Testing Documentation

# -TEST Plan

## -Objectives: We tested all Aiken-Lang Functions

## -Scope:

### **Core Functionality Testing**

* **Syntax and Parsing:** Verify correct interpretation of Aiken syntax and grammar.
* **Type System:** Ensure accurate type checking and inference.
* **Basic Operations:** Test arithmetic, logical, and comparison operations.
* **Control Flow:** Verify correct execution of conditional and iterative constructs.
* **Function Definition and Call:** Test function creation, parameter passing, and return values.
* **Data Structures:** Test operations on lists, tuples, and other supported data structures.

### **Advanced Feature Testing**

* **Higher-Order Functions:** Test function composition, currying, and partial application.
* **Pattern Matching:** Verify correct pattern recognition and value extraction.
* **Lazy Evaluation:** Test the behavior of lazy expressions and potential performance implications.
* **Module System:** Test module import, export, and dependency management.
* **Error Handling:** Test exception handling and recovery mechanisms.

### **Performance Testing**

* **Execution Speed:** Measure the time taken for different operations.
* **Memory Usage:** Assess memory consumption for various program scenarios.
* **Algorithm Efficiency:** Evaluate the performance of built-in algorithms and data structures.

### **Compatibility Testing**

* **Platform Compatibility:** Test Aiken on different operating systems and hardware architectures.
* **Compiler/Interpreter Compatibility:** Test with different Aiken implementations.
* **Library Compatibility:** Verify compatibility with external libraries or tools.

### **Security Testing**

* **Input Validation:** Test for vulnerabilities like injection attacks.
* **Data Privacy:** Assess how Aiken handles sensitive data.
* **Code Security:** Check for potential security flaws in the language implementation.

### **Usability Testing**

* **Readability:** Evaluate the clarity and understandability of Aiken code.
* **Developer Experience:** Gather feedback on development tools and productivity.
* **Error Messages:** Assess the helpfulness and clarity of error messages.

### **Specific Feature Testing**

* **Tailored Test Cases:** Create test cases that target the specific feature being tested.
* **Edge Cases:** Explore boundary conditions and unexpected inputs.
* **Integration Testing:** Verify how the feature interacts with other language components.

## -Approach:

## **Testing Approaches for Aiken Lang**

Aiken Lang, as a functional language designed for smart contracts, emphasizes correctness and security. Its testing approaches reflect these priorities.

### **Core Testing Approaches**

1. **Unit Testing:**
   * **Purpose:** To isolate and test individual functions or components.
   * **Approach:** Write test cases to verify expected outputs for various inputs.
   * **Example:** Test arithmetic operations, data structure manipulations, and function logic.
2. **Property-Based Testing:**
   * **Purpose:** To generate random test cases based on properties rather than specific inputs.
   * **Approach:** Define properties that the code should satisfy and use a testing framework to generate random inputs.
   * **Example:** Test associativity and commutativity of arithmetic operations, or invariant preservation in data structures.
3. **Fuzz Testing:**
   * **Purpose:** To find vulnerabilities by feeding random, invalid, or unexpected data to the system.
   * **Approach:** Generate random inputs and observe the system's behavior.
   * **Example:** Test for crashes, exceptions, or unexpected outputs when provided with malformed or invalid data.
4. **Integration Testing:**
   * **Purpose:** To test how different parts of the system interact.
   * **Approach:** Combine multiple components and test their interactions.
   * **Example:** Test how contracts interact with other contracts or with the blockchain.
5. **Model Checking:**
   * **Purpose:** To formally verify the correctness of the system against its specification.
   * **Approach:** Create a mathematical model of the system and use automated tools to check if it satisfies the desired properties.
   * **Example:** Verify that a smart contract adheres to its intended behavior and doesn't have any vulnerabilities.

### **Additional Testing Considerations**

* **Performance Testing:** Measure execution time, memory usage, and other performance metrics.
* **Security Testing:** Assess vulnerability to attacks like reentrancy, integer overflow, and access control issues.
* **Code Coverage:** Ensure that test cases cover different code paths.

### **Aiken Lang Specific Testing Tools and Features**

* **Built-in Testing Framework:** Aiken provides a built-in framework for writing unit and property-based tests.
* **Fuzz Testing Library:** The fuzz library supports property-based testing and fuzzing.
* **Model Checking Integration:** Aiken can potentially integrate with external model checking tools for formal verification.

### **Challenges in Aiken Lang Testing**

* **Complexity of Smart Contracts:** Smart contracts often involve complex logic and interactions, making testing challenging.
* **Limited Test Environments:** Access to real-world blockchain environments might be restricted for testing.
* **Statefulness:** Testing stateful systems can be complex due to the need to manage different states.

## -Resources:

## [**https://github.com/STEAKProtocol/STEAKProtocol?tab=readme-ov-file**](https://github.com/STEAKProtocol/STEAKProtocol?tab=readme-ov-file)

[**https://github.com/Anastasia-Labs/design-patterns/tree/main**](https://github.com/Anastasia-Labs/design-patterns/tree/main)

[**https://github.com/aiken-lang/awesome-aiken?tab=readme-ov-file#Dapps**](https://github.com/aiken-lang/awesome-aiken?tab=readme-ov-file#Dapps)

[**https://github.com/logicalmechanism/assist/tree/main/lib/assist**](https://github.com/logicalmechanism/assist/tree/main/lib/assist)

[**https://github.com/logicalmechanism/assist**](https://github.com/logicalmechanism/assist)

[**https://github.com/spacebudz/nebula/tree/main/contract/src/nebula**](https://github.com/spacebudz/nebula/tree/main/contract/src/nebula)

[**https://github.com/spacebudz/nebula/tree/main**](https://github.com/spacebudz/nebula/tree/main)

[**https://github.com/aiken-extra**](https://github.com/aiken-extra)

[**https://github.com/Cardano-Fans/acca**](https://github.com/Cardano-Fans/acca)

[**https://github.com/minswap/minswap-stableswap?tab=readme-ov-file**](https://github.com/minswap/minswap-stableswap?tab=readme-ov-file)

[**https://github.com/cardano-miners/fortuna**](https://github.com/cardano-miners/fortuna)

[**https://github.com/orgs/aiken-lang/projects/2/views/1**](https://github.com/orgs/aiken-lang/projects/2/views/1)

[**https://github.com/minswap/minswap-stableswap/blob/main/stableswap-docs/formula.md**](https://github.com/minswap/minswap-stableswap/blob/main/stableswap-docs/formula.md)

**https://github.com/MeshJS/marketplace-next-ts-template**

# -Test Cases

# Testing Result & Reports:

# Aiken-Lang Functions Testing

## ​​1. **Test Case Name:** Aiken Steak Protocol

**Link:** [**https://github.com/STEAKProtocol/STEAKProtocol?tab=readme-ov-file**](https://github.com/STEAKProtocol/STEAKProtocol?tab=readme-ov-file)

### **-Description:**

### **#### Technical Methods:**

1. Python

2. Aiken v1.0.25 alpha + 075668b

3. Makefile

4. Typescript

5. Node js

### **#### Technical:**

<div align="center">

<a href="https://github.com/STEAKProtocol/STEAKProtocol">

<img src="https://steakprotocol.com/logo.png" width="200" />

</a>

<h1> STEAK Protocol </h1>

</div>

> Cardano style Proof Of Stake as a Randomness Oracle Smart Contract

The STEAK protocol is a decentralized protocol similar to [ForTuna](https://github.com/cardano-miners/fortuna) that

distributes rewards to users based on their stake in the protocol.

The protocol is implemented as a smart contract on the Cardano blockchain.

It consecutively attaches a new block to the blockchain every 60 seconds (slots).

These blocks can be used as a source of randomness for other protocols as the exact

block hash is practically unpredictable for a given slot.

### **#### Architecture**

The STEAK protocol consists of the following components:

- \*\*Block Chain\*\*: A chain of consecutive blocks that refer to the previous block by its hash.

- \*\*Stake Holder\*\*: A position of tokens that participates in the protocol.

- \*\*Stake Pool\*\*: A withdrawal contract that ensures fair pooling of funds in a stake holder across different users.

Every 60 seconds, a slot leader is elected by the protocol to attach a new block to the blockchain.

The slot leader is chosen pseudorandomly based on the block hash and weighs participations based on the stake of the contributing pool.

### **#### Mainnet**

The chain can be observed on mainnet on any Chain Explorer like CardanoScan at this address: [addr1wytr9jvc6tnavc3s86ws76sfpdau3g3gnezpnz5xhhusnrcket8r9](https://cardanoscan.io/address/addr1wytr9jvc6tnavc3s86ws76sfpdau3g3gnezpnz5xhhusnrcket8r9).

The $STK token on mainnet has this token name and policy id: [`2a80c713e0d518f84e0553957e3d581c9326cfc5931ea1f4dd0fb292.0014df107374616b65636f696e`](https://cardanoscan.io/token/2a80c713e0d518f84e0553957e3d581c9326cfc5931ea1f4dd0fb2920014df107374616b65636f696e)

The $STK token can be obtained by trading on [MuesliSwap](https://muesliswap.com/swap/expert?base=."e=2a80c713e0d518f84e0553957e3d581c9326cfc5931ea1f4dd0fb292.0014df107374616b65636f696e&t=trading).

### **#### Preview Testnet**

The chain can be observed on preview testnet on any Chain Explorer like CexPlorer at this address: [addr\_test1wqtr9jvc6tnavc3s86ws76sfpdau3g3gnezpnz5xhhusnrcd3lmvq](https://preview.cexplorer.io/address/addr\_test1wqtr9jvc6tnavc3s86ws76sfpdau3g3gnezpnz5xhhusnrcd3lmvq/tx#data).

The $STK token on preview testnet has this token name and policy id: [`126a82c32b90b321eec79a79eaacd07abcc902d18cfcf62652708e6d.0014df107374616b65636f696e`](https://preview.cexplorer.io/asset/asset17xpvsxzxvdw4rraeyj4jk3p0gcm9vdetdrh0fa)

The $STK token can be obtained by trading on [preview SundaeSwap v2](https://v2.preview.sundaeswap.finance/).

You can obtain tADA at the [Cardano testnet faucet](https://docs.cardano.org/cardano-testnets/tools/faucet/) - make sure to select "Preview" testnet!

### **#### Mining Blocks**

The protocol uses a Proof of Stake (PoS) consensus mechanism to elect slot leaders.

The PoS mechanism is implemented as a smart contract that selects the slot leader based on the stake of the contributing pool.

The stake of a pool is determined by the number of tokens it holds.

The protocol uses a random seed generated from the previous block hash to select the slot leader.

The slot leader is responsible for attaching a new block to the blockchain every 60 seconds.

To become a participant in the protocol, a user must:

- Register a stake holder or stake pool with the protocol.

- Deposit $STK tokens into the stake holder position or stake pool.

To mine a block, the slot leader must:

- Generate a random seed from the previous block hash.

- Prove that the slot leader has the right to mine the block by providing a proof of stake.

- Attach the new block to the blockchain.

- Distribute rewards to the stake pool based on the stake of the pool.

Code for each of these activities is provided in the `steak\_protocol/offchain` directory.

An example is shown below.

```bash

# Obtain a blockfrost key for the correct network from https://blockfrost.io/

export BLOCKFROST\_PROJECT\_ID=<your-blockfrost-key>

# Alternatively, set up ogmios to point to localhost:1337

# Install all dependencies

poetry install

# Activate the poetry shell

poetry shell

# Create a key pair

python3 steak\_protocol/create\_key\_pair.py alice

# Deposit $STK tokens and at least 10 Ada at the created address

# Using a wallet or something similar

# Option A)

# Then, register as a stake holder

python3 -m steak\_protocol.offchain.stakeholder.init alice --stake\_amount 1000 --stakeholder\_id alice

# Finally, try mining a block

python3 -m steak\_protocol.offchain.stakechain.mine alice --pool\_id alice

# Option B)

# For stake pools, the process is a bit more involved

# First, register a stake pool

python3 -m steak\_protocol.offchain.stakepool.init alice --stakepool\_id apool

# The, place a request to join the pool

python3 -m steak\_protocol.offchain.stakepool.place\_request alice --stakepool\_id apool --stakecoin\_amount 10\_000\_000

# The request can be batched by anyone, but you may want to batch it yourself

python3 -m steak\_protocol.offchain.stakepool.fill\_request alice --no\_stake\_key

# Finally, try mining a block

python3 -m steak\_protocol.offchain.stakechain.mine alice --stakepool\_id apool

# To batch requests from a specific user placed through the frontend, put their staking key

python3 -m steak\_protocol.offchain.stakepool.fill\_request alice --stake\_key stake\_...

```

Note that stake pools can allow further users to join the pool by placing a request to join the pool.

Singleton stake holder can not be pooled, but can still mine blocks.

All corresponding scripts and documentation can be found in the `steak\_protocol/offchain/stakepool` directory.

Any script provides a `--help` flag to show the available options.

### **-Input**

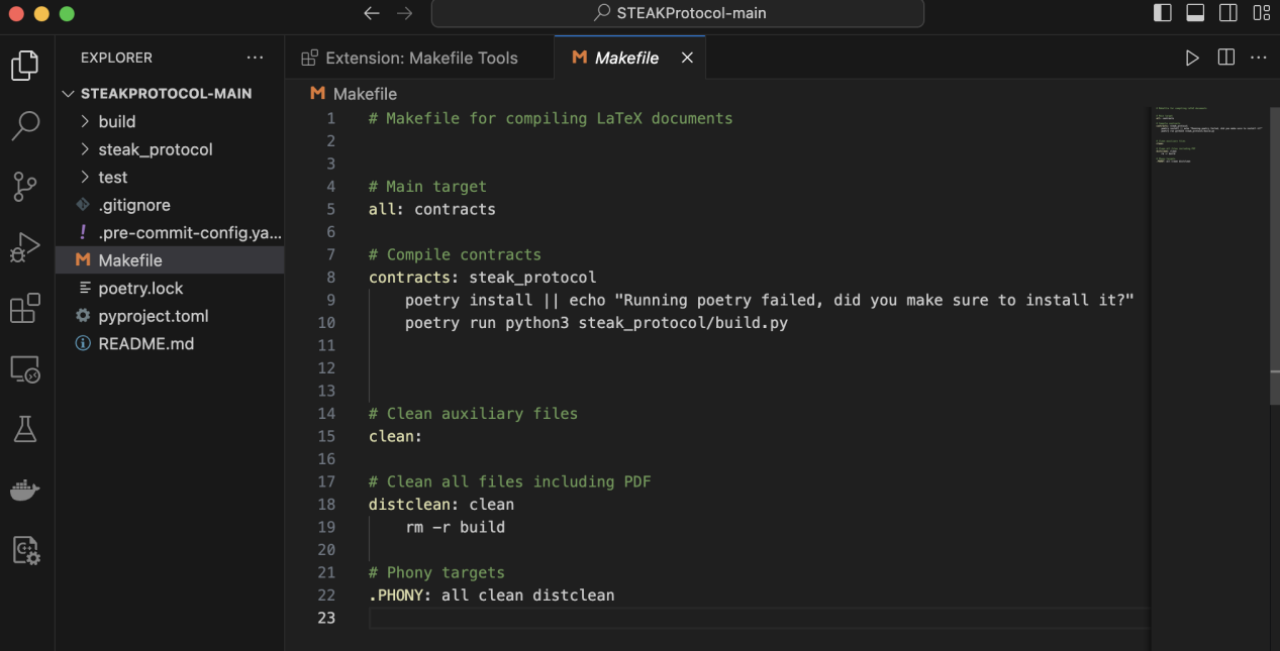
### **#### Building the contracts**

To reproduce the contract addresses, you can build the contracts using the following steps:

1. Install [poetry](https://python-poetry.org/docs/#installation) and [aiken v1.0.26-alpha+075668b

](https://aiken-lang.org/installation-instructions).

2. Run `make` in the root of the directory

****

### **-Output**

### **#### Test Result**

# This file is automatically @generated by Poetry 1.8.2 and should not be changed by hand.

[[package]]

name = "aiohttp"

version = "3.9.5"

description = "Async http client/server framework (asyncio)"

optional = false

python-versions = ">=3.8"

files = [

{file = "aiohttp-3.9.5-cp310-cp310-macosx\_10\_9\_universal2.whl", hash = "sha256:fcde4c397f673fdec23e6b05ebf8d4751314fa7c24f93334bf1f1364c1c69ac7"},

{file = "aiohttp-3.9.5-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:5d6b3f1fabe465e819aed2c421a6743d8debbde79b6a8600739300630a01bf2c"},

{file = "aiohttp-3.9.5-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:6ae79c1bc12c34082d92bf9422764f799aee4746fd7a392db46b7fd357d4a17a"},

{file = "aiohttp-3.9.5-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:4d3ebb9e1316ec74277d19c5f482f98cc65a73ccd5430540d6d11682cd857430"},

{file = "aiohttp-3.9.5-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:84dabd95154f43a2ea80deffec9cb44d2e301e38a0c9d331cc4aa0166fe28ae3"},

{file = "aiohttp-3.9.5-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:c8a02fbeca6f63cb1f0475c799679057fc9268b77075ab7cf3f1c600e81dd46b"},

{file = "aiohttp-3.9.5-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:c26959ca7b75ff768e2776d8055bf9582a6267e24556bb7f7bd29e677932be72"},

{file = "aiohttp-3.9.5-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:714d4e5231fed4ba2762ed489b4aec07b2b9953cf4ee31e9871caac895a839c0"},

{file = "aiohttp-3.9.5-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:e7a6a8354f1b62e15d48e04350f13e726fa08b62c3d7b8401c0a1314f02e3558"},

{file = "aiohttp-3.9.5-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:c413016880e03e69d166efb5a1a95d40f83d5a3a648d16486592c49ffb76d0db"},

{file = "aiohttp-3.9.5-cp310-cp310-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:ff84aeb864e0fac81f676be9f4685f0527b660f1efdc40dcede3c251ef1e867f"},

{file = "aiohttp-3.9.5-cp310-cp310-musllinux\_1\_1\_s390x.whl", hash = "sha256:ad7f2919d7dac062f24d6f5fe95d401597fbb015a25771f85e692d043c9d7832"},

{file = "aiohttp-3.9.5-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:702e2c7c187c1a498a4e2b03155d52658fdd6fda882d3d7fbb891a5cf108bb10"},

{file = "aiohttp-3.9.5-cp310-cp310-win32.whl", hash = "sha256:67c3119f5ddc7261d47163ed86d760ddf0e625cd6246b4ed852e82159617b5fb"},

{file = "aiohttp-3.9.5-cp310-cp310-win\_amd64.whl", hash = "sha256:471f0ef53ccedec9995287f02caf0c068732f026455f07db3f01a46e49d76bbb"},

{file = "aiohttp-3.9.5-cp311-cp311-macosx\_10\_9\_universal2.whl", hash = "sha256:e0ae53e33ee7476dd3d1132f932eeb39bf6125083820049d06edcdca4381f342"},

{file = "aiohttp-3.9.5-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:c088c4d70d21f8ca5c0b8b5403fe84a7bc8e024161febdd4ef04575ef35d474d"},

{file = "aiohttp-3.9.5-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:639d0042b7670222f33b0028de6b4e2fad6451462ce7df2af8aee37dcac55424"},

{file = "aiohttp-3.9.5-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:f26383adb94da5e7fb388d441bf09c61e5e35f455a3217bfd790c6b6bc64b2ee"},

{file = "aiohttp-3.9.5-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:66331d00fb28dc90aa606d9a54304af76b335ae204d1836f65797d6fe27f1ca2"},

{file = "aiohttp-3.9.5-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:4ff550491f5492ab5ed3533e76b8567f4b37bd2995e780a1f46bca2024223233"},

{file = "aiohttp-3.9.5-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:f22eb3a6c1080d862befa0a89c380b4dafce29dc6cd56083f630073d102eb595"},

{file = "aiohttp-3.9.5-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:a81b1143d42b66ffc40a441379387076243ef7b51019204fd3ec36b9f69e77d6"},

{file = "aiohttp-3.9.5-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:f64fd07515dad67f24b6ea4a66ae2876c01031de91c93075b8093f07c0a2d93d"},

{file = "aiohttp-3.9.5-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:93e22add827447d2e26d67c9ac0161756007f152fdc5210277d00a85f6c92323"},

