000000, Limit value 9801
          ✓ Drop 1 node by limitValue: Target amount is 350000000, Limit value 9801
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 350000000, Limit value 9800
          ✓ Drop all nodes: Target amount is 400000000, Limit value 9800
          ✓ Drop all nodes by limitValue: Target amount is 1000000000, Limit value 9800
        3 nodes in the tree
          ✓ Drop 1 node: Target amount is 500000000, Limit value 9802
          ✓ Drop 1 node by limitValue: Target amount is 600000000, Limit value 9802
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 600000000, Limit value 9801
          ✓ Drop 2 nodes: Target amount is 800000000, Limit value 9801
          ✓ Drop 2 nodes by limitValue: Target amount is 900000000, Limit value 9801
          ✓ Drop all nodes: Target amount is 900000000, Limit value 9800
          ✓ Drop all nodes by limitValue: Target amount is 1000000000, Limit value 9800
        3 discontinuous nodes in the tree
          ✓ Drop 1 node: Target amount is 1000000000, Limit value 9803
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 700000000, Limit value 9801
          ✓ Drop 2 node: Target amount is 1000000000, Limit value 9801
          ✓ Drop all nodes: Target amount is 1000000000, Limit value 9799
    Lending unwind orders
      Drop nodes from the tree
        1 nodes in the tree
          ✓ Fill 1 node partially: Unwind future value 125000000
          ✓ Drop all nodes: Unwind future value 250000000
          ✓ Drop all nodes without limits and amounts: Unwind future value 0
          ✓ Drop all nodes by an exceeding amount: Unwind future value 300000000
        2 nodes in the tree
          ✓ Fill 1 node partially: Unwind future value 125000000
          ✓ Drop 1 node: Unwind future value 250000000
          ✓ Drop 1 node, Fill 1 node partially: Unwind future value 350000000
          ✓ Drop all nodes: Unwind future value 1250000000
          ✓ Drop all nodes without limits and amounts: Unwind future value 0
          ✓ Drop all nodes by an exceeding amount: Unwind future value 2000000000
        3 nodes in the tree
          ✓ Fill 1 node partially: Unwind future value 125000000
          ✓ Drop 1 node: Unwind future value 250000000
          ✓ Drop 1 node, Fill 1 node partially: Unwind future value 350000000
          ✓ Drop 2 nodes: Unwind future value 1250000000
          ✓ Drop 2 nodes, Fill 1 node partially: Unwind future value 1350000000
          ✓ Drop all nodes: Unwind future value 2250000000
          ✓ Drop all nodes without limits and amounts: Unwind future value 0
          ✓ Drop all nodes by an exceeding amount: Unwind future value 3000000000
    Borrowing market orders
      Drop nodes from the tree by one action
        1 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop all nodes: Target amount is 100000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        2 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000
          ✓ Drop all nodes: Target amount is 400000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        3 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000
          ✓ Drop 2 nodes: Target amount is 400000000
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        3 nodes with multiple orders in the tree
          ✓ Fill 1 node partially, Remove 1 order with a unfilled amount: Target amount is 25000000
          ✓ Fill 1 node partially, Remove 1 order without a unfilled amount: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 150000000
          ✓ Drop 1 node, Fill 1 node partially, Remove 2 order with a unfilled amount: Target amount is
  280000000
          ✓ Drop 1 node, Fill 1 node partially, Remove 2 order without a unfilled amount: Target amount
  is 300000000
          ✓ Drop 2 nodes: Target amount is 400000000
          ✓ Drop 2 nodes, Fill 1 node partially: Target amount is 450000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 3 order with a unfilled amount: Target amount
  is 650000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 3 order without a unfilled amount: Target
  amount is 700000000
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        Many nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop multiple nodes less than the root: Target amount is 400000000
          ✓ Drop multiple nodes less than the root, Fill root node partially: Target amount is
  1300000000
          ✓ Drop multiple nodes less than or equal to the root: Target amount is 1600000000
          ✓ Drop multiple nodes across the root: Target amount is 3000000000
          ✓ Drop all nodes: Target amount is 7500000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 10000000000
      Drop nodes from the tree by multiple actions
        1 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop all nodes: Target amount is 100000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        2 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000
          ✓ Drop all nodes: Target amount is 400000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        3 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000
          ✓ Drop 2 nodes: Target amount is 400000000
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        3 nodes with multiple orders in the tree
          ✓ Fill 1 node partially, Remove 1 order with a unfilled amount: Target amount is 25000000
          ✓ Fill 1 node partially, Remove 1 order without a unfilled amount: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 150000000
          ✓ Drop 1 node, Fill 1 node partially, Remove 2 order with a unfilled amount: Target amount is
  280000000
          ✓ Drop 1 node, Fill 1 node partially, Remove 2 order without a unfilled amount: Target amount
  is 300000000
          ✓ Drop 2 nodes: Target amount is 400000000
          ✓ Drop 2 nodes, Fill 1 node partially: Target amount is 450000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 3 order with a unfilled amount: Target amount
  is 650000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 3 order without a unfilled amount: Target
  amount is 700000000
```