{file = "aiohttp-3.9.5-cp311-cp311-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:55b39c8684a46e56ef8c8d24faf02de4a2b2ac60d26cee93bc595651ff545de9"},

{file = "aiohttp-3.9.5-cp311-cp311-musllinux\_1\_1\_s390x.whl", hash = "sha256:4715a9b778f4293b9f8ae7a0a7cef9829f02ff8d6277a39d7f40565c737d3771"},

{file = "aiohttp-3.9.5-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:afc52b8d969eff14e069a710057d15ab9ac17cd4b6753042c407dcea0e40bf75"},

{file = "aiohttp-3.9.5-cp311-cp311-win32.whl", hash = "sha256:b3df71da99c98534be076196791adca8819761f0bf6e08e07fd7da25127150d6"},

{file = "aiohttp-3.9.5-cp311-cp311-win\_amd64.whl", hash = "sha256:88e311d98cc0bf45b62fc46c66753a83445f5ab20038bcc1b8a1cc05666f428a"},

{file = "aiohttp-3.9.5-cp312-cp312-macosx\_10\_9\_universal2.whl", hash = "sha256:c7a4b7a6cf5b6eb11e109a9755fd4fda7d57395f8c575e166d363b9fc3ec4678"},

{file = "aiohttp-3.9.5-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:0a158704edf0abcac8ac371fbb54044f3270bdbc93e254a82b6c82be1ef08f3c"},

{file = "aiohttp-3.9.5-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:d153f652a687a8e95ad367a86a61e8d53d528b0530ef382ec5aaf533140ed00f"},

{file = "aiohttp-3.9.5-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:82a6a97d9771cb48ae16979c3a3a9a18b600a8505b1115cfe354dfb2054468b4"},

{file = "aiohttp-3.9.5-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:60cdbd56f4cad9f69c35eaac0fbbdf1f77b0ff9456cebd4902f3dd1cf096464c"},

{file = "aiohttp-3.9.5-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:8676e8fd73141ded15ea586de0b7cda1542960a7b9ad89b2b06428e97125d4fa"},

{file = "aiohttp-3.9.5-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:da00da442a0e31f1c69d26d224e1efd3a1ca5bcbf210978a2ca7426dfcae9f58"},

{file = "aiohttp-3.9.5-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:18f634d540dd099c262e9f887c8bbacc959847cfe5da7a0e2e1cf3f14dbf2daf"},

{file = "aiohttp-3.9.5-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:320e8618eda64e19d11bdb3bd04ccc0a816c17eaecb7e4945d01deee2a22f95f"},

{file = "aiohttp-3.9.5-cp312-cp312-musllinux\_1\_1\_i686.whl", hash = "sha256:2faa61a904b83142747fc6a6d7ad8fccff898c849123030f8e75d5d967fd4a81"},

{file = "aiohttp-3.9.5-cp312-cp312-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:8c64a6dc3fe5db7b1b4d2b5cb84c4f677768bdc340611eca673afb7cf416ef5a"},

{file = "aiohttp-3.9.5-cp312-cp312-musllinux\_1\_1\_s390x.whl", hash = "sha256:393c7aba2b55559ef7ab791c94b44f7482a07bf7640d17b341b79081f5e5cd1a"},

{file = "aiohttp-3.9.5-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:c671dc117c2c21a1ca10c116cfcd6e3e44da7fcde37bf83b2be485ab377b25da"},

{file = "aiohttp-3.9.5-cp312-cp312-win32.whl", hash = "sha256:5a7ee16aab26e76add4afc45e8f8206c95d1d75540f1039b84a03c3b3800dd59"},

{file = "aiohttp-3.9.5-cp312-cp312-win\_amd64.whl", hash = "sha256:5ca51eadbd67045396bc92a4345d1790b7301c14d1848feaac1d6a6c9289e888"},

{file = "aiohttp-3.9.5-cp38-cp38-macosx\_10\_9\_universal2.whl", hash = "sha256:694d828b5c41255e54bc2dddb51a9f5150b4eefa9886e38b52605a05d96566e8"},

{file = "aiohttp-3.9.5-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:0605cc2c0088fcaae79f01c913a38611ad09ba68ff482402d3410bf59039bfb8"},

{file = "aiohttp-3.9.5-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:4558e5012ee03d2638c681e156461d37b7a113fe13970d438d95d10173d25f78"},

{file = "aiohttp-3.9.5-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:9dbc053ac75ccc63dc3a3cc547b98c7258ec35a215a92bd9f983e0aac95d3d5b"},

{file = "aiohttp-3.9.5-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:4109adee842b90671f1b689901b948f347325045c15f46b39797ae1bf17019de"},

{file = "aiohttp-3.9.5-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:a6ea1a5b409a85477fd8e5ee6ad8f0e40bf2844c270955e09360418cfd09abac"},

{file = "aiohttp-3.9.5-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:f3c2890ca8c59ee683fd09adf32321a40fe1cf164e3387799efb2acebf090c11"},

{file = "aiohttp-3.9.5-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:3916c8692dbd9d55c523374a3b8213e628424d19116ac4308e434dbf6d95bbdd"},

{file = "aiohttp-3.9.5-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:8d1964eb7617907c792ca00b341b5ec3e01ae8c280825deadbbd678447b127e1"},

{file = "aiohttp-3.9.5-cp38-cp38-musllinux\_1\_1\_i686.whl", hash = "sha256:d5ab8e1f6bee051a4bf6195e38a5c13e5e161cb7bad83d8854524798bd9fcd6e"},

{file = "aiohttp-3.9.5-cp38-cp38-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:52c27110f3862a1afbcb2af4281fc9fdc40327fa286c4625dfee247c3ba90156"},

{file = "aiohttp-3.9.5-cp38-cp38-musllinux\_1\_1\_s390x.whl", hash = "sha256:7f64cbd44443e80094309875d4f9c71d0401e966d191c3d469cde4642bc2e031"},

{file = "aiohttp-3.9.5-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:8b4f72fbb66279624bfe83fd5eb6aea0022dad8eec62b71e7bf63ee1caadeafe"},

{file = "aiohttp-3.9.5-cp38-cp38-win32.whl", hash = "sha256:6380c039ec52866c06d69b5c7aad5478b24ed11696f0e72f6b807cfb261453da"},

{file = "aiohttp-3.9.5-cp38-cp38-win\_amd64.whl", hash = "sha256:da22dab31d7180f8c3ac7c7635f3bcd53808f374f6aa333fe0b0b9e14b01f91a"},

{file = "aiohttp-3.9.5-cp39-cp39-macosx\_10\_9\_universal2.whl", hash = "sha256:1732102949ff6087589408d76cd6dea656b93c896b011ecafff418c9661dc4ed"},

{file = "aiohttp-3.9.5-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:c6021d296318cb6f9414b48e6a439a7f5d1f665464da507e8ff640848ee2a58a"},

{file = "aiohttp-3.9.5-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:239f975589a944eeb1bad26b8b140a59a3a320067fb3cd10b75c3092405a1372"},

{file = "aiohttp-3.9.5-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:3b7b30258348082826d274504fbc7c849959f1989d86c29bc355107accec6cfb"},

{file = "aiohttp-3.9.5-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:cd2adf5c87ff6d8b277814a28a535b59e20bfea40a101db6b3bdca7e9926bc24"},

{file = "aiohttp-3.9.5-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:e9a3d838441bebcf5cf442700e3963f58b5c33f015341f9ea86dcd7d503c07e2"},

{file = "aiohttp-3.9.5-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:9e3a1ae66e3d0c17cf65c08968a5ee3180c5a95920ec2731f53343fac9bad106"},

{file = "aiohttp-3.9.5-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:9c69e77370cce2d6df5d12b4e12bdcca60c47ba13d1cbbc8645dd005a20b738b"},

{file = "aiohttp-3.9.5-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:0cbf56238f4bbf49dab8c2dc2e6b1b68502b1e88d335bea59b3f5b9f4c001475"},

{file = "aiohttp-3.9.5-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:d1469f228cd9ffddd396d9948b8c9cd8022b6d1bf1e40c6f25b0fb90b4f893ed"},

{file = "aiohttp-3.9.5-cp39-cp39-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:45731330e754f5811c314901cebdf19dd776a44b31927fa4b4dbecab9e457b0c"},

{file = "aiohttp-3.9.5-cp39-cp39-musllinux\_1\_1\_s390x.whl", hash = "sha256:3fcb4046d2904378e3aeea1df51f697b0467f2aac55d232c87ba162709478c46"},

{file = "aiohttp-3.9.5-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:8cf142aa6c1a751fcb364158fd710b8a9be874b81889c2bd13aa8893197455e2"},

{file = "aiohttp-3.9.5-cp39-cp39-win32.whl", hash = "sha256:7b179eea70833c8dee51ec42f3b4097bd6370892fa93f510f76762105568cf09"},

{file = "aiohttp-3.9.5-cp39-cp39-win\_amd64.whl", hash = "sha256:38d80498e2e169bc61418ff36170e0aad0cd268da8b38a17c4cf29d254a8b3f1"},

{file = "aiohttp-3.9.5.tar.gz", hash = "sha256:edea7d15772ceeb29db4aff55e482d4bcfb6ae160ce144f2682de02f6d693551"},

]

[package.dependencies]

aiosignal = ">=1.1.2"

attrs = ">=17.3.0"

frozenlist = ">=1.1.1"

multidict = ">=4.5,<7.0"

yarl = ">=1.0,<2.0"

[package.extras]

speedups = ["Brotli", "aiodns", "brotlicffi"]

[[package]]

name = "aiosignal"

version = "1.3.1"

description = "aiosignal: a list of registered asynchronous callbacks"

optional = false

python-versions = ">=3.7"

files = [

{file = "aiosignal-1.3.1-py3-none-any.whl", hash = "sha256:f8376fb07dd1e86a584e4fcdec80b36b7f81aac666ebc724e2c090300dd83b17"},

{file = "aiosignal-1.3.1.tar.gz", hash = "sha256:54cd96e15e1649b75d6c87526a6ff0b6c1b0dd3459f43d9ca11d48c339b68cfc"},

]

[package.dependencies]

frozenlist = ">=1.1.0"

[[package]]

name = "annotated-types"

version = "0.6.0"

description = "Reusable constraint types to use with typing.Annotated"

optional = false

python-versions = ">=3.8"

files = [

{file = "annotated\_types-0.6.0-py3-none-any.whl", hash = "sha256:0641064de18ba7a25dee8f96403ebc39113d0cb953a01429249d5c7564666a43"},

{file = "annotated\_types-0.6.0.tar.gz", hash = "sha256:563339e807e53ffd9c267e99fc6d9ea23eb8443c08f112651963e24e22f84a5d"},

]

[[package]]

name = "anyio"

version = "4.3.0"

description = "High level compatibility layer for multiple asynchronous event loop implementations"

optional = false

python-versions = ">=3.8"

files = [

{file = "anyio-4.3.0-py3-none-any.whl", hash = "sha256:048e05d0f6caeed70d731f3db756d35dcc1f35747c8c403364a8332c630441b8"},

{file = "anyio-4.3.0.tar.gz", hash = "sha256:f75253795a87df48568485fd18cdd2a3fa5c4f7c5be8e5e36637733fce06fed6"},

]

[package.dependencies]

exceptiongroup = {version = ">=1.0.2", markers = "python\_version < \"3.11\""}

idna = ">=2.8"

sniffio = ">=1.1"

typing-extensions = {version = ">=4.1", markers = "python\_version < \"3.11\""}

[package.extras]

doc = ["Sphinx (>=7)", "packaging", "sphinx-autodoc-typehints (>=1.2.0)", "sphinx-rtd-theme"]

test = ["anyio[trio]", "coverage[toml] (>=7)", "exceptiongroup (>=1.2.0)", "hypothesis (>=4.0)", "psutil (>=5.9)", "pytest (>=7.0)", "pytest-mock (>=3.6.1)", "trustme", "uvloop (>=0.17)"]

trio = ["trio (>=0.23)"]

[[package]]

name = "appdirs"

version = "1.4.4"

description = "A small Python module for determining appropriate platform-specific dirs, e.g. a \"user data dir\"."

optional = false

python-versions = "\*"

files = [

{file = "appdirs-1.4.4-py2.py3-none-any.whl", hash = "sha256:a841dacd6b99318a741b166adb07e19ee71a274450e68237b4650ca1055ab128"},

{file = "appdirs-1.4.4.tar.gz", hash = "sha256:7d5d0167b2b1ba821647616af46a749d1c653740dd0d2415100fe26e27afdf41"},

]

[[package]]

name = "asn1crypto"

version = "1.5.1"

description = "Fast ASN.1 parser and serializer with definitions for private keys, public keys, certificates, CRL, OCSP, CMS, PKCS#3, PKCS#7, PKCS#8, PKCS#12, PKCS#5, X.509 and TSP"

optional = false

python-versions = "\*"

files = [

{file = "asn1crypto-1.5.1-py2.py3-none-any.whl", hash = "sha256:db4e40728b728508912cbb3d44f19ce188f218e9eba635821bb4b68564f8fd67"},

{file = "asn1crypto-1.5.1.tar.gz", hash = "sha256:13ae38502be632115abf8a24cbe5f4da52e3b5231990aff31123c805306ccb9c"},

]

[[package]]

name = "attrs"

version = "23.2.0"

description = "Classes Without Boilerplate"

optional = false

python-versions = ">=3.7"

files = [

{file = "attrs-23.2.0-py3-none-any.whl", hash = "sha256:99b87a485a5820b23b879f04c2305b44b951b502fd64be915879d77a7e8fc6f1"},

{file = "attrs-23.2.0.tar.gz", hash = "sha256:935dc3b529c262f6cf76e50877d35a4bd3c1de194fd41f47a2b7ae8f19971f30"},

]

[package.extras]

cov = ["attrs[tests]", "coverage[toml] (>=5.3)"]

dev = ["attrs[tests]", "pre-commit"]

docs = ["furo", "myst-parser", "sphinx", "sphinx-notfound-page", "sphinxcontrib-towncrier", "towncrier", "zope-interface"]

tests = ["attrs[tests-no-zope]", "zope-interface"]

tests-mypy = ["mypy (>=1.6)", "pytest-mypy-plugins"]

tests-no-zope = ["attrs[tests-mypy]", "cloudpickle", "hypothesis", "pympler", "pytest (>=4.3.0)", "pytest-xdist[psutil]"]

[[package]]

name = "black"

version = "23.12.1"

description = "The uncompromising code formatter."

optional = false

python-versions = ">=3.8"

files = [

{file = "black-23.12.1-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:e0aaf6041986767a5e0ce663c7a2f0e9eaf21e6ff87a5f95cbf3675bfd4c41d2"},

{file = "black-23.12.1-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:c88b3711d12905b74206227109272673edce0cb29f27e1385f33b0163c414bba"},

{file = "black-23.12.1-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:a920b569dc6b3472513ba6ddea21f440d4b4c699494d2e972a1753cdc25df7b0"},

{file = "black-23.12.1-cp310-cp310-win\_amd64.whl", hash = "sha256:3fa4be75ef2a6b96ea8d92b1587dd8cb3a35c7e3d51f0738ced0781c3aa3a5a3"},

{file = "black-23.12.1-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:8d4df77958a622f9b5a4c96edb4b8c0034f8434032ab11077ec6c56ae9f384ba"},

{file = "black-23.12.1-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:602cfb1196dc692424c70b6507593a2b29aac0547c1be9a1d1365f0d964c353b"},

{file = "black-23.12.1-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:9c4352800f14be5b4864016882cdba10755bd50805c95f728011bcb47a4afd59"},

{file = "black-23.12.1-cp311-cp311-win\_amd64.whl", hash = "sha256:0808494f2b2df923ffc5723ed3c7b096bd76341f6213989759287611e9837d50"},

{file = "black-23.12.1-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:25e57fd232a6d6ff3f4478a6fd0580838e47c93c83eaf1ccc92d4faf27112c4e"},

{file = "black-23.12.1-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:2d9e13db441c509a3763a7a3d9a49ccc1b4e974a47be4e08ade2a228876500ec"},

{file = "black-23.12.1-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:6d1bd9c210f8b109b1762ec9fd36592fdd528485aadb3f5849b2740ef17e674e"},

{file = "black-23.12.1-cp312-cp312-win\_amd64.whl", hash = "sha256:ae76c22bde5cbb6bfd211ec343ded2163bba7883c7bc77f6b756a1049436fbb9"},

{file = "black-23.12.1-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:1fa88a0f74e50e4487477bc0bb900c6781dbddfdfa32691e780bf854c3b4a47f"},

{file = "black-23.12.1-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:a4d6a9668e45ad99d2f8ec70d5c8c04ef4f32f648ef39048d010b0689832ec6d"},

{file = "black-23.12.1-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:b18fb2ae6c4bb63eebe5be6bd869ba2f14fd0259bda7d18a46b764d8fb86298a"},

{file = "black-23.12.1-cp38-cp38-win\_amd64.whl", hash = "sha256:c04b6d9d20e9c13f43eee8ea87d44156b8505ca8a3c878773f68b4e4812a421e"},

{file = "black-23.12.1-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:3e1b38b3135fd4c025c28c55ddfc236b05af657828a8a6abe5deec419a0b7055"},

{file = "black-23.12.1-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:4f0031eaa7b921db76decd73636ef3a12c942ed367d8c3841a0739412b260a54"},

{file = "black-23.12.1-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:97e56155c6b737854e60a9ab1c598ff2533d57e7506d97af5481141671abf3ea"},

{file = "black-23.12.1-cp39-cp39-win\_amd64.whl", hash = "sha256:dd15245c8b68fe2b6bd0f32c1556509d11bb33aec9b5d0866dd8e2ed3dba09c2"},

{file = "black-23.12.1-py3-none-any.whl", hash = "sha256:78baad24af0f033958cad29731e27363183e140962595def56423e626f4bee3e"},

{file = "black-23.12.1.tar.gz", hash = "sha256:4ce3ef14ebe8d9509188014d96af1c456a910d5b5cbf434a09fef7e024b3d0d5"},

]

[package.dependencies]

click = ">=8.0.0"

mypy-extensions = ">=0.4.3"

packaging = ">=22.0"

pathspec = ">=0.9.0"

platformdirs = ">=2"

tomli = {version = ">=1.1.0", markers = "python\_version < \"3.11\""}

typing-extensions = {version = ">=4.0.1", markers = "python\_version < \"3.11\""}

[package.extras]

colorama = ["colorama (>=0.4.3)"]

d = ["aiohttp (>=3.7.4)", "aiohttp (>=3.7.4,!=3.9.0)"]

jupyter = ["ipython (>=7.8.0)", "tokenize-rt (>=3.2.0)"]

uvloop = ["uvloop (>=0.15.2)"]

[[package]]

name = "blockfrost-python"

version = "0.5.3"

description = "The official Python SDK for Blockfrost API v0.1.37"

optional = false

python-versions = ">=3.7, <4"

files = [

{file = "blockfrost-python-0.5.3.tar.gz", hash = "sha256:3154b99867e7714c90064c9e1a37e3b7af97c107b64549dd0d424aaa3209017e"},

{file = "blockfrost\_python-0.5.3-py3-none-any.whl", hash = "sha256:b0e73f09f1ff06977c85ccd63f6afe7ec30fa1b5c48e94a15d8bc8cf1f61997b"},

]

[package.dependencies]

requests = "\*"

[[package]]

name = "cachetools"

version = "5.3.3"

description = "Extensible memoizing collections and decorators"

optional = false

python-versions = ">=3.7"

files = [

{file = "cachetools-5.3.3-py3-none-any.whl", hash = "sha256:0abad1021d3f8325b2fc1d2e9c8b9c9d57b04c3932657a72465447332c24d945"},

{file = "cachetools-5.3.3.tar.gz", hash = "sha256:ba29e2dfa0b8b556606f097407ed1aa62080ee108ab0dc5ec9d6a723a007d105"},

]

[[package]]

name = "cardano-tools"

version = "2.1.0"

description = "A collection of tools to enable development in the Cardano ecosystem using the Python programming language."

optional = false

python-versions = ">=3.8,<4.0"

files = [

{file = "cardano\_tools-2.1.0-py3-none-any.whl", hash = "sha256:c562c234b3d9a51540d41432f88aeae9a68c9441e0ea363c675dd712380ca06c"},

{file = "cardano\_tools-2.1.0.tar.gz", hash = "sha256:445c8a5c769f57e5e04494ac4e3012082c3d0f1bd9a9eaed7f834d37ad7a069e"},

]

[package.dependencies]

pexpect = ">=4.8.0,<5.0.0"

requests = ">=2.28.0,<3.0.0"

[[package]]

name = "cbor2"

version = "5.6.3"

description = "CBOR (de)serializer with extensive tag support"

optional = false

python-versions = ">=3.8"

files = [

{file = "cbor2-5.6.3-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:0201d5e8d9ad1557aeb50d35b907c0f170de0ae9ebb484b2894bcee3b2e13b80"},

{file = "cbor2-5.6.3-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:eda6965cca276d4c2cebdbee14572dec65b991c5359fc32a793f03f052e35985"},

{file = "cbor2-5.6.3-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:14561038b8eaab3fd5e867f09bc43f7525a1405e41ade14066925ea3d42513a8"},

{file = "cbor2-5.6.3-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:9a3cf6b339a005031e4b8c79b9541856e3b0077ea4c33d7bb6a019885136f53a"},

{file = "cbor2-5.6.3-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:4b7636d39de203ee30ac13575ed3e9a0510e993fa1671022b84b9e35e369825f"},

{file = "cbor2-5.6.3-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:23847075ce1bcda871c7698e5db0635685995ae470098a5e4c9a26c00f65f21a"},

{file = "cbor2-5.6.3-cp310-cp310-win\_amd64.whl", hash = "sha256:ca15be7142e861fb9f918e0248620b4d4153b9ff14ef6034f7204db5db2924a1"},

{file = "cbor2-5.6.3-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:b07ee755ae5b0dfad608dab37364b35895cab5d1222653da1fea32a10330c4b0"},

{file = "cbor2-5.6.3-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:9fc063843c14e9e95181faf8d807a53c958d77bb9d360eb4f2344d075ecfed36"},

{file = "cbor2-5.6.3-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:c66d4c227c2ed6c63ec5c2d50eb8ec0e1c41c07b452a867544e48ca41d4f0b64"},

{file = "cbor2-5.6.3-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:3af60ac82a733bfdfb2b1079c850fefea2621bdb8c8f87f4c5d12802d48a8c55"},

{file = "cbor2-5.6.3-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:acb93292843aa72768f089a135bfeec4c9b745132e8dc22f1b149490fc77cb0a"},

{file = "cbor2-5.6.3-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:193d1abdffd52893710d39389daa5c03e1569421cdf53585a28033689aef7aec"},

{file = "cbor2-5.6.3-cp311-cp311-win\_amd64.whl", hash = "sha256:d0c915db92b441f505f8a14a521c9461439ac8e5d959454845eb92f93db0bb3b"},

{file = "cbor2-5.6.3-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:9eaec8c04618124a6b597fe4471035cb7cb0d5114f43aaf2062821ad480ef57c"},

{file = "cbor2-5.6.3-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:d1e5181d4f858237ab4e1a28e21bdcaf31dab2657ab60a8d4a0701a078fe5926"},

{file = "cbor2-5.6.3-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:277997127402710a3abdf4372ac75e8f8bb2e75a303cd789312e515c8ef657dd"},

{file = "cbor2-5.6.3-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:add01e4b4663199940d10f8c8e1d926e70823d1b2f3f981cc097a4764125f110"},

{file = "cbor2-5.6.3-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:adc87485ffd7a4dad481e08e6819eebfcfbafc0918fffcca47aee4cdf8c6de04"},

{file = "cbor2-5.6.3-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:ea4a0412426155c3b78763449db56cf5c72c48788a37d7e60bd66c844b9c8634"},

{file = "cbor2-5.6.3-cp312-cp312-win\_amd64.whl", hash = "sha256:18b3dee4eddde9761c60298ce21c0cd4e770237978034c5ee1d4242e255683ec"},

{file = "cbor2-5.6.3-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:ff6fd1c54b97ee322c0b7180092305ca3b012ff78fddadad97b33490f5f8881f"},

{file = "cbor2-5.6.3-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:ac6f10b9d25f2d61c036f86238bf23e3ea0253f98faa8ab00f67228bf3c0ce2a"},

{file = "cbor2-5.6.3-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:7be74f2cbda547fdd57c83ee5b3470804f02c660db28efcf9d4016f001b66f40"},

{file = "cbor2-5.6.3-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:64ea120206f82492a4385bbc5e2639f9b67c8bc7bdc57bffcbe9a8fee8cd6342"},

{file = "cbor2-5.6.3-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:c3d2902e1aed155d56cdcae99cd4a9dae843e3fff6978148d2d5d5f9a0b986cd"},

{file = "cbor2-5.6.3-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:2d4f95a567e26d8d9d62db234cd089525c52f19e7fdd59152629d9f03bd94b4f"},

{file = "cbor2-5.6.3-cp38-cp38-win\_amd64.whl", hash = "sha256:33efbe7103bac090430d291fca2fe1c444b0ec55c4716e8051b72a81377e8b79"},

{file = "cbor2-5.6.3-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:81e619a2a59ae966cedb5fd3ea8a9487a3d4430824bbeacdcf5f74ad6112cc57"},

{file = "cbor2-5.6.3-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:2b7755b93d32638f4d79a0fa0744b423787f6faa3c96ccccac68b6dbf1848368"},

{file = "cbor2-5.6.3-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:0f0e95011ae8460265ef348fe380664fa22c51015fd52344ebd781579fa9552a"},

{file = "cbor2-5.6.3-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:7693e53c3ba0b2ad4e46b610f8d69159ffdbcb6ebe75ea1c1f5f40c3283639ca"},

{file = "cbor2-5.6.3-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:e3ec251db32516d383fc587874b15f4b5fb4e9049d9436b8696f5767b11c149b"},

{file = "cbor2-5.6.3-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:6081c1ab9791d5973a40b95ecb8b04b0fbf9fc04be170d89a3ad77d5964f52d5"},

{file = "cbor2-5.6.3-cp39-cp39-win\_amd64.whl", hash = "sha256:2aba8b75e36c9f84a42a7026271da8fd759035a871c1b799028439059527276b"},

{file = "cbor2-5.6.3-py3-none-any.whl", hash = "sha256:8a4b7404af6da719092a4ee5953d1930d095b93b684bf99e1ab74512be1910a4"},

{file = "cbor2-5.6.3.tar.gz", hash = "sha256:e6f0ae2751c2d333a960e0807c0611494eb1245631a167965acbc100509455d3"},

]

[package.extras]

benchmarks = ["pytest-benchmark (==4.0.0)"]

doc = ["Sphinx (>=7)", "packaging", "sphinx-autodoc-typehints (>=1.2.0)", "sphinx-rtd-theme (>=1.3.0)", "typing-extensions"]

test = ["coverage (>=7)", "hypothesis", "pytest"]

[[package]]

name = "certifi"

version = "2024.2.2"

description = "Python package for providing Mozilla's CA Bundle."

optional = false

python-versions = ">=3.6"

files = [

{file = "certifi-2024.2.2-py3-none-any.whl", hash = "sha256:dc383c07b76109f368f6106eee2b593b04a011ea4d55f652c6ca24a754d1cdd1"},

{file = "certifi-2024.2.2.tar.gz", hash = "sha256:0569859f95fc761b18b45ef421b1290a0f65f147e92a1e5eb3e635f9a5e4e66f"},

]

[[package]]

name = "certvalidator"

version = "0.11.1"

description = "Validates X.509 certificates and paths"

optional = false

python-versions = "\*"

files = [

{file = "certvalidator-0.11.1-py2.py3-none-any.whl", hash = "sha256:77520b269f516d4fb0902998d5bd0eb3727fe153b659aa1cb828dcf12ea6b8de"},

{file = "certvalidator-0.11.1.tar.gz", hash = "sha256:922d141c94393ab285ca34338e18dd4093e3ae330b1f278e96c837cb62cffaad"},

]

[package.dependencies]

asn1crypto = ">=0.18.1"

oscrypto = ">=0.16.1"

[[package]]

name = "cffi"

version = "1.16.0"

description = "Foreign Function Interface for Python calling C code."

optional = false

python-versions = ">=3.8"

files = [

{file = "cffi-1.16.0-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:6b3d6606d369fc1da4fd8c357d026317fbb9c9b75d36dc16e90e84c26854b088"},

{file = "cffi-1.16.0-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:ac0f5edd2360eea2f1daa9e26a41db02dd4b0451b48f7c318e217ee092a213e9"},

{file = "cffi-1.16.0-cp310-cp310-manylinux\_2\_12\_i686.manylinux2010\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:7e61e3e4fa664a8588aa25c883eab612a188c725755afff6289454d6362b9673"},

{file = "cffi-1.16.0-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a72e8961a86d19bdb45851d8f1f08b041ea37d2bd8d4fd19903bc3083d80c896"},

{file = "cffi-1.16.0-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:5b50bf3f55561dac5438f8e70bfcdfd74543fd60df5fa5f62d94e5867deca684"},

{file = "cffi-1.16.0-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:7651c50c8c5ef7bdb41108b7b8c5a83013bfaa8a935590c5d74627c047a583c7"},

{file = "cffi-1.16.0-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:e4108df7fe9b707191e55f33efbcb2d81928e10cea45527879a4749cbe472614"},

{file = "cffi-1.16.0-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:32c68ef735dbe5857c810328cb2481e24722a59a2003018885514d4c09af9743"},

{file = "cffi-1.16.0-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:673739cb539f8cdaa07d92d02efa93c9ccf87e345b9a0b556e3ecc666718468d"},

{file = "cffi-1.16.0-cp310-cp310-win32.whl", hash = "sha256:9f90389693731ff1f659e55c7d1640e2ec43ff725cc61b04b2f9c6d8d017df6a"},

{file = "cffi-1.16.0-cp310-cp310-win\_amd64.whl", hash = "sha256:e6024675e67af929088fda399b2094574609396b1decb609c55fa58b028a32a1"},

{file = "cffi-1.16.0-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:b84834d0cf97e7d27dd5b7f3aca7b6e9263c56308ab9dc8aae9784abb774d404"},

{file = "cffi-1.16.0-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:1b8ebc27c014c59692bb2664c7d13ce7a6e9a629be20e54e7271fa696ff2b417"},

{file = "cffi-1.16.0-cp311-cp311-manylinux\_2\_12\_i686.manylinux2010\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:ee07e47c12890ef248766a6e55bd38ebfb2bb8edd4142d56db91b21ea68b7627"},

{file = "cffi-1.16.0-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:d8a9d3ebe49f084ad71f9269834ceccbf398253c9fac910c4fd7053ff1386936"},

{file = "cffi-1.16.0-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:e70f54f1796669ef691ca07d046cd81a29cb4deb1e5f942003f401c0c4a2695d"},

{file = "cffi-1.16.0-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:5bf44d66cdf9e893637896c7faa22298baebcd18d1ddb6d2626a6e39793a1d56"},

{file = "cffi-1.16.0-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:7b78010e7b97fef4bee1e896df8a4bbb6712b7f05b7ef630f9d1da00f6444d2e"},

{file = "cffi-1.16.0-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:c6a164aa47843fb1b01e941d385aab7215563bb8816d80ff3a363a9f8448a8dc"},

{file = "cffi-1.16.0-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:e09f3ff613345df5e8c3667da1d918f9149bd623cd9070c983c013792a9a62eb"},

{file = "cffi-1.16.0-cp311-cp311-win32.whl", hash = "sha256:2c56b361916f390cd758a57f2e16233eb4f64bcbeee88a4881ea90fca14dc6ab"},

{file = "cffi-1.16.0-cp311-cp311-win\_amd64.whl", hash = "sha256:db8e577c19c0fda0beb7e0d4e09e0ba74b1e4c092e0e40bfa12fe05b6f6d75ba"},

{file = "cffi-1.16.0-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:fa3a0128b152627161ce47201262d3140edb5a5c3da88d73a1b790a959126956"},

{file = "cffi-1.16.0-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:68e7c44931cc171c54ccb702482e9fc723192e88d25a0e133edd7aff8fcd1f6e"},

{file = "cffi-1.16.0-cp312-cp312-manylinux\_2\_12\_i686.manylinux2010\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:abd808f9c129ba2beda4cfc53bde801e5bcf9d6e0f22f095e45327c038bfe68e"},

{file = "cffi-1.16.0-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:88e2b3c14bdb32e440be531ade29d3c50a1a59cd4e51b1dd8b0865c54ea5d2e2"},

{file = "cffi-1.16.0-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:fcc8eb6d5902bb1cf6dc4f187ee3ea80a1eba0a89aba40a5cb20a5087d961357"},

{file = "cffi-1.16.0-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:b7be2d771cdba2942e13215c4e340bfd76398e9227ad10402a8767ab1865d2e6"},

{file = "cffi-1.16.0-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:e715596e683d2ce000574bae5d07bd522c781a822866c20495e52520564f0969"},

{file = "cffi-1.16.0-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:2d92b25dbf6cae33f65005baf472d2c245c050b1ce709cc4588cdcdd5495b520"},

{file = "cffi-1.16.0-cp312-cp312-win32.whl", hash = "sha256:b2ca4e77f9f47c55c194982e10f058db063937845bb2b7a86c84a6cfe0aefa8b"},

{file = "cffi-1.16.0-cp312-cp312-win\_amd64.whl", hash = "sha256:68678abf380b42ce21a5f2abde8efee05c114c2fdb2e9eef2efdb0257fba1235"},

{file = "cffi-1.16.0-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:0c9ef6ff37e974b73c25eecc13952c55bceed9112be2d9d938ded8e856138bcc"},

{file = "cffi-1.16.0-cp38-cp38-manylinux\_2\_12\_i686.manylinux2010\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:a09582f178759ee8128d9270cd1344154fd473bb77d94ce0aeb2a93ebf0feaf0"},

{file = "cffi-1.16.0-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:e760191dd42581e023a68b758769e2da259b5d52e3103c6060ddc02c9edb8d7b"},

{file = "cffi-1.16.0-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:80876338e19c951fdfed6198e70bc88f1c9758b94578d5a7c4c91a87af3cf31c"},

{file = "cffi-1.16.0-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:a6a14b17d7e17fa0d207ac08642c8820f84f25ce17a442fd15e27ea18d67c59b"},

{file = "cffi-1.16.0-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:6602bc8dc6f3a9e02b6c22c4fc1e47aa50f8f8e6d3f78a5e16ac33ef5fefa324"},

{file = "cffi-1.16.0-cp38-cp38-win32.whl", hash = "sha256:131fd094d1065b19540c3d72594260f118b231090295d8c34e19a7bbcf2e860a"},

{file = "cffi-1.16.0-cp38-cp38-win\_amd64.whl", hash = "sha256:31d13b0f99e0836b7ff893d37af07366ebc90b678b6664c955b54561fc36ef36"},

{file = "cffi-1.16.0-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:582215a0e9adbe0e379761260553ba11c58943e4bbe9c36430c4ca6ac74b15ed"},

{file = "cffi-1.16.0-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:b29ebffcf550f9da55bec9e02ad430c992a87e5f512cd63388abb76f1036d8d2"},

{file = "cffi-1.16.0-cp39-cp39-manylinux\_2\_12\_i686.manylinux2010\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:dc9b18bf40cc75f66f40a7379f6a9513244fe33c0e8aa72e2d56b0196a7ef872"},

{file = "cffi-1.16.0-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:9cb4a35b3642fc5c005a6755a5d17c6c8b6bcb6981baf81cea8bfbc8903e8ba8"},

{file = "cffi-1.16.0-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:b86851a328eedc692acf81fb05444bdf1891747c25af7529e39ddafaf68a4f3f"},

{file = "cffi-1.16.0-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:c0f31130ebc2d37cdd8e44605fb5fa7ad59049298b3f745c74fa74c62fbfcfc4"},

{file = "cffi-1.16.0-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:8f8e709127c6c77446a8c0a8c8bf3c8ee706a06cd44b1e827c3e6a2ee6b8c098"},

{file = "cffi-1.16.0-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:748dcd1e3d3d7cd5443ef03ce8685043294ad6bd7c02a38d1bd367cfd968e000"},

{file = "cffi-1.16.0-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:8895613bcc094d4a1b2dbe179d88d7fb4a15cee43c052e8885783fac397d91fe"},

{file = "cffi-1.16.0-cp39-cp39-win32.whl", hash = "sha256:ed86a35631f7bfbb28e108dd96773b9d5a6ce4811cf6ea468bb6a359b256b1e4"},

{file = "cffi-1.16.0-cp39-cp39-win\_amd64.whl", hash = "sha256:3686dffb02459559c74dd3d81748269ffb0eb027c39a6fc99502de37d501faa8"},

{file = "cffi-1.16.0.tar.gz", hash = "sha256:bcb3ef43e58665bbda2fb198698fcae6776483e0c4a631aa5647806c25e02cc0"},

]

[package.dependencies]

pycparser = "\*"

[[package]]

name = "cfgv"

version = "3.4.0"

description = "Validate configuration and produce human readable error messages."

optional = false

python-versions = ">=3.8"

files = [

{file = "cfgv-3.4.0-py2.py3-none-any.whl", hash = "sha256:b7265b1f29fd3316bfcd2b330d63d024f2bfd8bcb8b0272f8e19a504856c48f9"},

{file = "cfgv-3.4.0.tar.gz", hash = "sha256:e52591d4c5f5dead8e0f673fb16db7949d2cfb3f7da4582893288f0ded8fe560"},

]

[[package]]

name = "charset-normalizer"

version = "3.3.2"

description = "The Real First Universal Charset Detector. Open, modern and actively maintained alternative to Chardet."

optional = false

python-versions = ">=3.7.0"

files = [

{file = "charset-normalizer-3.3.2.tar.gz", hash = "sha256:f30c3cb33b24454a82faecaf01b19c18562b1e89558fb6c56de4d9118a032fd5"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-macosx\_10\_9\_universal2.whl", hash = "sha256:25baf083bf6f6b341f4121c2f3c548875ee6f5339300e08be3f2b2ba1721cdd3"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:06435b539f889b1f6f4ac1758871aae42dc3a8c0e24ac9e60c2384973ad73027"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:9063e24fdb1e498ab71cb7419e24622516c4a04476b17a2dab57e8baa30d6e03"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:6897af51655e3691ff853668779c7bad41579facacf5fd7253b0133308cf000d"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:1d3193f4a680c64b4b6a9115943538edb896edc190f0b222e73761716519268e"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:cd70574b12bb8a4d2aaa0094515df2463cb429d8536cfb6c7ce983246983e5a6"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:8465322196c8b4d7ab6d1e049e4c5cb460d0394da4a27d23cc242fbf0034b6b5"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:a9a8e9031d613fd2009c182b69c7b2c1ef8239a0efb1df3f7c8da66d5dd3d537"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:beb58fe5cdb101e3a055192ac291b7a21e3b7ef4f67fa1d74e331a7f2124341c"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:e06ed3eb3218bc64786f7db41917d4e686cc4856944f53d5bdf83a6884432e12"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:2e81c7b9c8979ce92ed306c249d46894776a909505d8f5a4ba55b14206e3222f"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-musllinux\_1\_1\_s390x.whl", hash = "sha256:572c3763a264ba47b3cf708a44ce965d98555f618ca42c926a9c1616d8f34269"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:fd1abc0d89e30cc4e02e4064dc67fcc51bd941eb395c502aac3ec19fab46b519"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-win32.whl", hash = "sha256:3d47fa203a7bd9c5b6cee4736ee84ca03b8ef23193c0d1ca99b5089f72645c73"},