```
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        Many nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop multiple nodes less than the root: Target amount is 400000000
          ✓ Drop multiple nodes less than the root, Fill root node partially: Target amount is
1300000000
          ✓ Drop multiple nodes less than or equal to the root: Target amount is 1600000000
          ✓ Drop multiple nodes across the root: Target amount is 3000000000
          ✓ Drop all nodes: Target amount is 7500000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 10000000000
      Drop nodes from the tree by repeated inserting and dropping
        1 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop all nodes: Target amount is 100000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        2 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000
          ✓ Drop all nodes: Target amount is 400000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        3 nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000
          ✓ Drop 2 nodes: Target amount is 400000000
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        3 nodes with multiple orders in the tree
          ✓ Fill 1 node partially, Remove 1 order with a unfilled amount: Target amount is 25000000
          ✓ Fill 1 node partially, Remove 1 order without a unfilled amount: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 150000000
          ✓ Drop 1 node, Fill 1 node partially, Remove 2 order with a unfilled amount: Target amount is
280000000
          ✓ Drop 1 node, Fill 1 node partially, Remove 2 order without a unfilled amount: Target amount
is 300000000
          ✓ Drop 2 nodes: Target amount is 400000000
          ✓ Drop 2 nodes, Fill 1 node partially: Target amount is 450000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 3 order with a unfilled amount: Target amount
is 650000000
          ✓ Drop 2 nodes, Fill 1 node partially, Remove 3 order without a unfilled amount: Target
amount is 700000000
          ✓ Drop all nodes: Target amount is 900000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 1000000000
        Many nodes in the tree
          ✓ Fill 1 node partially: Target amount is 50000000
          ✓ Drop 1 node: Target amount is 100000000
          ✓ Drop multiple nodes less than the root: Target amount is 400000000
          ✓ Drop multiple nodes less than the root, Fill root node partially: Target amount is
1300000000
          ✓ Drop multiple nodes less than or equal to the root: Target amount is 1600000000
          ✓ Drop multiple nodes across the root: Target amount is 3000000000
          ✓ Drop all nodes: Target amount is 7500000000
          ✓ Drop all nodes using an exceeding amount: Target amount is 10000000000
    Borrowing limit orders
      Drop nodes from the tree
        1 nodes in the tree
          ✓ Drop all nodes: Target amount is 100000000, Limit value 9800
          ✓ Drop all nodes by limitValue: Target amount is 200000000, Limit value 9800
        2 nodes in the tree
          ✓ Drop 1 node: Target amount is 100000000, Limit value 9800
          ✓ Drop 1 node by limitValue: Target amount is 200000000, Limit value 9800
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000, Limit value 9801
          ✓ Drop all nodes: Target amount is 400000000, Limit value 9801
          ✓ Drop all nodes by limitValue: Target amount is 1000000000, Limit value 9801
        3 nodes in the tree
          ✓ Drop 1 node: Target amount is 100000000, Limit value 9800
          ✓ Drop 1 node by limitValue: Target amount is 200000000, Limit value 9800
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000, Limit value 9801
          ✓ Drop 2 nodes: Target amount is 400000000, Limit value 9801
          ✓ Drop 2 nodes by limitValue: Target amount is 900000000, Limit value 9801
          ✓ Drop all nodes: Target amount is 900000000, Limit value 9802
          ✓ Drop all nodes by limitValue: Target amount is 1000000000, Limit value 9802
        3 discontinuous nodes in the tree
          ✓ Drop 1 node: Target amount is 1000000000, Limit value 9801
          ✓ Drop 1 node, Fill 1 node partially: Target amount is 200000000, Limit value 9803
          ✓ Drop 2 node: Target amount is 1000000000, Limit value 9803
          ✓ Drop all nodes: Target amount is 1000000000, Limit value 9805
    Borrowing unwind orders
      Drop nodes from the tree
        1 nodes in the tree
          ✓ Fill 1 node partially: Unwind future value 125000000
          ✓ Drop all nodes: Unwind future value 250000000
          ✓ Drop all nodes without limits and amounts: Unwind future value 0
          ✓ Drop all nodes by an exceeding amount: Unwind future value 300000000
        2 nodes in the tree
          ✓ Fill 1 node partially: Unwind future value 125000000
          ✓ Drop 1 node: Unwind future value 250000000
          ✓ Drop 1 node, Fill 1 node partially: Unwind future value 375000000
          ✓ Drop all nodes: Unwind future value 800000000
          ✓ Drop all nodes without limits and amounts: Unwind future value 0
          ✓ Drop all nodes by an exceeding amount: Unwind future value 1000000000
        3 nodes in the tree
          ✓ Fill 1 node partially: Unwind future value 125000000
          ✓ Drop 1 node: Unwind future value 250000000
          ✓ Drop 1 node, Fill 1 node partially: Unwind future value 375000000
          ✓ Drop 2 nodes: Unwind future value 800000000
          ✓ Drop 2 nodes, Fill 1 node partially: Unwind future value 1000000000
          ✓ Drop all nodes: Unwind future value 1200000000
          ✓ Drop all nodes without limits and amounts: Unwind future value 0
          ✓ Drop all nodes by an exceeding amount: Unwind future value 2000000000
    Drop and Insert
      ✓ Insert a lend order to the dropped node
      ✓ Insert a borrow order to the dropped node
  Estimation
    Estimate the dropped amount from the lending tree
      Estimate the dropped FV amount by PV amount
        ✓ Drop 1 node partially
        ✓ Drop 1 node
        ✓ Drop 1 node, Fill 1 node partially
        ✓ Drop 2 nodes, Fill 1 node partially
      Estimate the dropped PV amount by FV amount
        ✓ Drop 1 node partially
        ✓ Drop 1 node
        ✓ Drop 1 node, Fill 1 node partially
        ✓ Drop 2 nodes, Fill 1 node partially
    Estimate the dropped amount from the borrowing tree
      Estimate the dropped FV amount by PV amount
        ✓ Drop 1 node partially
        ✓ Drop 1 node
        ✓ Drop 1 node, Fill 1 node partially
        ✓ Drop 2 nodes, Fill 1 node partially
      Estimate the dropped PV amount by FV amount
        ✓ Drop 1 node partially
        ✓ Drop 1 node
        ✓ Drop 1 node, Fill 1 node partially
        ✓ Drop 2 nodes, Fill 1 node partially

  OrderStatisticsTree - insert and delete
Number of steps: 214
element, orderCount
  4827 0
  1299 0
  5193 0
  287 0
  7688 0
  5427 0
  2447 0
  9023 0
  4457 0
  9057 0
```

```
7168 0
9373 0
1290 0
6498 0
7076 0
2191 0
1234 0
3971 0
1113 0
7367 0
4370 0
4366 0
1580 0
7946 0
2760 0
4816 0
5782 0
7919 0
7248 0
4438 0
6426 0
8751 0
9202 0
4669 0
4574 0
6952 0
8246 0
9558 0
2386 0
4680 0
4584 0
4760 0
4311 0
4888 0
5753 0
544 0
3471 0
2915 0
1760 0
797 0
4251 0
7509 0
5429 0
1815 0
8971 0
4641 0
412 0
6180 0
7897 0
8118 0
7940 0
641 0
6497 0
3082 0
6918 0
1005 0
1105 0
9416 0
23 0
396 0
637 0
467 0
7096 0
8187 0
6517 0
6479 0
8779 0
3696 0
4116 0
2283 0
4817 0
1741 0
```