{file = "charset\_normalizer-3.3.2-cp310-cp310-win\_amd64.whl", hash = "sha256:10955842570876604d404661fbccbc9c7e684caf432c09c715ec38fbae45ae09"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-macosx\_10\_9\_universal2.whl", hash = "sha256:802fe99cca7457642125a8a88a084cef28ff0cf9407060f7b93dca5aa25480db"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:573f6eac48f4769d667c4442081b1794f52919e7edada77495aaed9236d13a96"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:549a3a73da901d5bc3ce8d24e0600d1fa85524c10287f6004fbab87672bf3e1e"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:f27273b60488abe721a075bcca6d7f3964f9f6f067c8c4c605743023d7d3944f"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:1ceae2f17a9c33cb48e3263960dc5fc8005351ee19db217e9b1bb15d28c02574"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:65f6f63034100ead094b8744b3b97965785388f308a64cf8d7c34f2f2e5be0c4"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:753f10e867343b4511128c6ed8c82f7bec3bd026875576dfd88483c5c73b2fd8"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:4a78b2b446bd7c934f5dcedc588903fb2f5eec172f3d29e52a9096a43722adfc"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:e537484df0d8f426ce2afb2d0f8e1c3d0b114b83f8850e5f2fbea0e797bd82ae"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:eb6904c354526e758fda7167b33005998fb68c46fbc10e013ca97f21ca5c8887"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:deb6be0ac38ece9ba87dea880e438f25ca3eddfac8b002a2ec3d9183a454e8ae"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-musllinux\_1\_1\_s390x.whl", hash = "sha256:4ab2fe47fae9e0f9dee8c04187ce5d09f48eabe611be8259444906793ab7cbce"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:80402cd6ee291dcb72644d6eac93785fe2c8b9cb30893c1af5b8fdd753b9d40f"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-win32.whl", hash = "sha256:7cd13a2e3ddeed6913a65e66e94b51d80a041145a026c27e6bb76c31a853c6ab"},

{file = "charset\_normalizer-3.3.2-cp311-cp311-win\_amd64.whl", hash = "sha256:663946639d296df6a2bb2aa51b60a2454ca1cb29835324c640dafb5ff2131a77"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-macosx\_10\_9\_universal2.whl", hash = "sha256:0b2b64d2bb6d3fb9112bafa732def486049e63de9618b5843bcdd081d8144cd8"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:ddbb2551d7e0102e7252db79ba445cdab71b26640817ab1e3e3648dad515003b"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:55086ee1064215781fff39a1af09518bc9255b50d6333f2e4c74ca09fac6a8f6"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:8f4a014bc36d3c57402e2977dada34f9c12300af536839dc38c0beab8878f38a"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:a10af20b82360ab00827f916a6058451b723b4e65030c5a18577c8b2de5b3389"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:8d756e44e94489e49571086ef83b2bb8ce311e730092d2c34ca8f7d925cb20aa"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:90d558489962fd4918143277a773316e56c72da56ec7aa3dc3dbbe20fdfed15b"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:6ac7ffc7ad6d040517be39eb591cac5ff87416c2537df6ba3cba3bae290c0fed"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:7ed9e526742851e8d5cc9e6cf41427dfc6068d4f5a3bb03659444b4cabf6bc26"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-musllinux\_1\_1\_i686.whl", hash = "sha256:8bdb58ff7ba23002a4c5808d608e4e6c687175724f54a5dade5fa8c67b604e4d"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:6b3251890fff30ee142c44144871185dbe13b11bab478a88887a639655be1068"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-musllinux\_1\_1\_s390x.whl", hash = "sha256:b4a23f61ce87adf89be746c8a8974fe1c823c891d8f86eb218bb957c924bb143"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:efcb3f6676480691518c177e3b465bcddf57cea040302f9f4e6e191af91174d4"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-win32.whl", hash = "sha256:d965bba47ddeec8cd560687584e88cf699fd28f192ceb452d1d7ee807c5597b7"},

{file = "charset\_normalizer-3.3.2-cp312-cp312-win\_amd64.whl", hash = "sha256:96b02a3dc4381e5494fad39be677abcb5e6634bf7b4fa83a6dd3112607547001"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-macosx\_10\_9\_x86\_64.whl", hash = "sha256:95f2a5796329323b8f0512e09dbb7a1860c46a39da62ecb2324f116fa8fdc85c"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:c002b4ffc0be611f0d9da932eb0f704fe2602a9a949d1f738e4c34c75b0863d5"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:a981a536974bbc7a512cf44ed14938cf01030a99e9b3a06dd59578882f06f985"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:3287761bc4ee9e33561a7e058c72ac0938c4f57fe49a09eae428fd88aafe7bb6"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:42cb296636fcc8b0644486d15c12376cb9fa75443e00fb25de0b8602e64c1714"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:0a55554a2fa0d408816b3b5cedf0045f4b8e1a6065aec45849de2d6f3f8e9786"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-musllinux\_1\_1\_aarch64.whl", hash = "sha256:c083af607d2515612056a31f0a8d9e0fcb5876b7bfc0abad3ecd275bc4ebc2d5"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-musllinux\_1\_1\_i686.whl", hash = "sha256:87d1351268731db79e0f8e745d92493ee2841c974128ef629dc518b937d9194c"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:bd8f7df7d12c2db9fab40bdd87a7c09b1530128315d047a086fa3ae3435cb3a8"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-musllinux\_1\_1\_s390x.whl", hash = "sha256:c180f51afb394e165eafe4ac2936a14bee3eb10debc9d9e4db8958fe36afe711"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:8c622a5fe39a48f78944a87d4fb8a53ee07344641b0562c540d840748571b811"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-win32.whl", hash = "sha256:db364eca23f876da6f9e16c9da0df51aa4f104a972735574842618b8c6d999d4"},

{file = "charset\_normalizer-3.3.2-cp37-cp37m-win\_amd64.whl", hash = "sha256:86216b5cee4b06df986d214f664305142d9c76df9b6512be2738aa72a2048f99"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-macosx\_10\_9\_universal2.whl", hash = "sha256:6463effa3186ea09411d50efc7d85360b38d5f09b870c48e4600f63af490e56a"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:6c4caeef8fa63d06bd437cd4bdcf3ffefe6738fb1b25951440d80dc7df8c03ac"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:37e55c8e51c236f95b033f6fb391d7d7970ba5fe7ff453dad675e88cf303377a"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:fb69256e180cb6c8a894fee62b3afebae785babc1ee98b81cdf68bbca1987f33"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:ae5f4161f18c61806f411a13b0310bea87f987c7d2ecdbdaad0e94eb2e404238"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:b2b0a0c0517616b6869869f8c581d4eb2dd83a4d79e0ebcb7d373ef9956aeb0a"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:45485e01ff4d3630ec0d9617310448a8702f70e9c01906b0d0118bdf9d124cf2"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:eb00ed941194665c332bf8e078baf037d6c35d7c4f3102ea2d4f16ca94a26dc8"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:2127566c664442652f024c837091890cb1942c30937add288223dc895793f898"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-musllinux\_1\_1\_i686.whl", hash = "sha256:a50aebfa173e157099939b17f18600f72f84eed3049e743b68ad15bd69b6bf99"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:4d0d1650369165a14e14e1e47b372cfcb31d6ab44e6e33cb2d4e57265290044d"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-musllinux\_1\_1\_s390x.whl", hash = "sha256:923c0c831b7cfcb071580d3f46c4baf50f174be571576556269530f4bbd79d04"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:06a81e93cd441c56a9b65d8e1d043daeb97a3d0856d177d5c90ba85acb3db087"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-win32.whl", hash = "sha256:6ef1d82a3af9d3eecdba2321dc1b3c238245d890843e040e41e470ffa64c3e25"},

{file = "charset\_normalizer-3.3.2-cp38-cp38-win\_amd64.whl", hash = "sha256:eb8821e09e916165e160797a6c17edda0679379a4be5c716c260e836e122f54b"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-macosx\_10\_9\_universal2.whl", hash = "sha256:c235ebd9baae02f1b77bcea61bce332cb4331dc3617d254df3323aa01ab47bd4"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:5b4c145409bef602a690e7cfad0a15a55c13320ff7a3ad7ca59c13bb8ba4d45d"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:68d1f8a9e9e37c1223b656399be5d6b448dea850bed7d0f87a8311f1ff3dabb0"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:22afcb9f253dac0696b5a4be4a1c0f8762f8239e21b99680099abd9b2b1b2269"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:e27ad930a842b4c5eb8ac0016b0a54f5aebbe679340c26101df33424142c143c"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:1f79682fbe303db92bc2b1136016a38a42e835d932bab5b3b1bfcfbf0640e519"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:b261ccdec7821281dade748d088bb6e9b69e6d15b30652b74cbbac25e280b796"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:122c7fa62b130ed55f8f285bfd56d5f4b4a5b503609d181f9ad85e55c89f4185"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:d0eccceffcb53201b5bfebb52600a5fb483a20b61da9dbc885f8b103cbe7598c"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:9f96df6923e21816da7e0ad3fd47dd8f94b2a5ce594e00677c0013018b813458"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:7f04c839ed0b6b98b1a7501a002144b76c18fb1c1850c8b98d458ac269e26ed2"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-musllinux\_1\_1\_s390x.whl", hash = "sha256:34d1c8da1e78d2e001f363791c98a272bb734000fcef47a491c1e3b0505657a8"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:ff8fa367d09b717b2a17a052544193ad76cd49979c805768879cb63d9ca50561"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-win32.whl", hash = "sha256:aed38f6e4fb3f5d6bf81bfa990a07806be9d83cf7bacef998ab1a9bd660a581f"},

{file = "charset\_normalizer-3.3.2-cp39-cp39-win\_amd64.whl", hash = "sha256:b01b88d45a6fcb69667cd6d2f7a9aeb4bf53760d7fc536bf679ec94fe9f3ff3d"},

{file = "charset\_normalizer-3.3.2-py3-none-any.whl", hash = "sha256:3e4d1f6587322d2788836a99c69062fbb091331ec940e02d12d179c1d53e25fc"},

]

[[package]]

name = "click"

version = "8.1.7"

description = "Composable command line interface toolkit"

optional = false

python-versions = ">=3.7"

files = [

{file = "click-8.1.7-py3-none-any.whl", hash = "sha256:ae74fb96c20a0277a1d615f1e4d73c8414f5a98db8b799a7931d1582f3390c28"},

{file = "click-8.1.7.tar.gz", hash = "sha256:ca9853ad459e787e2192211578cc907e7594e294c7ccc834310722b41b9ca6de"},

]

[package.dependencies]

colorama = {version = "\*", markers = "platform\_system == \"Windows\""}

[[package]]

name = "colorama"

version = "0.4.6"

description = "Cross-platform colored terminal text."

optional = false

python-versions = "!=3.0.\*,!=3.1.\*,!=3.2.\*,!=3.3.\*,!=3.4.\*,!=3.5.\*,!=3.6.\*,>=2.7"

files = [

{file = "colorama-0.4.6-py2.py3-none-any.whl", hash = "sha256:4f1d9991f5acc0ca119f9d443620b77f9d6b33703e51011c16baf57afb285fc6"},

{file = "colorama-0.4.6.tar.gz", hash = "sha256:08695f5cb7ed6e0531a20572697297273c47b8cae5a63ffc6d6ed5c201be6e44"},

]

[[package]]

name = "coloredlogs"

version = "15.0.1"

description = "Colored terminal output for Python's logging module"

optional = false

python-versions = ">=2.7, !=3.0.\*, !=3.1.\*, !=3.2.\*, !=3.3.\*, !=3.4.\*"

files = [

{file = "coloredlogs-15.0.1-py2.py3-none-any.whl", hash = "sha256:612ee75c546f53e92e70049c9dbfcc18c935a2b9a53b66085ce9ef6a6e5c0934"},

{file = "coloredlogs-15.0.1.tar.gz", hash = "sha256:7c991aa71a4577af2f82600d8f8f3a89f936baeaf9b50a9c197da014e5bf16b0"},

]

[package.dependencies]

humanfriendly = ">=9.1"

[package.extras]

cron = ["capturer (>=2.4)"]

[[package]]

name = "cose"

version = "0.9.dev8"

description = "CBOR Object Signing and Encryption (COSE) implementation"

optional = false

python-versions = ">=3.6"

files = [

{file = "cose-0.9.dev8-py3-none-any.whl", hash = "sha256:f1c3be98e50724e846e3a1d23efe19a150665a4f24917ac8bfbc8e5abb31ccb0"},

{file = "cose-0.9.dev8.tar.gz", hash = "sha256:c48d1edcf7fbc564f4f4ac9d0daa52378ea9d26216e5c4bf4b324883ae5ef880"},

]

[package.dependencies]

attrs = "\*"

cbor2 = "\*"

certvalidator = "\*"

cryptography = "\*"

ecdsa = "\*"

[[package]]

name = "cryptography"

version = "42.0.7"

description = "cryptography is a package which provides cryptographic recipes and primitives to Python developers."

optional = false

python-versions = ">=3.7"

files = [

{file = "cryptography-42.0.7-cp37-abi3-macosx\_10\_12\_universal2.whl", hash = "sha256:a987f840718078212fdf4504d0fd4c6effe34a7e4740378e59d47696e8dfb477"},

{file = "cryptography-42.0.7-cp37-abi3-macosx\_10\_12\_x86\_64.whl", hash = "sha256:bd13b5e9b543532453de08bcdc3cc7cebec6f9883e886fd20a92f26940fd3e7a"},

{file = "cryptography-42.0.7-cp37-abi3-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a79165431551042cc9d1d90e6145d5d0d3ab0f2d66326c201d9b0e7f5bf43604"},

{file = "cryptography-42.0.7-cp37-abi3-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:a47787a5e3649008a1102d3df55424e86606c9bae6fb77ac59afe06d234605f8"},

{file = "cryptography-42.0.7-cp37-abi3-manylinux\_2\_28\_aarch64.whl", hash = "sha256:02c0eee2d7133bdbbc5e24441258d5d2244beb31da5ed19fbb80315f4bbbff55"},

{file = "cryptography-42.0.7-cp37-abi3-manylinux\_2\_28\_x86\_64.whl", hash = "sha256:5e44507bf8d14b36b8389b226665d597bc0f18ea035d75b4e53c7b1ea84583cc"},

{file = "cryptography-42.0.7-cp37-abi3-musllinux\_1\_1\_aarch64.whl", hash = "sha256:7f8b25fa616d8b846aef64b15c606bb0828dbc35faf90566eb139aa9cff67af2"},

{file = "cryptography-42.0.7-cp37-abi3-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:93a3209f6bb2b33e725ed08ee0991b92976dfdcf4e8b38646540674fc7508e13"},

{file = "cryptography-42.0.7-cp37-abi3-musllinux\_1\_2\_aarch64.whl", hash = "sha256:e6b8f1881dac458c34778d0a424ae5769de30544fc678eac51c1c8bb2183e9da"},

{file = "cryptography-42.0.7-cp37-abi3-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:3de9a45d3b2b7d8088c3fbf1ed4395dfeff79d07842217b38df14ef09ce1d8d7"},

{file = "cryptography-42.0.7-cp37-abi3-win32.whl", hash = "sha256:789caea816c6704f63f6241a519bfa347f72fbd67ba28d04636b7c6b7da94b0b"},

{file = "cryptography-42.0.7-cp37-abi3-win\_amd64.whl", hash = "sha256:8cb8ce7c3347fcf9446f201dc30e2d5a3c898d009126010cbd1f443f28b52678"},

{file = "cryptography-42.0.7-cp39-abi3-macosx\_10\_12\_universal2.whl", hash = "sha256:a3a5ac8b56fe37f3125e5b72b61dcde43283e5370827f5233893d461b7360cd4"},

{file = "cryptography-42.0.7-cp39-abi3-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:779245e13b9a6638df14641d029add5dc17edbef6ec915688f3acb9e720a5858"},

{file = "cryptography-42.0.7-cp39-abi3-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:0d563795db98b4cd57742a78a288cdbdc9daedac29f2239793071fe114f13785"},

{file = "cryptography-42.0.7-cp39-abi3-manylinux\_2\_28\_aarch64.whl", hash = "sha256:31adb7d06fe4383226c3e963471f6837742889b3c4caa55aac20ad951bc8ffda"},

{file = "cryptography-42.0.7-cp39-abi3-manylinux\_2\_28\_x86\_64.whl", hash = "sha256:efd0bf5205240182e0f13bcaea41be4fdf5c22c5129fc7ced4a0282ac86998c9"},

{file = "cryptography-42.0.7-cp39-abi3-musllinux\_1\_1\_aarch64.whl", hash = "sha256:a9bc127cdc4ecf87a5ea22a2556cab6c7eda2923f84e4f3cc588e8470ce4e42e"},

{file = "cryptography-42.0.7-cp39-abi3-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:3577d029bc3f4827dd5bf8bf7710cac13527b470bbf1820a3f394adb38ed7d5f"},

{file = "cryptography-42.0.7-cp39-abi3-musllinux\_1\_2\_aarch64.whl", hash = "sha256:2e47577f9b18723fa294b0ea9a17d5e53a227867a0a4904a1a076d1646d45ca1"},

{file = "cryptography-42.0.7-cp39-abi3-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:1a58839984d9cb34c855197043eaae2c187d930ca6d644612843b4fe8513c886"},

{file = "cryptography-42.0.7-cp39-abi3-win32.whl", hash = "sha256:e6b79d0adb01aae87e8a44c2b64bc3f3fe59515280e00fb6d57a7267a2583cda"},

{file = "cryptography-42.0.7-cp39-abi3-win\_amd64.whl", hash = "sha256:16268d46086bb8ad5bf0a2b5544d8a9ed87a0e33f5e77dd3c3301e63d941a83b"},

{file = "cryptography-42.0.7-pp310-pypy310\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:2954fccea107026512b15afb4aa664a5640cd0af630e2ee3962f2602693f0c82"},

{file = "cryptography-42.0.7-pp310-pypy310\_pp73-manylinux\_2\_28\_aarch64.whl", hash = "sha256:362e7197754c231797ec45ee081f3088a27a47c6c01eff2ac83f60f85a50fe60"},

{file = "cryptography-42.0.7-pp310-pypy310\_pp73-manylinux\_2\_28\_x86\_64.whl", hash = "sha256:4f698edacf9c9e0371112792558d2f705b5645076cc0aaae02f816a0171770fd"},

{file = "cryptography-42.0.7-pp310-pypy310\_pp73-win\_amd64.whl", hash = "sha256:5482e789294854c28237bba77c4c83be698be740e31a3ae5e879ee5444166582"},

{file = "cryptography-42.0.7-pp39-pypy39\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:e9b2a6309f14c0497f348d08a065d52f3020656f675819fc405fb63bbcd26562"},

{file = "cryptography-42.0.7-pp39-pypy39\_pp73-manylinux\_2\_28\_aarch64.whl", hash = "sha256:d8e3098721b84392ee45af2dd554c947c32cc52f862b6a3ae982dbb90f577f14"},

{file = "cryptography-42.0.7-pp39-pypy39\_pp73-manylinux\_2\_28\_x86\_64.whl", hash = "sha256:c65f96dad14f8528a447414125e1fc8feb2ad5a272b8f68477abbcc1ea7d94b9"},

{file = "cryptography-42.0.7-pp39-pypy39\_pp73-win\_amd64.whl", hash = "sha256:36017400817987670037fbb0324d71489b6ead6231c9604f8fc1f7d008087c68"},

{file = "cryptography-42.0.7.tar.gz", hash = "sha256:ecbfbc00bf55888edda9868a4cf927205de8499e7fabe6c050322298382953f2"},

]

[package.dependencies]

cffi = {version = ">=1.12", markers = "platform\_python\_implementation != \"PyPy\""}

[package.extras]

docs = ["sphinx (>=5.3.0)", "sphinx-rtd-theme (>=1.1.1)"]

docstest = ["pyenchant (>=1.6.11)", "readme-renderer", "sphinxcontrib-spelling (>=4.0.1)"]

nox = ["nox"]

pep8test = ["check-sdist", "click", "mypy", "ruff"]

sdist = ["build"]

ssh = ["bcrypt (>=3.1.5)"]

test = ["certifi", "pretend", "pytest (>=6.2.0)", "pytest-benchmark", "pytest-cov", "pytest-xdist"]

test-randomorder = ["pytest-randomly"]

[[package]]

name = "distlib"

version = "0.3.8"

description = "Distribution utilities"

optional = false

python-versions = "\*"

files = [

{file = "distlib-0.3.8-py2.py3-none-any.whl", hash = "sha256:034db59a0b96f8ca18035f36290806a9a6e6bd9d1ff91e45a7f172eb17e51784"},

{file = "distlib-0.3.8.tar.gz", hash = "sha256:1530ea13e350031b6312d8580ddb6b27a104275a31106523b8f123787f494f64"},

]

[[package]]

name = "docker"

version = "6.1.3"

description = "A Python library for the Docker Engine API."

optional = false

python-versions = ">=3.7"

files = [

{file = "docker-6.1.3-py3-none-any.whl", hash = "sha256:aecd2277b8bf8e506e484f6ab7aec39abe0038e29fa4a6d3ba86c3fe01844ed9"},

{file = "docker-6.1.3.tar.gz", hash = "sha256:aa6d17830045ba5ef0168d5eaa34d37beeb113948c413affe1d5991fc11f9a20"},

]

[package.dependencies]

packaging = ">=14.0"

pywin32 = {version = ">=304", markers = "sys\_platform == \"win32\""}

requests = ">=2.26.0"

urllib3 = ">=1.26.0"

websocket-client = ">=0.32.0"

[package.extras]

ssh = ["paramiko (>=2.4.3)"]

[[package]]

name = "ecdsa"

version = "0.19.0"

description = "ECDSA cryptographic signature library (pure python)"

optional = false

python-versions = "!=3.0.\*,!=3.1.\*,!=3.2.\*,!=3.3.\*,!=3.4.\*,>=2.6"

files = [

{file = "ecdsa-0.19.0-py2.py3-none-any.whl", hash = "sha256:2cea9b88407fdac7bbeca0833b189e4c9c53f2ef1e1eaa29f6224dbc809b707a"},

{file = "ecdsa-0.19.0.tar.gz", hash = "sha256:60eaad1199659900dd0af521ed462b793bbdf867432b3948e87416ae4caf6bf8"},

]

[package.dependencies]

six = ">=1.9.0"

[package.extras]

gmpy = ["gmpy"]

gmpy2 = ["gmpy2"]

[[package]]

name = "ecpy"

version = "1.2.5"

description = "Pure Pyhton Elliptic Curve Library"

optional = false

python-versions = "\*"

files = [

{file = "ECPy-1.2.5-py3-none-any.whl", hash = "sha256:559c92e42406d9d1a6b2b8fc26e6ad7bc985f33903b72f426a56cb1073a25ce3"},

{file = "ECPy-1.2.5.tar.gz", hash = "sha256:9635cffb9b6ecf7fd7f72aea1665829ac74a1d272006d0057d45a621aae20228"},

]

[[package]]

name = "exceptiongroup"

version = "1.2.1"

description = "Backport of PEP 654 (exception groups)"

optional = false

python-versions = ">=3.7"

files = [

{file = "exceptiongroup-1.2.1-py3-none-any.whl", hash = "sha256:5258b9ed329c5bbdd31a309f53cbfb0b155341807f6ff7606a1e801a891b29ad"},

{file = "exceptiongroup-1.2.1.tar.gz", hash = "sha256:a4785e48b045528f5bfe627b6ad554ff32def154f42372786903b7abcfe1aa16"},

]

[package.extras]

test = ["pytest (>=6)"]

[[package]]

name = "fastapi"

version = "0.109.2"

description = "FastAPI framework, high performance, easy to learn, fast to code, ready for production"

optional = false

python-versions = ">=3.8"

files = [

{file = "fastapi-0.109.2-py3-none-any.whl", hash = "sha256:2c9bab24667293b501cad8dd388c05240c850b58ec5876ee3283c47d6e1e3a4d"},

{file = "fastapi-0.109.2.tar.gz", hash = "sha256:f3817eac96fe4f65a2ebb4baa000f394e55f5fccdaf7f75250804bc58f354f73"},

]

[package.dependencies]

pydantic = ">=1.7.4,<1.8 || >1.8,<1.8.1 || >1.8.1,<2.0.0 || >2.0.0,<2.0.1 || >2.0.1,<2.1.0 || >2.1.0,<3.0.0"

starlette = ">=0.36.3,<0.37.0"

typing-extensions = ">=4.8.0"

[package.extras]

all = ["email-validator (>=2.0.0)", "httpx (>=0.23.0)", "itsdangerous (>=1.1.0)", "jinja2 (>=2.11.2)", "orjson (>=3.2.1)", "pydantic-extra-types (>=2.0.0)", "pydantic-settings (>=2.0.0)", "python-multipart (>=0.0.7)", "pyyaml (>=5.3.1)", "ujson (>=4.0.1,!=4.0.2,!=4.1.0,!=4.2.0,!=4.3.0,!=5.0.0,!=5.1.0)", "uvicorn[standard] (>=0.12.0)"]

[[package]]

name = "fastapi-cache2"

version = "0.2.1"

description = "Cache for FastAPI"

optional = false

python-versions = ">=3.7,<4.0"

files = [

{file = "fastapi\_cache2-0.2.1-py3-none-any.whl", hash = "sha256:532c24253ab8f162ba2f0a7dcccb98591424f0424c28da70346eb3c99daf2b06"},

{file = "fastapi\_cache2-0.2.1.tar.gz", hash = "sha256:0c68803c5a7fdebfec3e4430fd08482c091344dc4dd55a00209a882220712f89"},

]

[package.dependencies]

aiohttp = {version = ">=3.8.3", markers = "python\_version >= \"3.11\""}

fastapi = "\*"

pendulum = "\*"

uvicorn = "\*"

[package.extras]

all = ["aiobotocore (>=1.4.1,<2.0.0)", "aiomcache", "redis (>=4.2.0rc1,<5.0.0)"]

dynamodb = ["aiobotocore (>=1.4.1,<2.0.0)"]

memcache = ["aiomcache"]

redis = ["redis (>=4.2.0rc1,<5.0.0)"]

[[package]]

name = "filelock"

version = "3.14.0"

description = "A platform independent file lock."

optional = false

python-versions = ">=3.8"

files = [

{file = "filelock-3.14.0-py3-none-any.whl", hash = "sha256:43339835842f110ca7ae60f1e1c160714c5a6afd15a2873419ab185334975c0f"},

{file = "filelock-3.14.0.tar.gz", hash = "sha256:6ea72da3be9b8c82afd3edcf99f2fffbb5076335a5ae4d03248bb5b6c3eae78a"},

]

[package.extras]

docs = ["furo (>=2023.9.10)", "sphinx (>=7.2.6)", "sphinx-autodoc-typehints (>=1.25.2)"]

testing = ["covdefaults (>=2.3)", "coverage (>=7.3.2)", "diff-cover (>=8.0.1)", "pytest (>=7.4.3)", "pytest-cov (>=4.1)", "pytest-mock (>=3.12)", "pytest-timeout (>=2.2)"]

typing = ["typing-extensions (>=4.8)"]

[[package]]

name = "fire"

version = "0.5.0"

description = "A library for automatically generating command line interfaces."

optional = false

python-versions = "\*"

files = [

{file = "fire-0.5.0.tar.gz", hash = "sha256:a6b0d49e98c8963910021f92bba66f65ab440da2982b78eb1bbf95a0a34aacc6"},

]

[package.dependencies]

six = "\*"

termcolor = "\*"

[[package]]

name = "frozendict"

version = "2.4.4"

description = "A simple immutable dictionary"

optional = false

python-versions = ">=3.6"

files = [

{file = "frozendict-2.4.4-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:4a59578d47b3949437519b5c39a016a6116b9e787bb19289e333faae81462e59"},

{file = "frozendict-2.4.4-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:12a342e439aef28ccec533f0253ea53d75fe9102bd6ea928ff530e76eac38906"},

{file = "frozendict-2.4.4-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:7f79c26dff10ce11dad3b3627c89bb2e87b9dd5958c2b24325f16a23019b8b94"},

{file = "frozendict-2.4.4-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:2bd009cf4fc47972838a91e9b83654dc9a095dc4f2bb3a37c3f3124c8a364543"},

{file = "frozendict-2.4.4-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:87ebcde21565a14fe039672c25550060d6f6d88cf1f339beac094c3b10004eb0"},

{file = "frozendict-2.4.4-cp310-cp310-win\_amd64.whl", hash = "sha256:fefeb700bc7eb8b4c2dc48704e4221860d254c8989fb53488540bc44e44a1ac2"},

{file = "frozendict-2.4.4-cp310-cp310-win\_arm64.whl", hash = "sha256:4297d694eb600efa429769125a6f910ec02b85606f22f178bafbee309e7d3ec7"},

{file = "frozendict-2.4.4-cp36-cp36m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:812ab17522ba13637826e65454115a914c2da538356e85f43ecea069813e4b33"},

{file = "frozendict-2.4.4-cp36-cp36m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:7fee9420475bb6ff357000092aa9990c2f6182b2bab15764330f4ad7de2eae49"},

{file = "frozendict-2.4.4-cp36-cp36m-musllinux\_1\_1\_aarch64.whl", hash = "sha256:3148062675536724502c6344d7c485dd4667fdf7980ca9bd05e338ccc0c4471e"},

{file = "frozendict-2.4.4-cp36-cp36m-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:78c94991944dd33c5376f720228e5b252ee67faf3bac50ef381adc9e51e90d9d"},

{file = "frozendict-2.4.4-cp36-cp36m-win\_amd64.whl", hash = "sha256:1697793b5f62b416c0fc1d94638ec91ed3aa4ab277f6affa3a95216ecb3af170"},

{file = "frozendict-2.4.4-cp37-cp37m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:199a4d32194f3afed6258de7e317054155bc9519252b568d9cfffde7e4d834e5"},

{file = "frozendict-2.4.4-cp37-cp37m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:85375ec6e979e6373bffb4f54576a68bf7497c350861d20686ccae38aab69c0a"},

{file = "frozendict-2.4.4-cp37-cp37m-musllinux\_1\_1\_aarch64.whl", hash = "sha256:2d8536e068d6bf281f23fa835ac07747fb0f8851879dd189e9709f9567408b4d"},

{file = "frozendict-2.4.4-cp37-cp37m-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:259528ba6b56fa051bc996f1c4d8b57e30d6dd3bc2f27441891b04babc4b5e73"},

{file = "frozendict-2.4.4-cp37-cp37m-win\_amd64.whl", hash = "sha256:07c3a5dee8bbb84cba770e273cdbf2c87c8e035903af8f781292d72583416801"},

{file = "frozendict-2.4.4-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:6874fec816b37b6eb5795b00e0574cba261bf59723e2de607a195d5edaff0786"},

{file = "frozendict-2.4.4-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:c8f92425686323a950337da4b75b4c17a3327b831df8c881df24038d560640d4"},

{file = "frozendict-2.4.4-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:5d58d9a8d9e49662c6dafbea5e641f97decdb3d6ccd76e55e79818415362ba25"},

{file = "frozendict-2.4.4-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:93a7b19afb429cbf99d56faf436b45ef2fa8fe9aca89c49eb1610c3bd85f1760"},

{file = "frozendict-2.4.4-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:2b70b431e3a72d410a2cdf1497b3aba2f553635e0c0f657ce311d841bf8273b6"},

{file = "frozendict-2.4.4-cp38-cp38-win\_amd64.whl", hash = "sha256:e1b941132d79ce72d562a13341d38fc217bc1ee24d8c35a20d754e79ff99e038"},

{file = "frozendict-2.4.4-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:dc2228874eacae390e63fd4f2bb513b3144066a977dc192163c9f6c7f6de6474"},

{file = "frozendict-2.4.4-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:63aa49f1919af7d45fb8fd5dec4c0859bc09f46880bd6297c79bb2db2969b63d"},

{file = "frozendict-2.4.4-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:c6bf9260018d653f3cab9bd147bd8592bf98a5c6e338be0491ced3c196c034a3"},

{file = "frozendict-2.4.4-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:6eb716e6a6d693c03b1d53280a1947716129f5ef9bcdd061db5c17dea44b80fe"},

{file = "frozendict-2.4.4-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:d13b4310db337f4d2103867c5a05090b22bc4d50ca842093779ef541ea9c9eea"},

{file = "frozendict-2.4.4-cp39-cp39-win\_amd64.whl", hash = "sha256:b3b967d5065872e27b06f785a80c0ed0a45d1f7c9b85223da05358e734d858ca"},

{file = "frozendict-2.4.4-cp39-cp39-win\_arm64.whl", hash = "sha256:4ae8d05c8d0b6134bfb6bfb369d5fa0c4df21eabb5ca7f645af95fdc6689678e"},

{file = "frozendict-2.4.4.tar.gz", hash = "sha256:3f7c031b26e4ee6a3f786ceb5e3abf1181c4ade92dce1f847da26ea2c96008c7"},

]

[[package]]

name = "frozenlist"

version = "1.4.1"

description = "A list-like structure which implements collections.abc.MutableSequence"

optional = false

python-versions = ">=3.8"

files = [

{file = "frozenlist-1.4.1-cp310-cp310-macosx\_10\_9\_universal2.whl", hash = "sha256:f9aa1878d1083b276b0196f2dfbe00c9b7e752475ed3b682025ff20c1c1f51ac"},

{file = "frozenlist-1.4.1-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:29acab3f66f0f24674b7dc4736477bcd4bc3ad4b896f5f45379a67bce8b96868"},

{file = "frozenlist-1.4.1-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:74fb4bee6880b529a0c6560885fce4dc95936920f9f20f53d99a213f7bf66776"},

{file = "frozenlist-1.4.1-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:590344787a90ae57d62511dd7c736ed56b428f04cd8c161fcc5e7232c130c69a"},

{file = "frozenlist-1.4.1-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:068b63f23b17df8569b7fdca5517edef76171cf3897eb68beb01341131fbd2ad"},

{file = "frozenlist-1.4.1-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:5c849d495bf5154cd8da18a9eb15db127d4dba2968d88831aff6f0331ea9bd4c"},

{file = "frozenlist-1.4.1-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:9750cc7fe1ae3b1611bb8cfc3f9ec11d532244235d75901fb6b8e42ce9229dfe"},

{file = "frozenlist-1.4.1-cp310-cp310-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:a9b2de4cf0cdd5bd2dee4c4f63a653c61d2408055ab77b151c1957f221cabf2a"},

{file = "frozenlist-1.4.1-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:0633c8d5337cb5c77acbccc6357ac49a1770b8c487e5b3505c57b949b4b82e98"},

{file = "frozenlist-1.4.1-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:27657df69e8801be6c3638054e202a135c7f299267f1a55ed3a598934f6c0d75"},

{file = "frozenlist-1.4.1-cp310-cp310-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:f9a3ea26252bd92f570600098783d1371354d89d5f6b7dfd87359d669f2109b5"},

{file = "frozenlist-1.4.1-cp310-cp310-musllinux\_1\_1\_s390x.whl", hash = "sha256:4f57dab5fe3407b6c0c1cc907ac98e8a189f9e418f3b6e54d65a718aaafe3950"},

{file = "frozenlist-1.4.1-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:e02a0e11cf6597299b9f3bbd3f93d79217cb90cfd1411aec33848b13f5c656cc"},

{file = "frozenlist-1.4.1-cp310-cp310-win32.whl", hash = "sha256:a828c57f00f729620a442881cc60e57cfcec6842ba38e1b19fd3e47ac0ff8dc1"},

{file = "frozenlist-1.4.1-cp310-cp310-win\_amd64.whl", hash = "sha256:f56e2333dda1fe0f909e7cc59f021eba0d2307bc6f012a1ccf2beca6ba362439"},

{file = "frozenlist-1.4.1-cp311-cp311-macosx\_10\_9\_universal2.whl", hash = "sha256:a0cb6f11204443f27a1628b0e460f37fb30f624be6051d490fa7d7e26d4af3d0"},

{file = "frozenlist-1.4.1-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:b46c8ae3a8f1f41a0d2ef350c0b6e65822d80772fe46b653ab6b6274f61d4a49"},

{file = "frozenlist-1.4.1-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:fde5bd59ab5357e3853313127f4d3565fc7dad314a74d7b5d43c22c6a5ed2ced"},

{file = "frozenlist-1.4.1-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:722e1124aec435320ae01ee3ac7bec11a5d47f25d0ed6328f2273d287bc3abb0"},

{file = "frozenlist-1.4.1-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:2471c201b70d58a0f0c1f91261542a03d9a5e088ed3dc6c160d614c01649c106"},

{file = "frozenlist-1.4.1-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:c757a9dd70d72b076d6f68efdbb9bc943665ae954dad2801b874c8c69e185068"},

{file = "frozenlist-1.4.1-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:f146e0911cb2f1da549fc58fc7bcd2b836a44b79ef871980d605ec392ff6b0d2"},

{file = "frozenlist-1.4.1-cp311-cp311-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:4f9c515e7914626b2a2e1e311794b4c35720a0be87af52b79ff8e1429fc25f19"},

{file = "frozenlist-1.4.1-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:c302220494f5c1ebeb0912ea782bcd5e2f8308037b3c7553fad0e48ebad6ad82"},

{file = "frozenlist-1.4.1-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:442acde1e068288a4ba7acfe05f5f343e19fac87bfc96d89eb886b0363e977ec"},

{file = "frozenlist-1.4.1-cp311-cp311-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:1b280e6507ea8a4fa0c0a7150b4e526a8d113989e28eaaef946cc77ffd7efc0a"},

{file = "frozenlist-1.4.1-cp311-cp311-musllinux\_1\_1\_s390x.whl", hash = "sha256:fe1a06da377e3a1062ae5fe0926e12b84eceb8a50b350ddca72dc85015873f74"},

{file = "frozenlist-1.4.1-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:db9e724bebd621d9beca794f2a4ff1d26eed5965b004a97f1f1685a173b869c2"},

{file = "frozenlist-1.4.1-cp311-cp311-win32.whl", hash = "sha256:e774d53b1a477a67838a904131c4b0eef6b3d8a651f8b138b04f748fccfefe17"},

{file = "frozenlist-1.4.1-cp311-cp311-win\_amd64.whl", hash = "sha256:fb3c2db03683b5767dedb5769b8a40ebb47d6f7f45b1b3e3b4b51ec8ad9d9825"},

{file = "frozenlist-1.4.1-cp312-cp312-macosx\_10\_9\_universal2.whl", hash = "sha256:1979bc0aeb89b33b588c51c54ab0161791149f2461ea7c7c946d95d5f93b56ae"},

{file = "frozenlist-1.4.1-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:cc7b01b3754ea68a62bd77ce6020afaffb44a590c2289089289363472d13aedb"},

{file = "frozenlist-1.4.1-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:c9c92be9fd329ac801cc420e08452b70e7aeab94ea4233a4804f0915c14eba9b"},

{file = "frozenlist-1.4.1-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:5c3894db91f5a489fc8fa6a9991820f368f0b3cbdb9cd8849547ccfab3392d86"},

{file = "frozenlist-1.4.1-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:ba60bb19387e13597fb059f32cd4d59445d7b18b69a745b8f8e5db0346f33480"},

{file = "frozenlist-1.4.1-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:8aefbba5f69d42246543407ed2461db31006b0f76c4e32dfd6f42215a2c41d09"},

{file = "frozenlist-1.4.1-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:780d3a35680ced9ce682fbcf4cb9c2bad3136eeff760ab33707b71db84664e3a"},

{file = "frozenlist-1.4.1-cp312-cp312-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:9acbb16f06fe7f52f441bb6f413ebae6c37baa6ef9edd49cdd567216da8600cd"},

{file = "frozenlist-1.4.1-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:23b701e65c7b36e4bf15546a89279bd4d8675faabc287d06bbcfac7d3c33e1e6"},

{file = "frozenlist-1.4.1-cp312-cp312-musllinux\_1\_1\_i686.whl", hash = "sha256:3e0153a805a98f5ada7e09826255ba99fb4f7524bb81bf6b47fb702666484ae1"},

{file = "frozenlist-1.4.1-cp312-cp312-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:dd9b1baec094d91bf36ec729445f7769d0d0cf6b64d04d86e45baf89e2b9059b"},

{file = "frozenlist-1.4.1-cp312-cp312-musllinux\_1\_1\_s390x.whl", hash = "sha256:1a4471094e146b6790f61b98616ab8e44f72661879cc63fa1049d13ef711e71e"},

{file = "frozenlist-1.4.1-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:5667ed53d68d91920defdf4035d1cdaa3c3121dc0b113255124bcfada1cfa1b8"},

{file = "frozenlist-1.4.1-cp312-cp312-win32.whl", hash = "sha256:beee944ae828747fd7cb216a70f120767fc9f4f00bacae8543c14a6831673f89"},

{file = "frozenlist-1.4.1-cp312-cp312-win\_amd64.whl", hash = "sha256:64536573d0a2cb6e625cf309984e2d873979709f2cf22839bf2d61790b448ad5"},