```
6465 0
2750 0
9414 0
1505 0
9757 0
627 0
969 0
500 0
3277 0
2316 0
8343 0
6151 0
7609 0
7603 0
8419 0
3148 0
8626 0
4042 0
246 0
6016 0
9136 0
5044 0
9843 0
8382 0
1715 0
4827 0
1299 0
5193 0
287 0
7688 0
5427 0
2447 0
9023 0
4457 0
9057 0
7168 0
9373 0
1290 0
6498 0
7076 0
2191 0
1234 0
3971 0
1113 0
7367 0
4370 0
4366 0
1580 0
7946 0
2760 0
4816 0
5782 0
7919 0
7248 0
4438 0
6426 0
8751 0
9202 0
4669 0
4574 0
6952 0
8246 0
9558 0
2386 0
4680 0
4584 0
4760 0
4311 0
4888 0
5753 0
544 0
3471 0
```

```
  2915 0
  1760 0
  797 0
  4251 0
  7509 0
  5429 0
  1815 0
  8971 0
  4641 0
  412 0
  6180 0
  7897 0
  8118 0
  7940 0
  641 0
  6497 0
  3082 0
  6918 0
  1005 0
  1105 0
  9416 0
  23 0
  396 0
  637 0
  467 0
  7096 0
  8187 0
  6517 0
  6479 0
  8779 0
  3696 0
  4116 0
  2283 0
  4817 0
  1741 0
  6465 0
  2750 0
  9414 0
  1505 0
  9757 0
  627 0
  969 0
  500 0
  3277 0
  2316 0
  8343 0
  6151 0
  7609 0
  7603 0
  8419 0
  3148 0
  8626 0
  4042 0
  246 0
  6016 0
  9136 0
  5044 0
  9843 0
  8382 0
  1715 0
Tree Properties
  Root Count 0
  First 0
  Last 0
  Root Value 0
Node Details, (crawled in order), value, parent, left, right, red, head, tail, orderCounter

See if values exists
value, exists
  4827 false
  1299 false
  5193 false
```

```
287 false
7688 false
5427 false
2447 false
9023 false
4457 false
9057 false
7168 false
9373 false
1290 false
6498 false
7076 false
2191 false
1234 false
3971 false
1113 false
7367 false
4370 false
4366 false
1580 false
7946 false
2760 false
4816 false
5782 false
7919 false
7248 false
4438 false
6426 false
8751 false
9202 false
4669 false
4574 false
6952 false
8246 false
9558 false
2386 false
4680 false
4584 false
4760 false
4311 false
4888 false
5753 false
544 false
3471 false
2915 false
1760 false
797 false
4251 false
7509 false
5429 false
1815 false
8971 false
4641 false
412 false
6180 false
7897 false
8118 false
7940 false
641 false
6497 false
3082 false
6918 false
1005 false
1105 false
9416 false
23 false
396 false
637 false
467 false
7096 false
8187 false
6517 false
```

```
6479 false
8779 false
3696 false
4116 false
2283 false
4817 false
1741 false
6465 false
2750 false
9414 false
1505 false
9757 false
627 false
969 false
500 false
3277 false
2316 false
8343 false
6151 false
7609 false
7603 false
8419 false
3148 false
8626 false
4042 false
246 false
6016 false
9136 false
5044 false
9843 false
8382 false
1715 false
4827 false
1299 false
5193 false
287 false
7688 false
5427 false
2447 false
9023 false
4457 false
9057 false
7168 false
9373 false
1290 false
6498 false
7076 false
2191 false
1234 false
3971 false
1113 false
7367 false
4370 false
4366 false
1580 false
7946 false
2760 false
4816 false
5782 false
7919 false
7248 false
4438 false
6426 false
8751 false
9202 false
4669 false
4574 false
6952 false
8246 false
9558 false
2386 false
4680 false
```

```
4584 false
4760 false
4311 false
4888 false
5753 false
544 false
3471 false
2915 false
1760 false
797 false
4251 false
7509 false
5429 false
1815 false
8971 false
4641 false
412 false
6180 false
7897 false
8118 false
7940 false
641 false
6497 false
3082 false
6918 false
1005 false
1105 false
9416 false
23 false
396 false
637 false
467 false
7096 false
8187 false
6517 false
6479 false
8779 false
3696 false
4116 false
2283 false
4817 false
1741 false
6465 false
2750 false
9414 false
1505 false
9757 false
627 false
969 false
500 false
3277 false
2316 false
8343 false
6151 false
7609 false
7603 false
8419 false
3148 false
8626 false
4042 false
246 false
6016 false
9136 false
5044 false
9843 false
8382 false
1715 false
  ✓ Insert all orders and delete after

ProxyController
  Initialize
    ✓ Deploy a ProxyController contract
  Register contracts
    ✓ Register a CurrencyController contract
    ✓ Fail to set a contract due to invalid caller
    ✓ Update a CurrencyController contract
    ✓ Fail to set a contract due to invalid input
    ✓ Register multiple contracts using multicall
    ✓ Fail to register contracts due to execution by non-owner
  Get contract address
    ✓ Successfully get a proxy address
    ✓ Fail to get a proxy address due to empty data
    ✓ Fail to call a contract due to missing address
  Use contracts through the Proxy
    ✓ Successfully call a CurrencyController contract
    ✓ Fail to call a CurrencyController contract due to direct access
  Change Admin
    ✓ Successfully change admins of a proxy contract
    ✓ Fail to change admins of a proxy contract due to execution by non-owner

ReserveFund
  Initialize
    ✓ Fail to call initialization due to duplicate execution
    ✓ Fail to call initialization due to execution by non-proxy contract
  Pause
    ✓ Pause and Unpause
    ✓ Change the operator
    ✓ Remove operator role from another user
    ✓ Fail to pause due to non-operator caller
    ✓ Fail to unpause due to non-operator caller
    ✓ Fail to revoke role due to own role
    ✓ Fail to renounce role due to not allowed access
  Deposit
    ✓ Deposit ERC20 token
    ✓ Deposit ETH
    ✓ Fail to deposit token due to execution by non-owner
  Withdraw
    ✓ Withdraw funds
    ✓ Fail to withdraw token due to execution by non-owner
  Execute transaction
    ✓ Execute emergency settlement
    ✓ Execute a deposit transaction
    ✓ Fail to execute a transaction due to execution by non-owner
    ✓ Fail to execute transactions due to execution by non-owner
    ✓ Fail to execute transactions due to empty inputs
    ✓ Fail to execute transactions due to input array length mismatch: _data
    ✓ Fail to execute transactions due to input array length mismatch: _values

BokkyPooBahsDateTimeContract
  Test functions library functions
    Test basic read time functions
      ✓ timestampToDate
      ✓ getDaysInMonth
      ✓ getDayOfWeek
      ✓ getYear
      ✓ getMonth
      ✓ getDay
      ✓ getHour
      ✓ getMinute
    Test basic time addition functions
      ✓ addedYears
      ✓ addMonths
      ✓ addDays
      ✓ addHours
      ✓ addMinutes
    Test basic time subtraction functions
      ✓ subYears
      ✓ subMonths
      ✓ subDays
      ✓ subHours
      ✓ subMinutes
    Test basic time difference functions
      ✓ diffYears
      ✓ diffMonths
      ✓ diffDays
      ✓ diffHours
      ✓ diffMinutes

TokenVault
  Initialize
    ✓ Update the liquidation configuration
    ✓ Fail to call updateLiquidationConfiguration due to invalid rate
    ✓ Fail to call initialization due to duplicate execution
    ✓ Fail to call initialization due to execution by non-proxy contract
  Currencies
    ✓ Register currency
    ✓ Update collateral currency to non-collateral currency
    ✓ Register non-collateral currency to collateral currency
    ✓ Fail to receive ETH due to execution by non-WETH contract
    ✓ Fail to register currency due to execution by non-owner
    ✓ Fail to update currency due to execution by non-owner
    ✓ Fail to register currency due to nonexistent currency
    ✓ Fail to register currency due to duplicate registration
    ✓ Fail to register currency due to zero address
    ✓ Fail to register currency due to market termination
    ✓ Fail to update currency due to market termination
  Coverage
    ✓ Calculate the coverage without deposit
    ✓ Calculate the coverage with deposit
    ✓ Calculate the coverage for borrowing orders
    ✓ Calculate the coverage for lending orders
    ✓ Calculate the coverage for lending orders that exceed the deposit amount.
  Deposit & Withdraw
    ✓ Deposit ERC20 token
    ✓ Deposit ETH
    ✓ Deposit multiple tokens using multicall
    ✓ Get the withdrawable amount with the working orders & Withdraw collateral
    ✓ Get the withdrawable amount with the borrowed amount
    ✓ Get the withdrawable amount with with the debt amount
    ✓ Add and remove the collateral amount
    ✓ Reset the collateral amount
    ✓ Add an amount in a currency that is not accepted as collateral
    ✓ Get the withdrawable amount per currency
    ✓ Get the withdrawable amount per currency with the borrowing working orders
    ✓ Get the withdrawable amount per currency with the lending working orders
    ✓ Get the liquidation amount
    ✓ Get the liquidation amount decreased by a maximum
    ✓ Get the liquidation fees
    ✓ Fail to deposit token due to unregistered currency
    ✓ Fail to withdraw token due to unregistered currency
    ✓ Fail to call addDepositAmount due to unregistered currency
    ✓ Fail to call removeDepositAmount due to unregistered currency
    ✓ Fail to call executeForcedReset due to unregistered currency
    ✓ Fail to call transferFrom due to unregistered currency
    ✓ Fail to call addDepositAmount due to invalid caller
    ✓ Fail to call removeDepositAmount due to invalid caller
    ✓ Fail to call executeForcedReset due to invalid caller
    ✓ Fail to call transferFrom due to invalid caller
    ✓ Fail to call addDepositAmount due to invalid amount
    ✓ Fail to call deposit due to zero amount
    ✓ Fail to call withdraw due to zero amount
    ✓ Fail to call deposit due to no transfer of native token
    ✓ Fail to deposit token due to transfer of native token
    ✓ Fail to call deposit due to lending market termination
    ✓ Fail to withdraw due to redemption required
    ✓ Fail to withdraw due to insolvency
    ✓ Fail to get liquidation amount due to no collateral
    ✓ Deposit funds from Alice
    ✓ Withdraw funds from Alice
    ✓ Fail to call deposit from Alice due to lending market termination
  Transfer
    ✓ Transfer from Alice to Bob
    ✓ Transfer from Alice to Bob with over amount
    ✓ Fail to transfer deposits due to invalid caller
  Pause/Unpause operations
    ✓ Pause token vault
```

```
      ✓ Unpause token vault
      ✓ Change the operator
    Borrowable amount calculations
      ✓ Without collateral
      ✓ With collateral, unused
      ✓ With collateral, partially used
      ✓ With collateral, totally used
      ✓ Without collateral, has claimable amount
      ✓ With collateral, has claimable amount
      ✓ With collateral, has funds (claimable > collateral)
      ✓ With collateral, has funds (claimable == collateral)