{file = "frozenlist-1.4.1-cp38-cp38-macosx\_10\_9\_universal2.whl", hash = "sha256:20b51fa3f588ff2fe658663db52a41a4f7aa6c04f6201449c6c7c476bd255c0d"},

{file = "frozenlist-1.4.1-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:410478a0c562d1a5bcc2f7ea448359fcb050ed48b3c6f6f4f18c313a9bdb1826"},

{file = "frozenlist-1.4.1-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:c6321c9efe29975232da3bd0af0ad216800a47e93d763ce64f291917a381b8eb"},

{file = "frozenlist-1.4.1-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:48f6a4533887e189dae092f1cf981f2e3885175f7a0f33c91fb5b7b682b6bab6"},

{file = "frozenlist-1.4.1-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:6eb73fa5426ea69ee0e012fb59cdc76a15b1283d6e32e4f8dc4482ec67d1194d"},

{file = "frozenlist-1.4.1-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:fbeb989b5cc29e8daf7f976b421c220f1b8c731cbf22b9130d8815418ea45887"},

{file = "frozenlist-1.4.1-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:32453c1de775c889eb4e22f1197fe3bdfe457d16476ea407472b9442e6295f7a"},

{file = "frozenlist-1.4.1-cp38-cp38-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:693945278a31f2086d9bf3df0fe8254bbeaef1fe71e1351c3bd730aa7d31c41b"},

{file = "frozenlist-1.4.1-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:1d0ce09d36d53bbbe566fe296965b23b961764c0bcf3ce2fa45f463745c04701"},

{file = "frozenlist-1.4.1-cp38-cp38-musllinux\_1\_1\_i686.whl", hash = "sha256:3a670dc61eb0d0eb7080890c13de3066790f9049b47b0de04007090807c776b0"},

{file = "frozenlist-1.4.1-cp38-cp38-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:dca69045298ce5c11fd539682cff879cc1e664c245d1c64da929813e54241d11"},

{file = "frozenlist-1.4.1-cp38-cp38-musllinux\_1\_1\_s390x.whl", hash = "sha256:a06339f38e9ed3a64e4c4e43aec7f59084033647f908e4259d279a52d3757d09"},

{file = "frozenlist-1.4.1-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:b7f2f9f912dca3934c1baec2e4585a674ef16fe00218d833856408c48d5beee7"},

{file = "frozenlist-1.4.1-cp38-cp38-win32.whl", hash = "sha256:e7004be74cbb7d9f34553a5ce5fb08be14fb33bc86f332fb71cbe5216362a497"},

{file = "frozenlist-1.4.1-cp38-cp38-win\_amd64.whl", hash = "sha256:5a7d70357e7cee13f470c7883a063aae5fe209a493c57d86eb7f5a6f910fae09"},

{file = "frozenlist-1.4.1-cp39-cp39-macosx\_10\_9\_universal2.whl", hash = "sha256:bfa4a17e17ce9abf47a74ae02f32d014c5e9404b6d9ac7f729e01562bbee601e"},

{file = "frozenlist-1.4.1-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:b7e3ed87d4138356775346e6845cccbe66cd9e207f3cd11d2f0b9fd13681359d"},

{file = "frozenlist-1.4.1-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:c99169d4ff810155ca50b4da3b075cbde79752443117d89429595c2e8e37fed8"},

{file = "frozenlist-1.4.1-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:edb678da49d9f72c9f6c609fbe41a5dfb9a9282f9e6a2253d5a91e0fc382d7c0"},

{file = "frozenlist-1.4.1-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:6db4667b187a6742b33afbbaf05a7bc551ffcf1ced0000a571aedbb4aa42fc7b"},

{file = "frozenlist-1.4.1-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:55fdc093b5a3cb41d420884cdaf37a1e74c3c37a31f46e66286d9145d2063bd0"},

{file = "frozenlist-1.4.1-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:82e8211d69a4f4bc360ea22cd6555f8e61a1bd211d1d5d39d3d228b48c83a897"},

{file = "frozenlist-1.4.1-cp39-cp39-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:89aa2c2eeb20957be2d950b85974b30a01a762f3308cd02bb15e1ad632e22dc7"},

{file = "frozenlist-1.4.1-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:9d3e0c25a2350080e9319724dede4f31f43a6c9779be48021a7f4ebde8b2d742"},

{file = "frozenlist-1.4.1-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:7268252af60904bf52c26173cbadc3a071cece75f873705419c8681f24d3edea"},

{file = "frozenlist-1.4.1-cp39-cp39-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:0c250a29735d4f15321007fb02865f0e6b6a41a6b88f1f523ca1596ab5f50bd5"},

{file = "frozenlist-1.4.1-cp39-cp39-musllinux\_1\_1\_s390x.whl", hash = "sha256:96ec70beabbd3b10e8bfe52616a13561e58fe84c0101dd031dc78f250d5128b9"},

{file = "frozenlist-1.4.1-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:23b2d7679b73fe0e5a4560b672a39f98dfc6f60df63823b0a9970525325b95f6"},

{file = "frozenlist-1.4.1-cp39-cp39-win32.whl", hash = "sha256:a7496bfe1da7fb1a4e1cc23bb67c58fab69311cc7d32b5a99c2007b4b2a0e932"},

{file = "frozenlist-1.4.1-cp39-cp39-win\_amd64.whl", hash = "sha256:e6a20a581f9ce92d389a8c7d7c3dd47c81fd5d6e655c8dddf341e14aa48659d0"},

{file = "frozenlist-1.4.1-py3-none-any.whl", hash = "sha256:04ced3e6a46b4cfffe20f9ae482818e34eba9b5fb0ce4056e4cc9b6e212d09b7"},

{file = "frozenlist-1.4.1.tar.gz", hash = "sha256:c037a86e8513059a2613aaba4d817bb90b9d9b6b69aace3ce9c877e8c8ed402b"},

]

[[package]]

name = "frozenlist2"

version = "1.0.0"

description = "An immutable list"

optional = false

python-versions = "\*"

files = [

{file = "frozenlist2-1.0.0-py3-none-any.whl", hash = "sha256:caffe66813e42de320b10d08b8c0604c7eb3f142a8482ad3130243e084f37a2f"},

{file = "frozenlist2-1.0.0.tar.gz", hash = "sha256:33f6c6bb2c7d38524ec3c2d6f2d8a3ee2625a9e13096d8bc64db012b516a95e0"},

]

[[package]]

name = "gelidum"

version = "0.7.0"

description = "Freeze your python objects"

optional = false

python-versions = "\*"

files = [

{file = "gelidum-0.7.0.tar.gz", hash = "sha256:726029916cca89cc0e5b4eec388ae3df15468a8fa8e96f996963520bec593709"},

]

[[package]]

name = "h11"

version = "0.14.0"

description = "A pure-Python, bring-your-own-I/O implementation of HTTP/1.1"

optional = false

python-versions = ">=3.7"

files = [

{file = "h11-0.14.0-py3-none-any.whl", hash = "sha256:e3fe4ac4b851c468cc8363d500db52c2ead036020723024a109d37346efaa761"},

{file = "h11-0.14.0.tar.gz", hash = "sha256:8f19fbbe99e72420ff35c00b27a34cb9937e902a8b810e2c88300c6f0a3b699d"},

]

[[package]]

name = "humanfriendly"

version = "10.0"

description = "Human friendly output for text interfaces using Python"

optional = false

python-versions = ">=2.7, !=3.0.\*, !=3.1.\*, !=3.2.\*, !=3.3.\*, !=3.4.\*"

files = [

{file = "humanfriendly-10.0-py2.py3-none-any.whl", hash = "sha256:1697e1a8a8f550fd43c2865cd84542fc175a61dcb779b6fee18cf6b6ccba1477"},

{file = "humanfriendly-10.0.tar.gz", hash = "sha256:6b0b831ce8f15f7300721aa49829fc4e83921a9a301cc7f606be6686a2288ddc"},

]

[package.dependencies]

pyreadline3 = {version = "\*", markers = "sys\_platform == \"win32\" and python\_version >= \"3.8\""}

[[package]]

name = "hypothesis"

version = "6.102.4"

description = "A library for property-based testing"

optional = false

python-versions = ">=3.8"

files = [

{file = "hypothesis-6.102.4-py3-none-any.whl", hash = "sha256:013df31b04a4daede13756f497e60e451963d86f426395a79f99c5d692919bbd"},

{file = "hypothesis-6.102.4.tar.gz", hash = "sha256:59b4d144346d5cffb482cc1bafbd21b13ff31608e8c4b3e4630339aee3e87763"},

]

[package.dependencies]

attrs = ">=22.2.0"

exceptiongroup = {version = ">=1.0.0", markers = "python\_version < \"3.11\""}

sortedcontainers = ">=2.1.0,<3.0.0"

[package.extras]

all = ["backports.zoneinfo (>=0.2.1)", "black (>=19.10b0)", "click (>=7.0)", "crosshair-tool (>=0.0.54)", "django (>=3.2)", "dpcontracts (>=0.4)", "hypothesis-crosshair (>=0.0.2)", "lark (>=0.10.1)", "libcst (>=0.3.16)", "numpy (>=1.17.3)", "pandas (>=1.1)", "pytest (>=4.6)", "python-dateutil (>=1.4)", "pytz (>=2014.1)", "redis (>=3.0.0)", "rich (>=9.0.0)", "tzdata (>=2024.1)"]

cli = ["black (>=19.10b0)", "click (>=7.0)", "rich (>=9.0.0)"]

codemods = ["libcst (>=0.3.16)"]

crosshair = ["crosshair-tool (>=0.0.54)", "hypothesis-crosshair (>=0.0.2)"]

dateutil = ["python-dateutil (>=1.4)"]

django = ["django (>=3.2)"]

dpcontracts = ["dpcontracts (>=0.4)"]

ghostwriter = ["black (>=19.10b0)"]

lark = ["lark (>=0.10.1)"]

numpy = ["numpy (>=1.17.3)"]

pandas = ["pandas (>=1.1)"]

pytest = ["pytest (>=4.6)"]

pytz = ["pytz (>=2014.1)"]

redis = ["redis (>=3.0.0)"]

zoneinfo = ["backports.zoneinfo (>=0.2.1)", "tzdata (>=2024.1)"]

[[package]]

name = "identify"

version = "2.5.36"

description = "File identification library for Python"

optional = false

python-versions = ">=3.8"

files = [

{file = "identify-2.5.36-py2.py3-none-any.whl", hash = "sha256:37d93f380f4de590500d9dba7db359d0d3da95ffe7f9de1753faa159e71e7dfa"},

{file = "identify-2.5.36.tar.gz", hash = "sha256:e5e00f54165f9047fbebeb4a560f9acfb8af4c88232be60a488e9b68d122745d"},

]

[package.extras]

license = ["ukkonen"]

[[package]]

name = "idna"

version = "3.7"

description = "Internationalized Domain Names in Applications (IDNA)"

optional = false

python-versions = ">=3.5"

files = [

{file = "idna-3.7-py3-none-any.whl", hash = "sha256:82fee1fc78add43492d3a1898bfa6d8a904cc97d8427f683ed8e798d07761aa0"},

{file = "idna-3.7.tar.gz", hash = "sha256:028ff3aadf0609c1fd278d8ea3089299412a7a8b9bd005dd08b9f8285bcb5cfc"},

]

[[package]]

name = "iniconfig"

version = "2.0.0"

description = "brain-dead simple config-ini parsing"

optional = false

python-versions = ">=3.7"

files = [

{file = "iniconfig-2.0.0-py3-none-any.whl", hash = "sha256:b6a85871a79d2e3b22d2d1b94ac2824226a63c6b741c88f7ae975f18b6778374"},

{file = "iniconfig-2.0.0.tar.gz", hash = "sha256:2d91e135bf72d31a410b17c16da610a82cb55f6b0477d1a902134b24a455b8b3"},

]

[[package]]

name = "mnemonic"

version = "0.20"

description = "Implementation of Bitcoin BIP-0039"

optional = false

python-versions = ">=3.5"

files = [

{file = "mnemonic-0.20-py3-none-any.whl", hash = "sha256:acd2168872d0379e7a10873bb3e12bf6c91b35de758135c4fbd1015ef18fafc5"},

{file = "mnemonic-0.20.tar.gz", hash = "sha256:7c6fb5639d779388027a77944680aee4870f0fcd09b1e42a5525ee2ce4c625f6"},

]

[[package]]

name = "multidict"

version = "6.0.5"

description = "multidict implementation"

optional = false

python-versions = ">=3.7"

files = [

{file = "multidict-6.0.5-cp310-cp310-macosx\_10\_9\_universal2.whl", hash = "sha256:228b644ae063c10e7f324ab1ab6b548bdf6f8b47f3ec234fef1093bc2735e5f9"},

{file = "multidict-6.0.5-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:896ebdcf62683551312c30e20614305f53125750803b614e9e6ce74a96232604"},

{file = "multidict-6.0.5-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:411bf8515f3be9813d06004cac41ccf7d1cd46dfe233705933dd163b60e37600"},

{file = "multidict-6.0.5-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:1d147090048129ce3c453f0292e7697d333db95e52616b3793922945804a433c"},

{file = "multidict-6.0.5-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:215ed703caf15f578dca76ee6f6b21b7603791ae090fbf1ef9d865571039ade5"},

{file = "multidict-6.0.5-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:7c6390cf87ff6234643428991b7359b5f59cc15155695deb4eda5c777d2b880f"},

{file = "multidict-6.0.5-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:21fd81c4ebdb4f214161be351eb5bcf385426bf023041da2fd9e60681f3cebae"},

{file = "multidict-6.0.5-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:3cc2ad10255f903656017363cd59436f2111443a76f996584d1077e43ee51182"},

{file = "multidict-6.0.5-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:6939c95381e003f54cd4c5516740faba40cf5ad3eeff460c3ad1d3e0ea2549bf"},

{file = "multidict-6.0.5-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:220dd781e3f7af2c2c1053da9fa96d9cf3072ca58f057f4c5adaaa1cab8fc442"},

{file = "multidict-6.0.5-cp310-cp310-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:766c8f7511df26d9f11cd3a8be623e59cca73d44643abab3f8c8c07620524e4a"},

{file = "multidict-6.0.5-cp310-cp310-musllinux\_1\_1\_s390x.whl", hash = "sha256:fe5d7785250541f7f5019ab9cba2c71169dc7d74d0f45253f8313f436458a4ef"},

{file = "multidict-6.0.5-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:c1c1496e73051918fcd4f58ff2e0f2f3066d1c76a0c6aeffd9b45d53243702cc"},

{file = "multidict-6.0.5-cp310-cp310-win32.whl", hash = "sha256:7afcdd1fc07befad18ec4523a782cde4e93e0a2bf71239894b8d61ee578c1319"},

{file = "multidict-6.0.5-cp310-cp310-win\_amd64.whl", hash = "sha256:99f60d34c048c5c2fabc766108c103612344c46e35d4ed9ae0673d33c8fb26e8"},

{file = "multidict-6.0.5-cp311-cp311-macosx\_10\_9\_universal2.whl", hash = "sha256:f285e862d2f153a70586579c15c44656f888806ed0e5b56b64489afe4a2dbfba"},

{file = "multidict-6.0.5-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:53689bb4e102200a4fafa9de9c7c3c212ab40a7ab2c8e474491914d2305f187e"},

{file = "multidict-6.0.5-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:612d1156111ae11d14afaf3a0669ebf6c170dbb735e510a7438ffe2369a847fd"},

{file = "multidict-6.0.5-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:7be7047bd08accdb7487737631d25735c9a04327911de89ff1b26b81745bd4e3"},

{file = "multidict-6.0.5-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:de170c7b4fe6859beb8926e84f7d7d6c693dfe8e27372ce3b76f01c46e489fcf"},

{file = "multidict-6.0.5-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:04bde7a7b3de05732a4eb39c94574db1ec99abb56162d6c520ad26f83267de29"},

{file = "multidict-6.0.5-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:85f67aed7bb647f93e7520633d8f51d3cbc6ab96957c71272b286b2f30dc70ed"},

{file = "multidict-6.0.5-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:425bf820055005bfc8aa9a0b99ccb52cc2f4070153e34b701acc98d201693733"},

{file = "multidict-6.0.5-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:d3eb1ceec286eba8220c26f3b0096cf189aea7057b6e7b7a2e60ed36b373b77f"},

{file = "multidict-6.0.5-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:7901c05ead4b3fb75113fb1dd33eb1253c6d3ee37ce93305acd9d38e0b5f21a4"},

{file = "multidict-6.0.5-cp311-cp311-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:e0e79d91e71b9867c73323a3444724d496c037e578a0e1755ae159ba14f4f3d1"},

{file = "multidict-6.0.5-cp311-cp311-musllinux\_1\_1\_s390x.whl", hash = "sha256:29bfeb0dff5cb5fdab2023a7a9947b3b4af63e9c47cae2a10ad58394b517fddc"},

{file = "multidict-6.0.5-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:e030047e85cbcedbfc073f71836d62dd5dadfbe7531cae27789ff66bc551bd5e"},

{file = "multidict-6.0.5-cp311-cp311-win32.whl", hash = "sha256:2f4848aa3baa109e6ab81fe2006c77ed4d3cd1e0ac2c1fbddb7b1277c168788c"},

{file = "multidict-6.0.5-cp311-cp311-win\_amd64.whl", hash = "sha256:2faa5ae9376faba05f630d7e5e6be05be22913782b927b19d12b8145968a85ea"},

{file = "multidict-6.0.5-cp312-cp312-macosx\_10\_9\_universal2.whl", hash = "sha256:51d035609b86722963404f711db441cf7134f1889107fb171a970c9701f92e1e"},

{file = "multidict-6.0.5-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:cbebcd5bcaf1eaf302617c114aa67569dd3f090dd0ce8ba9e35e9985b41ac35b"},

{file = "multidict-6.0.5-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:2ffc42c922dbfddb4a4c3b438eb056828719f07608af27d163191cb3e3aa6cc5"},

{file = "multidict-6.0.5-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:ceb3b7e6a0135e092de86110c5a74e46bda4bd4fbfeeb3a3bcec79c0f861e450"},

{file = "multidict-6.0.5-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:79660376075cfd4b2c80f295528aa6beb2058fd289f4c9252f986751a4cd0496"},

{file = "multidict-6.0.5-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:e4428b29611e989719874670fd152b6625500ad6c686d464e99f5aaeeaca175a"},

{file = "multidict-6.0.5-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:d84a5c3a5f7ce6db1f999fb9438f686bc2e09d38143f2d93d8406ed2dd6b9226"},

{file = "multidict-6.0.5-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:76c0de87358b192de7ea9649beb392f107dcad9ad27276324c24c91774ca5271"},

{file = "multidict-6.0.5-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:79a6d2ba910adb2cbafc95dad936f8b9386e77c84c35bc0add315b856d7c3abb"},

{file = "multidict-6.0.5-cp312-cp312-musllinux\_1\_1\_i686.whl", hash = "sha256:92d16a3e275e38293623ebf639c471d3e03bb20b8ebb845237e0d3664914caef"},

{file = "multidict-6.0.5-cp312-cp312-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:fb616be3538599e797a2017cccca78e354c767165e8858ab5116813146041a24"},

{file = "multidict-6.0.5-cp312-cp312-musllinux\_1\_1\_s390x.whl", hash = "sha256:14c2976aa9038c2629efa2c148022ed5eb4cb939e15ec7aace7ca932f48f9ba6"},

{file = "multidict-6.0.5-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:435a0984199d81ca178b9ae2c26ec3d49692d20ee29bc4c11a2a8d4514c67eda"},

{file = "multidict-6.0.5-cp312-cp312-win32.whl", hash = "sha256:9fe7b0653ba3d9d65cbe7698cca585bf0f8c83dbbcc710db9c90f478e175f2d5"},

{file = "multidict-6.0.5-cp312-cp312-win\_amd64.whl", hash = "sha256:01265f5e40f5a17f8241d52656ed27192be03bfa8764d88e8220141d1e4b3556"},

{file = "multidict-6.0.5-cp37-cp37m-macosx\_10\_9\_x86\_64.whl", hash = "sha256:19fe01cea168585ba0f678cad6f58133db2aa14eccaf22f88e4a6dccadfad8b3"},

{file = "multidict-6.0.5-cp37-cp37m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:6bf7a982604375a8d49b6cc1b781c1747f243d91b81035a9b43a2126c04766f5"},