·-------------------------------------------------------------------------|---------------------------|---
----------|-----------------------------·
|                          Solc version: 0.8.19                           ·  Optimizer enabled: true  ·
Runs: 200  ·  Block limit: 30000000 gas  │
··························································································|···························|
·············|······························
|  Methods
│
··································|·········································|··············|··············|
·············|···············|··············
|  Contract                       ·  Method                                 ·  Min         ·  Max         ·
Avg         ·  # calls      ·  usd (avg)  │
··································|·········································|··············|··············|
·············|···············|··············
|  AddressResolver                ·  importAddresses                        ·       35714  ·      491154  ·
325232  ·           99  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  BeaconProxyController          ·  changeBeaconProxyAdmins                ·           -  ·           -  ·
40461  ·            1  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  BeaconProxyController          ·  multicall                              ·      190536  ·     1037350  ·
543445  ·            3  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  BeaconProxyController          ·  setFutureValueVaultImpl                ·       48853  ·      346212  ·
335957  ·           58  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  BeaconProxyController          ·  setLendingMarketImpl                   ·       48800  ·      346159  ·
328317  ·           50  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  BeaconProxyControllerCaller   ·  deployFutureValueVault                 ·           -  ·           -  ·
472280  ·            2  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  BeaconProxyControllerCaller   ·  deployLendingMarket                    ·           -  ·           -  ·
576787  ·            1  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  CurrencyController             ·  addCurrency                            ·      123049  ·      259537  ·
223010  ·           89  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  CurrencyController             ·  removeCurrency                         ·       36690  ·       36700  ·
36693  ·            7  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  CurrencyController             ·  updateHaircut                          ·       33411  ·       55335  ·
39773  ·            8  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  CurrencyController             ·  updatePriceFeed                        ·       75418  ·      128982  ·
102200  ·            4  ·          -  │
··································|·········································|··············|··············|
·············|···············|··············
|  FutureValueVaultCaller         ·  decrease                               ·           -  ·           -  ·
112243  ·            3  ·          -  │
```