{file = "multidict-6.0.5-cp37-cp37m-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:107c0cdefe028703fb5dafe640a409cb146d44a6ae201e55b35a4af8e95457dd"},

{file = "multidict-6.0.5-cp37-cp37m-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:403c0911cd5d5791605808b942c88a8155c2592e05332d2bf78f18697a5fa15e"},

{file = "multidict-6.0.5-cp37-cp37m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:aeaf541ddbad8311a87dd695ed9642401131ea39ad7bc8cf3ef3967fd093b626"},

{file = "multidict-6.0.5-cp37-cp37m-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:e4972624066095e52b569e02b5ca97dbd7a7ddd4294bf4e7247d52635630dd83"},

{file = "multidict-6.0.5-cp37-cp37m-musllinux\_1\_1\_aarch64.whl", hash = "sha256:d946b0a9eb8aaa590df1fe082cee553ceab173e6cb5b03239716338629c50c7a"},

{file = "multidict-6.0.5-cp37-cp37m-musllinux\_1\_1\_i686.whl", hash = "sha256:b55358304d7a73d7bdf5de62494aaf70bd33015831ffd98bc498b433dfe5b10c"},

{file = "multidict-6.0.5-cp37-cp37m-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:a3145cb08d8625b2d3fee1b2d596a8766352979c9bffe5d7833e0503d0f0b5e5"},

{file = "multidict-6.0.5-cp37-cp37m-musllinux\_1\_1\_s390x.whl", hash = "sha256:d65f25da8e248202bd47445cec78e0025c0fe7582b23ec69c3b27a640dd7a8e3"},

{file = "multidict-6.0.5-cp37-cp37m-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:c9bf56195c6bbd293340ea82eafd0071cb3d450c703d2c93afb89f93b8386ccc"},

{file = "multidict-6.0.5-cp37-cp37m-win32.whl", hash = "sha256:69db76c09796b313331bb7048229e3bee7928eb62bab5e071e9f7fcc4879caee"},

{file = "multidict-6.0.5-cp37-cp37m-win\_amd64.whl", hash = "sha256:fce28b3c8a81b6b36dfac9feb1de115bab619b3c13905b419ec71d03a3fc1423"},

{file = "multidict-6.0.5-cp38-cp38-macosx\_10\_9\_universal2.whl", hash = "sha256:76f067f5121dcecf0d63a67f29080b26c43c71a98b10c701b0677e4a065fbd54"},

{file = "multidict-6.0.5-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:b82cc8ace10ab5bd93235dfaab2021c70637005e1ac787031f4d1da63d493c1d"},

{file = "multidict-6.0.5-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:5cb241881eefd96b46f89b1a056187ea8e9ba14ab88ba632e68d7a2ecb7aadf7"},

{file = "multidict-6.0.5-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:e8e94e6912639a02ce173341ff62cc1201232ab86b8a8fcc05572741a5dc7d93"},

{file = "multidict-6.0.5-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:09a892e4a9fb47331da06948690ae38eaa2426de97b4ccbfafbdcbe5c8f37ff8"},

{file = "multidict-6.0.5-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:55205d03e8a598cfc688c71ca8ea5f66447164efff8869517f175ea632c7cb7b"},

{file = "multidict-6.0.5-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:37b15024f864916b4951adb95d3a80c9431299080341ab9544ed148091b53f50"},

{file = "multidict-6.0.5-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:f2a1dee728b52b33eebff5072817176c172050d44d67befd681609b4746e1c2e"},

{file = "multidict-6.0.5-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:edd08e6f2f1a390bf137080507e44ccc086353c8e98c657e666c017718561b89"},

{file = "multidict-6.0.5-cp38-cp38-musllinux\_1\_1\_i686.whl", hash = "sha256:60d698e8179a42ec85172d12f50b1668254628425a6bd611aba022257cac1386"},

{file = "multidict-6.0.5-cp38-cp38-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:3d25f19500588cbc47dc19081d78131c32637c25804df8414463ec908631e453"},

{file = "multidict-6.0.5-cp38-cp38-musllinux\_1\_1\_s390x.whl", hash = "sha256:4cc0ef8b962ac7a5e62b9e826bd0cd5040e7d401bc45a6835910ed699037a461"},

{file = "multidict-6.0.5-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:eca2e9d0cc5a889850e9bbd68e98314ada174ff6ccd1129500103df7a94a7a44"},

{file = "multidict-6.0.5-cp38-cp38-win32.whl", hash = "sha256:4a6a4f196f08c58c59e0b8ef8ec441d12aee4125a7d4f4fef000ccb22f8d7241"},

{file = "multidict-6.0.5-cp38-cp38-win\_amd64.whl", hash = "sha256:0275e35209c27a3f7951e1ce7aaf93ce0d163b28948444bec61dd7badc6d3f8c"},

{file = "multidict-6.0.5-cp39-cp39-macosx\_10\_9\_universal2.whl", hash = "sha256:e7be68734bd8c9a513f2b0cfd508802d6609da068f40dc57d4e3494cefc92929"},

{file = "multidict-6.0.5-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:1d9ea7a7e779d7a3561aade7d596649fbecfa5c08a7674b11b423783217933f9"},

{file = "multidict-6.0.5-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:ea1456df2a27c73ce51120fa2f519f1bea2f4a03a917f4a43c8707cf4cbbae1a"},

{file = "multidict-6.0.5-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:cf590b134eb70629e350691ecca88eac3e3b8b3c86992042fb82e3cb1830d5e1"},

{file = "multidict-6.0.5-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:5c0631926c4f58e9a5ccce555ad7747d9a9f8b10619621f22f9635f069f6233e"},

{file = "multidict-6.0.5-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:dce1c6912ab9ff5f179eaf6efe7365c1f425ed690b03341911bf4939ef2f3046"},

{file = "multidict-6.0.5-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:c0868d64af83169e4d4152ec612637a543f7a336e4a307b119e98042e852ad9c"},

{file = "multidict-6.0.5-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:141b43360bfd3bdd75f15ed811850763555a251e38b2405967f8e25fb43f7d40"},

{file = "multidict-6.0.5-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:7df704ca8cf4a073334e0427ae2345323613e4df18cc224f647f251e5e75a527"},

{file = "multidict-6.0.5-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:6214c5a5571802c33f80e6c84713b2c79e024995b9c5897f794b43e714daeec9"},

{file = "multidict-6.0.5-cp39-cp39-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:cd6c8fca38178e12c00418de737aef1261576bd1b6e8c6134d3e729a4e858b38"},

{file = "multidict-6.0.5-cp39-cp39-musllinux\_1\_1\_s390x.whl", hash = "sha256:e02021f87a5b6932fa6ce916ca004c4d441509d33bbdbeca70d05dff5e9d2479"},

{file = "multidict-6.0.5-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:ebd8d160f91a764652d3e51ce0d2956b38efe37c9231cd82cfc0bed2e40b581c"},

{file = "multidict-6.0.5-cp39-cp39-win32.whl", hash = "sha256:04da1bb8c8dbadf2a18a452639771951c662c5ad03aefe4884775454be322c9b"},

{file = "multidict-6.0.5-cp39-cp39-win\_amd64.whl", hash = "sha256:d6f6d4f185481c9669b9447bf9d9cf3b95a0e9df9d169bbc17e363b7d5487755"},

{file = "multidict-6.0.5-py3-none-any.whl", hash = "sha256:0d63c74e3d7ab26de115c49bffc92cc77ed23395303d496eae515d4204a625e7"},

{file = "multidict-6.0.5.tar.gz", hash = "sha256:f7e301075edaf50500f0b341543c41194d8df3ae5caf4702f2095f3ca73dd8da"},

]

[[package]]

name = "mypy-extensions"

version = "1.0.0"

description = "Type system extensions for programs checked with the mypy type checker."

optional = false

python-versions = ">=3.5"

files = [

{file = "mypy\_extensions-1.0.0-py3-none-any.whl", hash = "sha256:4392f6c0eb8a5668a69e23d168ffa70f0be9ccfd32b5cc2d26a34ae5b844552d"},

{file = "mypy\_extensions-1.0.0.tar.gz", hash = "sha256:75dbf8955dc00442a438fc4d0666508a9a97b6bd41aa2f0ffe9d2f2725af0782"},

]

[[package]]

name = "nodeenv"

version = "1.8.0"

description = "Node.js virtual environment builder"

optional = false

python-versions = ">=2.7,!=3.0.\*,!=3.1.\*,!=3.2.\*,!=3.3.\*,!=3.4.\*,!=3.5.\*,!=3.6.\*"

files = [

{file = "nodeenv-1.8.0-py2.py3-none-any.whl", hash = "sha256:df865724bb3c3adc86b3876fa209771517b0cfe596beff01a92700e0e8be4cec"},

{file = "nodeenv-1.8.0.tar.gz", hash = "sha256:d51e0c37e64fbf47d017feac3145cdbb58836d7eee8c6f6d3b6880c5456227d2"},

]

[package.dependencies]

setuptools = "\*"

[[package]]

name = "ogmios"

version = "1.0.6"

description = "Ogmios is a lightweight bridge interface for cardano-node. It offers a WebSockets API that enables local clients to speak Ouroboros' mini-protocols via JSON/RPC. ogmios-python is an Ogmios client written in Python designed for ease of use."

optional = false

python-versions = ">=3.10"

files = [

{file = "ogmios-1.0.6-py3-none-any.whl", hash = "sha256:536be82ccea0111feaf20ac1b583457562d58c9f1796e4422d57c494ce0e5a11"},

{file = "ogmios-1.0.6.tar.gz", hash = "sha256:fe5113267d7afccaee9bc4449743554375475256e56c45c8ff8b9d3919182163"},

]

[package.dependencies]

cachetools = "\*"

cardano-tools = "\*"

coloredlogs = "\*"

orjson = "\*"

pycardano = "\*"

pydantic = ">=2.0"

websockets = "\*"

[package.extras]

dev = ["black", "datamodel-code-generator", "flake8-pyproject", "isort", "sphinx", "sphinx-rtd-theme"]

testing = ["coverage[toml] (>=6.5)", "pytest"]

[[package]]

name = "opshin"

version = "0.21.1"

description = "A simple pythonic programming language for Smart Contracts on Cardano"

optional = false

python-versions = "<3.12,>=3.8"

files = [

{file = "opshin-0.21.1-py3-none-any.whl", hash = "sha256:6b8d96bb941bd66c331c397290219c081dae44293355151aa92e567f2b4ec61d"},

{file = "opshin-0.21.1.tar.gz", hash = "sha256:5485f899fea26102910ca72a67d0c71454b1430042aa1a4ac0f4fbe0263f4b63"},

]

[package.dependencies]

frozenlist2 = ">=1.0.0,<2.0.0"

ordered-set = ">=4.1.0,<5.0.0"

pluthon = ">=0.5.4,<0.6.0"

pycardano = ">=0.10.0,<0.11.0"

uplc = ">=0.6.9,<0.7.0"

[[package]]

name = "ordered-set"

version = "4.1.0"

description = "An OrderedSet is a custom MutableSet that remembers its order, so that every"

optional = false

python-versions = ">=3.7"

files = [

{file = "ordered-set-4.1.0.tar.gz", hash = "sha256:694a8e44c87657c59292ede72891eb91d34131f6531463aab3009191c77364a8"},

{file = "ordered\_set-4.1.0-py3-none-any.whl", hash = "sha256:046e1132c71fcf3330438a539928932caf51ddbc582496833e23de611de14562"},

]

[package.extras]

dev = ["black", "mypy", "pytest"]

[[package]]

name = "orjson"

version = "3.10.3"

description = "Fast, correct Python JSON library supporting dataclasses, datetimes, and numpy"

optional = false

python-versions = ">=3.8"

files = [

{file = "orjson-3.10.3-cp310-cp310-macosx\_10\_15\_x86\_64.macosx\_11\_0\_arm64.macosx\_10\_15\_universal2.whl", hash = "sha256:9fb6c3f9f5490a3eb4ddd46fc1b6eadb0d6fc16fb3f07320149c3286a1409dd8"},

{file = "orjson-3.10.3-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:252124b198662eee80428f1af8c63f7ff077c88723fe206a25df8dc57a57b1fa"},

{file = "orjson-3.10.3-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:9f3e87733823089a338ef9bbf363ef4de45e5c599a9bf50a7a9b82e86d0228da"},

{file = "orjson-3.10.3-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:c8334c0d87103bb9fbbe59b78129f1f40d1d1e8355bbed2ca71853af15fa4ed3"},

{file = "orjson-3.10.3-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:1952c03439e4dce23482ac846e7961f9d4ec62086eb98ae76d97bd41d72644d7"},

{file = "orjson-3.10.3-cp310-cp310-musllinux\_1\_2\_aarch64.whl", hash = "sha256:c0403ed9c706dcd2809f1600ed18f4aae50be263bd7112e54b50e2c2bc3ebd6d"},

{file = "orjson-3.10.3-cp310-cp310-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:382e52aa4270a037d41f325e7d1dfa395b7de0c367800b6f337d8157367bf3a7"},

{file = "orjson-3.10.3-cp310-none-win32.whl", hash = "sha256:be2aab54313752c04f2cbaab4515291ef5af8c2256ce22abc007f89f42f49109"},

{file = "orjson-3.10.3-cp310-none-win\_amd64.whl", hash = "sha256:416b195f78ae461601893f482287cee1e3059ec49b4f99479aedf22a20b1098b"},

{file = "orjson-3.10.3-cp311-cp311-macosx\_10\_15\_x86\_64.macosx\_11\_0\_arm64.macosx\_10\_15\_universal2.whl", hash = "sha256:73100d9abbbe730331f2242c1fc0bcb46a3ea3b4ae3348847e5a141265479700"},

{file = "orjson-3.10.3-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:544a12eee96e3ab828dbfcb4d5a0023aa971b27143a1d35dc214c176fdfb29b3"},

{file = "orjson-3.10.3-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:520de5e2ef0b4ae546bea25129d6c7c74edb43fc6cf5213f511a927f2b28148b"},

{file = "orjson-3.10.3-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:ccaa0a401fc02e8828a5bedfd80f8cd389d24f65e5ca3954d72c6582495b4bcf"},

{file = "orjson-3.10.3-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:9a7bc9e8bc11bac40f905640acd41cbeaa87209e7e1f57ade386da658092dc16"},

{file = "orjson-3.10.3-cp311-cp311-musllinux\_1\_2\_aarch64.whl", hash = "sha256:3582b34b70543a1ed6944aca75e219e1192661a63da4d039d088a09c67543b08"},

{file = "orjson-3.10.3-cp311-cp311-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:1c23dfa91481de880890d17aa7b91d586a4746a4c2aa9a145bebdbaf233768d5"},

{file = "orjson-3.10.3-cp311-none-win32.whl", hash = "sha256:1770e2a0eae728b050705206d84eda8b074b65ee835e7f85c919f5705b006c9b"},

{file = "orjson-3.10.3-cp311-none-win\_amd64.whl", hash = "sha256:93433b3c1f852660eb5abdc1f4dd0ced2be031ba30900433223b28ee0140cde5"},

{file = "orjson-3.10.3-cp312-cp312-macosx\_10\_15\_x86\_64.macosx\_11\_0\_arm64.macosx\_10\_15\_universal2.whl", hash = "sha256:a39aa73e53bec8d410875683bfa3a8edf61e5a1c7bb4014f65f81d36467ea098"},

{file = "orjson-3.10.3-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:0943a96b3fa09bee1afdfccc2cb236c9c64715afa375b2af296c73d91c23eab2"},

{file = "orjson-3.10.3-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:e852baafceff8da3c9defae29414cc8513a1586ad93e45f27b89a639c68e8176"},

{file = "orjson-3.10.3-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:18566beb5acd76f3769c1d1a7ec06cdb81edc4d55d2765fb677e3eaa10fa99e0"},

{file = "orjson-3.10.3-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:1bd2218d5a3aa43060efe649ec564ebedec8ce6ae0a43654b81376216d5ebd42"},

{file = "orjson-3.10.3-cp312-cp312-musllinux\_1\_2\_aarch64.whl", hash = "sha256:cf20465e74c6e17a104ecf01bf8cd3b7b252565b4ccee4548f18b012ff2f8069"},

{file = "orjson-3.10.3-cp312-cp312-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:ba7f67aa7f983c4345eeda16054a4677289011a478ca947cd69c0a86ea45e534"},

{file = "orjson-3.10.3-cp312-none-win32.whl", hash = "sha256:17e0713fc159abc261eea0f4feda611d32eabc35708b74bef6ad44f6c78d5ea0"},

{file = "orjson-3.10.3-cp312-none-win\_amd64.whl", hash = "sha256:4c895383b1ec42b017dd2c75ae8a5b862fc489006afde06f14afbdd0309b2af0"},

{file = "orjson-3.10.3-cp38-cp38-macosx\_10\_15\_x86\_64.macosx\_11\_0\_arm64.macosx\_10\_15\_universal2.whl", hash = "sha256:be2719e5041e9fb76c8c2c06b9600fe8e8584e6980061ff88dcbc2691a16d20d"},

{file = "orjson-3.10.3-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:cb0175a5798bdc878956099f5c54b9837cb62cfbf5d0b86ba6d77e43861bcec2"},

{file = "orjson-3.10.3-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:978be58a68ade24f1af7758626806e13cff7748a677faf95fbb298359aa1e20d"},

{file = "orjson-3.10.3-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:16bda83b5c61586f6f788333d3cf3ed19015e3b9019188c56983b5a299210eb5"},

{file = "orjson-3.10.3-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:4ad1f26bea425041e0a1adad34630c4825a9e3adec49079b1fb6ac8d36f8b754"},

{file = "orjson-3.10.3-cp38-cp38-musllinux\_1\_2\_aarch64.whl", hash = "sha256:9e253498bee561fe85d6325ba55ff2ff08fb5e7184cd6a4d7754133bd19c9195"},

{file = "orjson-3.10.3-cp38-cp38-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:0a62f9968bab8a676a164263e485f30a0b748255ee2f4ae49a0224be95f4532b"},

{file = "orjson-3.10.3-cp38-none-win32.whl", hash = "sha256:8d0b84403d287d4bfa9bf7d1dc298d5c1c5d9f444f3737929a66f2fe4fb8f134"},

{file = "orjson-3.10.3-cp38-none-win\_amd64.whl", hash = "sha256:8bc7a4df90da5d535e18157220d7915780d07198b54f4de0110eca6b6c11e290"},

{file = "orjson-3.10.3-cp39-cp39-macosx\_10\_15\_x86\_64.macosx\_11\_0\_arm64.macosx\_10\_15\_universal2.whl", hash = "sha256:9059d15c30e675a58fdcd6f95465c1522b8426e092de9fff20edebfdc15e1cb0"},

{file = "orjson-3.10.3-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:8d40c7f7938c9c2b934b297412c067936d0b54e4b8ab916fd1a9eb8f54c02294"},

{file = "orjson-3.10.3-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:d4a654ec1de8fdaae1d80d55cee65893cb06494e124681ab335218be6a0691e7"},

{file = "orjson-3.10.3-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:831c6ef73f9aa53c5f40ae8f949ff7681b38eaddb6904aab89dca4d85099cb78"},

{file = "orjson-3.10.3-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:99b880d7e34542db89f48d14ddecbd26f06838b12427d5a25d71baceb5ba119d"},

{file = "orjson-3.10.3-cp39-cp39-musllinux\_1\_2\_aarch64.whl", hash = "sha256:2e5e176c994ce4bd434d7aafb9ecc893c15f347d3d2bbd8e7ce0b63071c52e25"},

{file = "orjson-3.10.3-cp39-cp39-musllinux\_1\_2\_x86\_64.whl", hash = "sha256:b69a58a37dab856491bf2d3bbf259775fdce262b727f96aafbda359cb1d114d8"},

{file = "orjson-3.10.3-cp39-none-win32.whl", hash = "sha256:b8d4d1a6868cde356f1402c8faeb50d62cee765a1f7ffcfd6de732ab0581e063"},

{file = "orjson-3.10.3-cp39-none-win\_amd64.whl", hash = **"sha256:5102f50c5fc46d94f2033fe00d392588564378260d64377aec702f21a7a22912"},**