```
································|·········································|·············|·············|
·············|···············|··············
|  FutureValueVaultCaller       ·  deployFutureValueVault                  ·          -  ·          -  ·
472673  ·           20  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  FutureValueVaultCaller       ·  executeForcedReset                      ·      43657  ·      47086  ·
45372  ·            4  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  FutureValueVaultCaller       ·  executeForcedReset                      ·      42766  ·      45969  ·
44368  ·            4  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  FutureValueVaultCaller       ·  increase                                ·     111993  ·     112005  ·
112004  ·            8  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  FutureValueVaultCaller       ·  transferFrom                            ·          -  ·          -  ·
92593  ·            2  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  cleanUpBalance                          ·      40208  ·     121854  ·
81031  ·            4  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  executeAutoRoll                         ·     180407  ·     180519  ·
180509  ·           18  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  executeForcedReset                      ·      49996  ·      54771  ·
52384  ·            4  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  executeForcedReset                      ·          -  ·          -  ·
37953  ·            2  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  initializeCurrencySetting               ·     240506  ·     240518  ·
240516  ·           39  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  transferFrom                            ·          -  ·          -  ·
61187  ·            2  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  updateGenesisValueWithFutureValue       ·      66274  ·     137889  ·
121555  ·           24  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  updateGenesisValueWithResidualAmount    ·          -  ·          -  ·
107800  ·            1  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller      ·  updateInitialCompoundFactor             ·          -  ·          -  ·
70911  ·            1  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarket                ·  pause                                   ·      52116  ·      52481  ·
52220  ·            7  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarket                ·  unpause                                 ·      30249  ·      30633  ·
30359  ·            7  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  cleanUpOrders                           ·      86478  ·      86518  ·
86498  ·            4  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  createOrderBook                         ·     132123  ·     186252  ·
149452  ·           69  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  deployLendingMarket                    ·     576962  ·     576974  ·
576968  ·           14  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  executeAutoRoll                        ·      73798  ·      90898  ·
82348  ·            2  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  executeItayoseCall                     ·      95518  ·     307048  ·
189394  ·           20  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  executeOrder                           ·     123462  ·     395200  ·
267052  ·          199  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  executePreOrder                        ·     340564  ·     403091  ·
369016  ·           57  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  pause                                  ·          -  ·          -  ·
63211  ·            1  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  unpause                                ·          -  ·          -  ·
41321  ·            1  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketCaller          ·  unwindPosition                         ·      95696  ·     156941  ·
133345  ·            4  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  cancelOrder                            ·     123136  ·     143434  ·
125681  ·           21  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  cleanUpAllFunds                        ·     205749  ·     769436  ·
403690  ·           26  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  createOrderBook                        ·     230674  ·     284975  ·
260393  ·         1158  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  depositAndExecuteOrder                 ·     582668  ·    1213789  ·
879234  ·          183  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  depositAndExecutesPreOrder             ·     656047  ·     832881  ·
788670  ·            4  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeEmergencySettlement             ·     230516  ·     720375  ·
471821  ·           34  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeEmergencyTermination            ·     214012  ·     456339  ·
305085  ·           22  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeForcedRepayment                 ·     443811  ·     550978  ·
495423  ·            6  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeItayoseCall                     ·     120023  ·     460987  ·
127338  ·          890  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeLiquidationCall                 ·     612155  ·    1007964  ·
702327  ·           14  ·           -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeOrder                            ·     253037  ·    1664463  ·
736687  ·         3067  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executePreOrder                         ·     349952  ·     708868  ·
545586  ·           98  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeRedemption                       ·     230310  ·     321544  ·
270860  ·            5  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  executeRepayment                        ·     319635  ·     449029  ·
373574  ·            5  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  initializeLendingMarket                 ·    1317098  ·    1319434  ·
1318777  ·          224  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  pauseLendingMarket                      ·          -  ·          -  ·
76446  ·            3  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  rotateOrderBooks                        ·     379882  ·     691739  ·
414447  ·          912  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  unpauseLendingMarket                    ·          -  ·          -  ·
54548  ·            2  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  unwindPosition                          ·     456631  ·     652391  ·
546539  ·           15  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  updateCircuitBreakerLimitRange          ·          -  ·          -  ·
52849  ·            2  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  updateMinDebtUnitPrice                  ·          -  ·          -  ·
41308  ·            1  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  LendingMarketController      ·  updateOrderFeeRate                      ·          -  ·          -  ·
59597  ·            2  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  Liquidator                   ·  addOperator                             ·      56526  ·      56591  ·
56561  ·            4  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  Liquidator                   ·  deposit                                 ·      59234  ·     206803  ·
106091  ·         1098  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  Liquidator                   ·  executeForcedRepayment                  ·    1009316  ·    1102646  ·
1052928  ·            8  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  Liquidator                   ·  executeLiquidationCall                  ·    1283151  ·    3001142  ·
1910029  ·           21  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  Liquidator                   ·  executeTransaction                      ·      43254  ·     611064  ·
305307  ·           14  ·          -  |
································|··········································|·············|·············|
·············|···············|··············
|  Liquidator                   ·  executeTransactions                     ·      65611  ·     196645  ·
151237  ·            9  ·          -  |
·································|·········································|·············|·············|
·············|···············|··············
|  Liquidator                    ·  removeOperator                          ·      34690  ·      34790  ·
34742  ·            3  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  Liquidator                    ·  renounceOwnership                       ·          -  ·          -  ·
28815  ·            1  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  Liquidator                    ·  revokeRole                              ·          -  ·          -  ·
35182  ·            1  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  Liquidator                    ·  transferOwnership                       ·          -  ·          -  ·
34224  ·            1  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  Liquidator                    ·  withdraw                                ·      58241  ·     609293  ·
315102  ·           64  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MigrationAddressResolver      ·  buildCaches                             ·      72018  ·     433732  ·
180322  ·           83  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockERC20                     ·  approve                                 ·          -  ·          -  ·
46302  ·            1  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockERC20                     ·  transfer                                ·          -  ·          -  ·
46786  ·            1  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockUniswapQuoter             ·  setToken                                ·      44167  ·      44179  ·
44175  ·            3  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockUniswapRouter             ·  setToken                                ·      44167  ·      44179  ·
44175  ·            3  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockUSDC                      ·  approve                                 ·      26702  ·      46602  ·
44455  ·          112  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockUSDC                      ·  transfer                                ·      51514  ·      51574  ·
51549  ·          131  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockV3Aggregator              ·  updateAnswer                            ·     102972  ·     105808  ·
104487  ·           43  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockWBTC                      ·  approve                                 ·          -  ·          -  ·
46266  ·            8  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockWBTC                      ·  transfer                                ·      51538  ·      51550  ·
51549  ·           23  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockWFIL                      ·  approve                                 ·      29214  ·      46602  ·
46253  ·          885  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  MockWFIL                      ·  transfer                                ·      34486  ·      51598  ·
51378  ·          164  ·           -  |
·································|·········································|·············|·············|
·············|···············|··············
|  OrderStatisticsTreeContract   ·  dropValuesFromFirst                     ·      36226  ·     182465  ·
84911  ·          215  ·           -  |
································|·········································|·············|·············|
·············|···············|··············
|  OrderStatisticsTreeContract  ·  dropValuesFromLast                      ·      36247  ·     215352  ·
86547  ·          215  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  OrderStatisticsTreeContract  ·  insertAmountValue                       ·      95653  ·     325118  ·
169546  ·         2573  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  OrderStatisticsTreeContract  ·  removeAmountValue                       ·      59914  ·     206332  ·
117150  ·          107  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  changeProxyAdmins                       ·          -  ·          -  ·
35374  ·            1  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  multicall                               ·          -  ·          -  ·
761440  ·            2  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setAddressResolverImpl                  ·     392678  ·     392702  ·
392701  ·          115  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setBeaconProxyControllerImpl            ·     406177  ·     406201  ·
406200  ·          160  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setCurrencyControllerImpl               ·      54431  ·     383139  ·
347410  ·           46  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setGenesisValueVaultImpl                ·     381687  ·     381699  ·
381698  ·           66  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setLendingMarketControllerImpl          ·     478430  ·     478442  ·
478441  ·           62  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setReserveFundImpl                      ·          -  ·          -  ·
481518  ·           30  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  ProxyController              ·  setTokenVaultImpl                       ·     554510  ·     554522  ·
554521  ·           30  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  TokenVault                   ·  registerCurrency                        ·      80184  ·     154506  ·
122925  ·           98  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  TokenVault                   ·  updateCurrency                          ·      48227  ·     112711  ·
95456  ·           21  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  TokenVault                   ·  updateLiquidationConfiguration          ·      36071  ·      40635  ·
36901  ·           11  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  TokenVaultCaller             ·  addDepositAmount                        ·      51585  ·     133194  ·
113450  ·            5  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  TokenVaultCaller             ·  cleanUpFunds                            ·      56693  ·     588492  ·
256968  ·           41  ·          -  |
································|·········································|·············|·············|
·············|···············|··············
|  TokenVaultCaller             ·  depositFrom                             ·      89337  ·     170910  ·
154595  ·            5  ·          -  |
·······························|·········································|·············|·············|
·············|···············|··············
|  TokenVaultCaller             ·  executeForcedReset                        ·          -  ·          -  ·
49919  ·           1  ·          -  |
·······························|·········································|·············|·············|
·············|···············|··············
|  TokenVaultCaller             ·  removeDepositAmount                       ·      52092  ·      52424  ·
52258  ·           2  ·          -  |
·······························|·········································|·············|·············|
·············|···············|··············
|  TokenVaultCaller             ·  transferFrom                              ·      58472  ·     128523  ·
100108  ·           5  ·          -  |
·······························|·········································|·············|·············|
·············|···············|··············
|  Deployments                                                              ·
·  % of limit   ·             |
·········································································|·············|·············|
·············|···············|··············
|  AddressResolver                                                          ·          -  ·          -  ·
852927  ·        2.8 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  BeaconProxyController                                                    ·          -  ·          -  ·
2358928  ·        7.9 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  BeaconProxyControllerCaller                                              ·          -  ·          -  ·
221772  ·        0.7 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  BokkyPooBahsDateTimeContract                                             ·          -  ·          -  ·
1304957  ·        4.3 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  CurrencyController                                                       ·          -  ·          -  ·
2083149  ·        6.9 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  DepositManagementLogic                                                   ·          -  ·          -  ·
2074019  ·        6.9 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  FundManagementLogic                                                      ·    4382755  ·    4382767  ·
4382765  ·       14.6 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  FutureValueVault                                                         ·          -  ·          -  ·
1514440  ·          5 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  FutureValueVaultCaller                                                   ·     461350  ·     461362  ·
461361  ·        1.5 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  GenesisValueVault                                                        ·          -  ·          -  ·
2574645  ·        8.6 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  GenesisValueVaultCaller                                                  ·          -  ·          -  ·
469422  ·        1.6 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  LendingMarket                                                            ·    3014801  ·    3015185  ·
3015144  ·       10.1 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  LendingMarketCaller                                                      ·     888849  ·     888861  ·
888861  ·          3 %  ·          -  |
·········································································|·············|·············|
·············|···············|··············
|  LendingMarketController                                                  ·    4670556  ·    4670808  ·
4670753  ·       15.6 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  LendingMarketOperationLogic                                  ·          -  ·          -  ·
2732218  ·        9.1 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  LendingMarketReader                                          ·    2650315  ·    2650327  ·
2650325  ·        8.8 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  LendingMarketUserLogic                                       ·    1691416  ·    1691536  ·
1691517  ·        5.6 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  LiquidationLogic                                             ·    2010769  ·    2010937  ·
2010910  ·        6.7 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  Liquidator                                                   ·    2305145  ·    2323302  ·
2317922  ·        7.7 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MigrationAddressResolver                                     ·          -  ·          -  ·
192897  ·        0.6 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockERC20                                                    ·          -  ·          -  ·
1290134  ·        4.3 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockUniswapQuoter                                            ·          -  ·          -  ·
627758  ·        2.1 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockUniswapRouter                                            ·          -  ·          -  ·
944558  ·        3.1 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockUSDC                                                     ·    1336564  ·    1336648  ·
1336617  ·        4.5 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockV3Aggregator                                             ·     663084  ·     663120  ·
663103  ·        2.2 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockWBTC                                                     ·    1336588  ·    1336672  ·
1336615  ·        4.5 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockWETH9                                                    ·          -  ·          -  ·
559063  ·        1.9 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  MockWFIL                                                     ·    1336860  ·    1336956  ·
1336936  ·        4.5 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  OrderActionLogic                                             ·    4365197  ·    4365245  ·
4365240  ·       14.6 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  OrderBookLogic                                               ·          -  ·          -  ·
3110723  ·       10.4 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  OrderReaderLogic                                             ·          -  ·          -  ·
1764108  ·        5.9 %  ·          -  |
·························································|·············|·············|
·············|···············|·············
|  OrderStatisticsTreeContract                                  ·          -  ·          -  ·
2623359  ·        8.7 %  ·          -  |
·····························································|············|············|
············|··············|·············
|  ProxyController                                             ·    1714950  ·    1742727  ·
1715181  ·       5.7 %  ·          -  |
·····························································|············|············|
············|··············|·············
|  QuickSort                                                   ·          -  ·          -  ·
294449  ·         1 %  ·          -  |
·····························································|············|············|
············|··············|·············
|  ReserveFund                                                 ·          -  ·          -  ·
1923568  ·       6.4 %  ·          -  |
·····························································|············|············|
············|··············|·············
|  TokenVault                                                  ·    3416748  ·    3416976  ·
3416963  ·      11.4 %  ·          -  |
·····························································|············|············|
············|··············|·············
|  TokenVaultCaller                                            ·          -  ·          -  ·
742719  ·       2.5 %  ·          -  |
·------------------------------------------------------------|------------|------------|---
---------|--------------|-------------·

  1890 passing (27m)
```

Code Coverage

While line and statement coverage metrics are high, we highly recommend improving the branch coverage to a minimum of `95%`.

Fix review update: the branch coverage was improved to `87.94 %`. It is recommended to cover the `100%` of branches in the codebase.

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
migration/	100	50	100	100	
MigrationAddressResolver.sol	100	50	100	100	
protocol/	99.78	93.31	99.59	99.68	
AddressResolver.sol	100	100	100	100	
BeaconProxyController.sol	100	95.45	100	100	
CurrencyController.sol	100	100	100	100	
FutureValueVault.sol	100	96.43	100	100	
GenesisValueVault.sol	100	92.45	100	99.34	413
LendingMarket.sol	100	93.55	100	100	
LendingMarketController.sol	98.65	84.82	98	98.86	131
ProxyController.sol	100	92.31	100	100	
ReserveFund.sol	100	100	100	100	
TokenVault.sol	100	97.56	100	100	
protocol/interfaces/	100	100	100	100	

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
IAddressResolver.sol	100	100	100	100	
IBeaconProxyController.sol	100	100	100	100	
ICurrencyController.sol	100	100	100	100	
IFutureValueVault.sol	100	100	100	100	
IGenesisValueVault.sol	100	100	100	100	
ILendingMarket.sol	100	100	100	100	
ILendingMarketController.sol	100	100	100	100	
ILiquidationReceiver.sol	100	100	100	100	
INativeToken.sol	100	100	100	100	
IProxyController.sol	100	100	100	100	
IReserveFund.sol	100	100	100	100	
ITokenVault.sol	100	100	100	100	
protocol/libraries/	94.92	83.11	89.63	94.44	
AddressResolverLib.sol	100	100	100	100	
BokkyPooBahsDateTimeLibrary.sol	83.05	54.17	71.79	76.86	... 407,458,459
Constants.sol	100	100	100	100	
Contracts.sol	100	100	100	100	
OrderBookLib.sol	100	94.53	100	100	
OrderStatisticsTreeLib.sol	95.08	87.73	95.65	95.6	... 3,1052,1053
QuickSort.sol	93.33	75	100	100	
TransferHelper.sol	96	69.23	90.91	96.15	14
protocol/libraries/logics/	99.19	89.68	99.25	98.72	
DepositManagementLogic.sol	97.44	92.31	100	95.16	... 355,358,361
FundManagementLogic.sol	99.03	92.35	96.88	98.36	... 9,1160,1162
LendingMarketOperationLogic.sol	100	91.67	100	100	
LendingMarketUserLogic.sol	100	90.48	100	100	
LiquidationLogic.sol	100	79.69	100	98.97	373

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
OrderActionLogic.sol	100	88.24	100	100	
OrderBookLogic.sol	100	85	100	100	
OrderReaderLogic.sol	97.22	94.44	100	100	
protocol/libraries/math/	100	90	100	100	
RoundingInt256.sol	100	83.33	100	100	
RoundingUint256.sol	100	100	100	100	
protocol/mixins/	95.31	89.13	88.24	96.2	
MixinAccessControl.sol	100	100	100	100	
MixinAddressResolver.sol	85.71	75	69.23	88	73,81,89
MixinLendingMarketConfiguration.sol	100	100	100	100	
MixinLiquidationConfiguration.sol	100	92.86	100	100	
MixinWallet.sol	100	87.5	100	100	
protocol/storages/	100	100	100	100	
AddressResolverStorage.sol	100	100	100	100	
BeaconProxyControllerStorage.sol	100	100	100	100	
CurrencyControllerStorage.sol	100	100	100	100	
FutureValueVaultStorage.sol	100	100	100	100	
GenesisValueVaultStorage.sol	100	100	100	100	
LendingMarketControllerStorage.sol	100	100	100	100	
LendingMarketStorage.sol	100	100	100	100	
ReserveFundStorage.sol	100	100	100	100	
TokenVaultStorage.sol	100	100	100	100	
protocol/storages/libraries/	100	100	100	100	
TransferHelperStorage.sol	100	100	100	100	
protocol/storages/mixins/	100	100	100	100	

File	% Stmts	% Branch	% Funcs	% Lines	Uncovered Lines
MixinAddressResolverStorage.sol	100	100	100	100	
protocol/storages/utils/	100	100	100	100	
AccessControlStorage.sol	100	100	100	100	
OwnableStorage.sol	100	100	100	100	
PausableStorage.sol	100	100	100	100	
protocol/types/	100	100	100	100	
ProtocolTypes.sol	100	100	100	100	
protocol/utils/	83.05	61.11	89.36	84.62	
AccessControl.sol	61.11	35.71	69.23	63.64	... 216,217,218
LockAndMsgSender.sol	100	50	100	83.33	17
Ownable.sol	100	100	100	100	
Pausable.sol	100	87.5	100	100	
Proxyable.sol	100	100	100	100	
UpgradeabilityBeaconProxy.sol	71.43	33.33	83.33	75	15,20
UpgradeabilityProxy.sol	85.71	50	100	87.5	15
UpgradeableBeacon.sol	100	50	100	100	
All files	97.48	87.94	95.91	97.07	

Changelog

- 2023-11-30 - Initial report
- 2023-12-12 - Final report

About Quantstamp

Quantstamp is a global leader in blockchain security. Founded in 2017, Quantstamp's mission is to securely onboard the next billion users to Web3 through its best-in-class Web3 security products and services.

Quantstamp's team consists of cybersecurity experts hailing from globally recognized organizations including Microsoft, AWS, BMW, Meta, and the Ethereum Foundation. Quantstamp engineers hold PhDs or advanced computer science degrees, with decades of combined experience in formal verification, static analysis, blockchain audits, penetration testing, and original leading-edge research.

To date, Quantstamp has performed more than 500 audits and secured over $200 billion in digital asset risk from hackers. Quantstamp has worked with a diverse range of customers, including startups, category leaders and financial institutions. Brands that Quantstamp has worked with include Ethereum 2.0, Binance, Visa, PayPal, Polygon, Avalanche, Curve, Solana, Compound, Lido, MakerDAO, Arbitrum, OpenSea and the World Economic Forum.

Quantstamp's collaborations and partnerships showcase our commitment to world-class research, development and security. We're honored to work with some of the top names in the industry and proud to secure the future of web3.

Notable Collaborations & Customers:

- Blockchains: Ethereum 2.0, Near, Flow, Avalanche, Solana, Cardano, Binance Smart Chain, Hedera Hashgraph, Tezos
- DeFi: Curve, Compound, Maker, Lido, Polygon, Arbitrum, SushiSwap
- NFT: OpenSea, Parallel, Dapper Labs, Decentraland, Sandbox, Axie Infinity, Illuvium, NBA Top Shot, Zora
- Academic institutions: National University of Singapore, MIT

Timeliness of content

The content contained in the report is current as of the date appearing on the report and is subject to change without notice, unless indicated otherwise by Quantstamp; however, Quantstamp does not guarantee or warrant the accuracy, timeliness, or completeness of any report you access using the internet or other means, and assumes no obligation to update any information following publication or other making available of the report to you by Quantstamp.

Notice of confidentiality

This report, including the content, data, and underlying methodologies, are subject to the confidentiality and feedback provisions in your agreement with Quantstamp. These materials are not to be disclosed, extracted, copied, or distributed except to the extent expressly authorized by Quantstamp.

Links to other websites

You may, through hypertext or other computer links, gain access to web sites operated by persons other than Quantstamp. Such hyperlinks are provided for your reference and convenience only, and are the exclusive responsibility of such web sites&aspo; owners. You agree that Quantstamp are not responsible for the content or operation of such web sites, and that Quantstamp shall have no liability to you or any other person or entity for the use of third-party web sites. Except as described below, a hyperlink from this web site to another web site does not imply or mean that Quantstamp endorses the content on that web site or the operator or operations of that site. You are solely responsible for determining the extent to which you may use any content at any other web sites to which you link from the report. Quantstamp assumes no responsibility for the use of third-party software on any website and shall have no liability whatsoever to any person or entity for the accuracy or completeness of any output generated by such software.

Disclaimer

The review and this report are provided on an as-is, where-is, and as-available basis. To the fullest extent permitted by law, Quantstamp disclaims all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement. You agree that your access and/or use of the report and other results of the review, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE. This report is based on the scope of materials and documentation provided for a limited review at the time provided. You acknowledge that Blockchain technology remains under development and is subject to unknown risks and flaws and, as such, the report may not be complete or inclusive of all vulnerabilities. The review is limited to the materials identified in the report and does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. The report does not indicate the endorsement by Quantstamp of any particular project or team, nor guarantee its security, and may not be represented as such. No third party is entitled to rely on the report in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. Quantstamp does not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party, or any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, or any related services and products, any hyperlinked websites, or any other websites or mobile applications, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third party. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.