**{file = "orjson-3.10.3.tar.gz", hash = "sha256:2b166507acae7ba2f7c315dcf185a9111ad5e992ac81f2d507aac39193c2c818"},**

**]**

**[[package]]**

**name = "oscrypto"**

**version = "1.3.0"**

**description = "TLS (SSL) sockets, key generation, encryption, decryption, signing, verification and KDFs using the OS crypto libraries. Does not require a compiler, and relies on the OS for patching. Works on Windows, OS X and Linux/BSD."**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "oscrypto-1.3.0-py2.py3-none-any.whl", hash = "sha256:2b2f1d2d42ec152ca90ccb5682f3e051fb55986e1b170ebde472b133713e7085"},**

**{file = "oscrypto-1.3.0.tar.gz", hash = "sha256:6f5fef59cb5b3708321db7cca56aed8ad7e662853351e7991fcf60ec606d47a4"},**

**]**

**[package.dependencies]**

**asn1crypto = ">=1.5.1"**

**[[package]]**

**name = "packaging"**

**version = "24.0"**

**description = "Core utilities for Python packages"**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "packaging-24.0-py3-none-any.whl", hash = "sha256:2ddfb553fdf02fb784c234c7ba6ccc288296ceabec964ad2eae3777778130bc5"},**

**{file = "packaging-24.0.tar.gz", hash = "sha256:eb82c5e3e56209074766e6885bb04b8c38a0c015d0a30036ebe7ece34c9989e9"},**

**]**

**[[package]]**

**name = "pathspec"**

**version = "0.12.1"**

**description = "Utility library for gitignore style pattern matching of file paths."**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "pathspec-0.12.1-py3-none-any.whl", hash = "sha256:a0d503e138a4c123b27490a4f7beda6a01c6f288df0e4a8b79c7eb0dc7b4cc08"},**

**{file = "pathspec-0.12.1.tar.gz", hash = "sha256:a482d51503a1ab33b1c67a6c3813a26953dbdc71c31dacaef9a838c4e29f5712"},**

**]**

**[[package]]**

**name = "peewee"**

**version = "3.17.5"**

**description = "a little orm"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "peewee-3.17.5.tar.gz", hash = "sha256:e1b6a64192207fd3ddb4e1188054820f42aef0aadfa749e3981af3c119a76420"},**

**]**

**[[package]]**

**name = "pendulum"**

**version = "3.0.0"**

**description = "Python datetimes made easy"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "pendulum-3.0.0-cp310-cp310-macosx\_10\_12\_x86\_64.whl", hash = "sha256:2cf9e53ef11668e07f73190c805dbdf07a1939c3298b78d5a9203a86775d1bfd"},**

**{file = "pendulum-3.0.0-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:fb551b9b5e6059377889d2d878d940fd0bbb80ae4810543db18e6f77b02c5ef6"},**

**{file = "pendulum-3.0.0-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:6c58227ac260d5b01fc1025176d7b31858c9f62595737f350d22124a9a3ad82d"},**

**{file = "pendulum-3.0.0-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:60fb6f415fea93a11c52578eaa10594568a6716602be8430b167eb0d730f3332"},**

**{file = "pendulum-3.0.0-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:b69f6b4dbcb86f2c2fe696ba991e67347bcf87fe601362a1aba6431454b46bde"},**

**{file = "pendulum-3.0.0-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:138afa9c373ee450ede206db5a5e9004fd3011b3c6bbe1e57015395cd076a09f"},**

**{file = "pendulum-3.0.0-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:83d9031f39c6da9677164241fd0d37fbfc9dc8ade7043b5d6d62f56e81af8ad2"},**

**{file = "pendulum-3.0.0-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:0c2308af4033fa534f089595bcd40a95a39988ce4059ccd3dc6acb9ef14ca44a"},**

**{file = "pendulum-3.0.0-cp310-none-win\_amd64.whl", hash = "sha256:9a59637cdb8462bdf2dbcb9d389518c0263799189d773ad5c11db6b13064fa79"},**

**{file = "pendulum-3.0.0-cp311-cp311-macosx\_10\_12\_x86\_64.whl", hash = "sha256:3725245c0352c95d6ca297193192020d1b0c0f83d5ee6bb09964edc2b5a2d508"},**

**{file = "pendulum-3.0.0-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:6c035f03a3e565ed132927e2c1b691de0dbf4eb53b02a5a3c5a97e1a64e17bec"},**

**{file = "pendulum-3.0.0-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:597e66e63cbd68dd6d58ac46cb7a92363d2088d37ccde2dae4332ef23e95cd00"},**

**{file = "pendulum-3.0.0-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:99a0f8172e19f3f0c0e4ace0ad1595134d5243cf75985dc2233e8f9e8de263ca"},**

**{file = "pendulum-3.0.0-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:77d8839e20f54706aed425bec82a83b4aec74db07f26acd039905d1237a5e1d4"},**

**{file = "pendulum-3.0.0-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:afde30e8146292b059020fbc8b6f8fd4a60ae7c5e6f0afef937bbb24880bdf01"},**

**{file = "pendulum-3.0.0-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:660434a6fcf6303c4efd36713ca9212c753140107ee169a3fc6c49c4711c2a05"},**

**{file = "pendulum-3.0.0-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:dee9e5a48c6999dc1106eb7eea3e3a50e98a50651b72c08a87ee2154e544b33e"},**

**{file = "pendulum-3.0.0-cp311-none-win\_amd64.whl", hash = "sha256:d4cdecde90aec2d67cebe4042fd2a87a4441cc02152ed7ed8fb3ebb110b94ec4"},**

**{file = "pendulum-3.0.0-cp311-none-win\_arm64.whl", hash = "sha256:773c3bc4ddda2dda9f1b9d51fe06762f9200f3293d75c4660c19b2614b991d83"},**

**{file = "pendulum-3.0.0-cp312-cp312-macosx\_10\_12\_x86\_64.whl", hash = "sha256:409e64e41418c49f973d43a28afe5df1df4f1dd87c41c7c90f1a63f61ae0f1f7"},**

**{file = "pendulum-3.0.0-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:a38ad2121c5ec7c4c190c7334e789c3b4624798859156b138fcc4d92295835dc"},**

**{file = "pendulum-3.0.0-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:fde4d0b2024b9785f66b7f30ed59281bd60d63d9213cda0eb0910ead777f6d37"},**

**{file = "pendulum-3.0.0-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:4b2c5675769fb6d4c11238132962939b960fcb365436b6d623c5864287faa319"},**

**{file = "pendulum-3.0.0-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:8af95e03e066826f0f4c65811cbee1b3123d4a45a1c3a2b4fc23c4b0dff893b5"},**

**{file = "pendulum-3.0.0-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:2165a8f33cb15e06c67070b8afc87a62b85c5a273e3aaa6bc9d15c93a4920d6f"},**

**{file = "pendulum-3.0.0-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:ad5e65b874b5e56bd942546ea7ba9dd1d6a25121db1c517700f1c9de91b28518"},**

**{file = "pendulum-3.0.0-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:17fe4b2c844bbf5f0ece69cfd959fa02957c61317b2161763950d88fed8e13b9"},**

**{file = "pendulum-3.0.0-cp312-none-win\_amd64.whl", hash = "sha256:78f8f4e7efe5066aca24a7a57511b9c2119f5c2b5eb81c46ff9222ce11e0a7a5"},**

**{file = "pendulum-3.0.0-cp312-none-win\_arm64.whl", hash = "sha256:28f49d8d1e32aae9c284a90b6bb3873eee15ec6e1d9042edd611b22a94ac462f"},**

**{file = "pendulum-3.0.0-cp37-cp37m-macosx\_10\_12\_x86\_64.whl", hash = "sha256:d4e2512f4e1a4670284a153b214db9719eb5d14ac55ada5b76cbdb8c5c00399d"},**

**{file = "pendulum-3.0.0-cp37-cp37m-macosx\_11\_0\_arm64.whl", hash = "sha256:3d897eb50883cc58d9b92f6405245f84b9286cd2de6e8694cb9ea5cb15195a32"},**

**{file = "pendulum-3.0.0-cp37-cp37m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:2e169cc2ca419517f397811bbe4589cf3cd13fca6dc38bb352ba15ea90739ebb"},**

**{file = "pendulum-3.0.0-cp37-cp37m-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:f17c3084a4524ebefd9255513692f7e7360e23c8853dc6f10c64cc184e1217ab"},**

**{file = "pendulum-3.0.0-cp37-cp37m-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:826d6e258052715f64d05ae0fc9040c0151e6a87aae7c109ba9a0ed930ce4000"},**

**{file = "pendulum-3.0.0-cp37-cp37m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:d2aae97087872ef152a0c40e06100b3665d8cb86b59bc8471ca7c26132fccd0f"},**

**{file = "pendulum-3.0.0-cp37-cp37m-musllinux\_1\_1\_aarch64.whl", hash = "sha256:ac65eeec2250d03106b5e81284ad47f0d417ca299a45e89ccc69e36130ca8bc7"},**

**{file = "pendulum-3.0.0-cp37-cp37m-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:a5346d08f3f4a6e9e672187faa179c7bf9227897081d7121866358af369f44f9"},**

**{file = "pendulum-3.0.0-cp37-none-win\_amd64.whl", hash = "sha256:235d64e87946d8f95c796af34818c76e0f88c94d624c268693c85b723b698aa9"},**

**{file = "pendulum-3.0.0-cp38-cp38-macosx\_10\_12\_x86\_64.whl", hash = "sha256:6a881d9c2a7f85bc9adafcfe671df5207f51f5715ae61f5d838b77a1356e8b7b"},**

**{file = "pendulum-3.0.0-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:d7762d2076b9b1cb718a6631ad6c16c23fc3fac76cbb8c454e81e80be98daa34"},**

**{file = "pendulum-3.0.0-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:4e8e36a8130819d97a479a0e7bf379b66b3b1b520e5dc46bd7eb14634338df8c"},**

**{file = "pendulum-3.0.0-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:7dc843253ac373358ffc0711960e2dd5b94ab67530a3e204d85c6e8cb2c5fa10"},**

**{file = "pendulum-3.0.0-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:0a78ad3635d609ceb1e97d6aedef6a6a6f93433ddb2312888e668365908c7120"},**

**{file = "pendulum-3.0.0-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:b30a137e9e0d1f751e60e67d11fc67781a572db76b2296f7b4d44554761049d6"},**

**{file = "pendulum-3.0.0-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:c95984037987f4a457bb760455d9ca80467be792236b69d0084f228a8ada0162"},**

**{file = "pendulum-3.0.0-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:d29c6e578fe0f893766c0d286adbf0b3c726a4e2341eba0917ec79c50274ec16"},**

**{file = "pendulum-3.0.0-cp38-none-win\_amd64.whl", hash = "sha256:deaba8e16dbfcb3d7a6b5fabdd5a38b7c982809567479987b9c89572df62e027"},**

**{file = "pendulum-3.0.0-cp39-cp39-macosx\_10\_12\_x86\_64.whl", hash = "sha256:b11aceea5b20b4b5382962b321dbc354af0defe35daa84e9ff3aae3c230df694"},**

**{file = "pendulum-3.0.0-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:a90d4d504e82ad236afac9adca4d6a19e4865f717034fc69bafb112c320dcc8f"},**

**{file = "pendulum-3.0.0-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:825799c6b66e3734227756fa746cc34b3549c48693325b8b9f823cb7d21b19ac"},**

**{file = "pendulum-3.0.0-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:ad769e98dc07972e24afe0cff8d365cb6f0ebc7e65620aa1976fcfbcadc4c6f3"},**

**{file = "pendulum-3.0.0-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:a6fc26907eb5fb8cc6188cc620bc2075a6c534d981a2f045daa5f79dfe50d512"},**

**{file = "pendulum-3.0.0-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:0c717eab1b6d898c00a3e0fa7781d615b5c5136bbd40abe82be100bb06df7a56"},**

**{file = "pendulum-3.0.0-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:3ddd1d66d1a714ce43acfe337190be055cdc221d911fc886d5a3aae28e14b76d"},**

**{file = "pendulum-3.0.0-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:822172853d7a9cf6da95d7b66a16c7160cb99ae6df55d44373888181d7a06edc"},**

**{file = "pendulum-3.0.0-cp39-none-win\_amd64.whl", hash = "sha256:840de1b49cf1ec54c225a2a6f4f0784d50bd47f68e41dc005b7f67c7d5b5f3ae"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:3b1f74d1e6ffe5d01d6023870e2ce5c2191486928823196f8575dcc786e107b1"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-macosx\_11\_0\_arm64.whl", hash = "sha256:729e9f93756a2cdfa77d0fc82068346e9731c7e884097160603872686e570f07"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:e586acc0b450cd21cbf0db6bae386237011b75260a3adceddc4be15334689a9a"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:22e7944ffc1f0099a79ff468ee9630c73f8c7835cd76fdb57ef7320e6a409df4"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-musllinux\_1\_1\_aarch64.whl", hash = "sha256:fa30af36bd8e50686846bdace37cf6707bdd044e5cb6e1109acbad3277232e04"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:440215347b11914ae707981b9a57ab9c7b6983ab0babde07063c6ee75c0dc6e7"},**

**{file = "pendulum-3.0.0-pp310-pypy310\_pp73-win\_amd64.whl", hash = "sha256:314c4038dc5e6a52991570f50edb2f08c339debdf8cea68ac355b32c4174e820"},**

**{file = "pendulum-3.0.0-pp37-pypy37\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:5acb1d386337415f74f4d1955c4ce8d0201978c162927d07df8eb0692b2d8533"},**

**{file = "pendulum-3.0.0-pp37-pypy37\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a789e12fbdefaffb7b8ac67f9d8f22ba17a3050ceaaa635cd1cc4645773a4b1e"},**

**{file = "pendulum-3.0.0-pp37-pypy37\_pp73-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:860aa9b8a888e5913bd70d819306749e5eb488e6b99cd6c47beb701b22bdecf5"},**

**{file = "pendulum-3.0.0-pp37-pypy37\_pp73-musllinux\_1\_1\_aarch64.whl", hash = "sha256:5ebc65ea033ef0281368217fbf59f5cb05b338ac4dd23d60959c7afcd79a60a0"},**

**{file = "pendulum-3.0.0-pp37-pypy37\_pp73-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:d9fef18ab0386ef6a9ac7bad7e43ded42c83ff7ad412f950633854f90d59afa8"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:1c134ba2f0571d0b68b83f6972e2307a55a5a849e7dac8505c715c531d2a8795"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-macosx\_11\_0\_arm64.whl", hash = "sha256:385680812e7e18af200bb9b4a49777418c32422d05ad5a8eb85144c4a285907b"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:9eec91cd87c59fb32ec49eb722f375bd58f4be790cae11c1b70fac3ee4f00da0"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:4386bffeca23c4b69ad50a36211f75b35a4deb6210bdca112ac3043deb7e494a"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-musllinux\_1\_1\_aarch64.whl", hash = "sha256:dfbcf1661d7146d7698da4b86e7f04814221081e9fe154183e34f4c5f5fa3bf8"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:04a1094a5aa1daa34a6b57c865b25f691848c61583fb22722a4df5699f6bf74c"},**

**{file = "pendulum-3.0.0-pp38-pypy38\_pp73-win\_amd64.whl", hash = "sha256:5b0ec85b9045bd49dd3a3493a5e7ddfd31c36a2a60da387c419fa04abcaecb23"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:0a15b90129765b705eb2039062a6daf4d22c4e28d1a54fa260892e8c3ae6e157"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-macosx\_11\_0\_arm64.whl", hash = "sha256:bb8f6d7acd67a67d6fedd361ad2958ff0539445ef51cbe8cd288db4306503cd0"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:fd69b15374bef7e4b4440612915315cc42e8575fcda2a3d7586a0d88192d0c88"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:dc00f8110db6898360c53c812872662e077eaf9c75515d53ecc65d886eec209a"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-musllinux\_1\_1\_aarch64.whl", hash = "sha256:83a44e8b40655d0ba565a5c3d1365d27e3e6778ae2a05b69124db9e471255c4a"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:1a3604e9fbc06b788041b2a8b78f75c243021e0f512447806a6d37ee5214905d"},**

**{file = "pendulum-3.0.0-pp39-pypy39\_pp73-win\_amd64.whl", hash = "sha256:92c307ae7accebd06cbae4729f0ba9fa724df5f7d91a0964b1b972a22baa482b"},**

**{file = "pendulum-3.0.0.tar.gz", hash = "sha256:5d034998dea404ec31fae27af6b22cff1708f830a1ed7353be4d1019bb9f584e"},**

**]**

**[package.dependencies]**

**python-dateutil = ">=2.6"**

**tzdata = ">=2020.1"**

**[package.extras]**

**test = ["time-machine (>=2.6.0)"]**

**[[package]]**

**name = "pexpect"**

**version = "4.9.0"**

**description = "Pexpect allows easy control of interactive console applications."**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "pexpect-4.9.0-py2.py3-none-any.whl", hash = "sha256:7236d1e080e4936be2dc3e326cec0af72acf9212a7e1d060210e70a47e253523"},**

**{file = "pexpect-4.9.0.tar.gz", hash = "sha256:ee7d41123f3c9911050ea2c2dac107568dc43b2d3b0c7557a33212c398ead30f"},**

**]**

**[package.dependencies]**

**ptyprocess = ">=0.5"**

**[[package]]**

**name = "platformdirs"**

**version = "4.2.2"**

**description = "A small Python package for determining appropriate platform-specific dirs, e.g. a `user data dir`."**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "platformdirs-4.2.2-py3-none-any.whl", hash = "sha256:2d7a1657e36a80ea911db832a8a6ece5ee53d8de21edd5cc5879af6530b1bfee"},**

**{file = "platformdirs-4.2.2.tar.gz", hash = "sha256:38b7b51f512eed9e84a22788b4bce1de17c0adb134d6becb09836e37d8654cd3"},**

**]**

**[package.extras]**

**docs = ["furo (>=2023.9.10)", "proselint (>=0.13)", "sphinx (>=7.2.6)", "sphinx-autodoc-typehints (>=1.25.2)"]**

**test = ["appdirs (==1.4.4)", "covdefaults (>=2.3)", "pytest (>=7.4.3)", "pytest-cov (>=4.1)", "pytest-mock (>=3.12)"]**

**type = ["mypy (>=1.8)"]**

**[[package]]**

**name = "pluggy"**

**version = "1.5.0"**

**description = "plugin and hook calling mechanisms for python"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "pluggy-1.5.0-py3-none-any.whl", hash = "sha256:44e1ad92c8ca002de6377e165f3e0f1be63266ab4d554740532335b9d75ea669"},**

**{file = "pluggy-1.5.0.tar.gz", hash = "sha256:2cffa88e94fdc978c4c574f15f9e59b7f4201d439195c3715ca9e2486f1d0cf1"},**

**]**

**[package.extras]**

**dev = ["pre-commit", "tox"]**

**testing = ["pytest", "pytest-benchmark"]**

**[[package]]**

**name = "pluthon"**

**version = "0.5.4"**

**description = "Pluto-like programming language for Cardano Smart Contracts in Python"**

**optional = false**

**python-versions = ">=3"**

**files = [**

**{file = "pluthon-0.5.4-py3-none-any.whl", hash = "sha256:57254ba532d42833253abb8432ee18671127383c809eb4ef7114d6d0aca485fe"},**

**{file = "pluthon-0.5.4.tar.gz", hash = "sha256:4f7272674522bcedb46c4496a4030a09841d2ea896ba42f974c1a28a33ef8175"},**

**]**

**[package.dependencies]**

**ordered-set = ">=4.1.0,<5.0.0"**

**uplc = ">=0.6.2,<0.7.0"**

**[[package]]**

**name = "pprintpp"**

**version = "0.4.0"**

**description = "A drop-in replacement for pprint that's actually pretty"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "pprintpp-0.4.0-py2.py3-none-any.whl", hash = "sha256:b6b4dcdd0c0c0d75e4d7b2f21a9e933e5b2ce62b26e1a54537f9651ae5a5c01d"},**

**{file = "pprintpp-0.4.0.tar.gz", hash = "sha256:ea826108e2c7f49dc6d66c752973c3fc9749142a798d6b254e1e301cfdbc6403"},**

**]**

**[[package]]**

**name = "pre-commit"**

**version = "3.7.1"**

**description = "A framework for managing and maintaining multi-language pre-commit hooks."**

**optional = false**

**python-versions = ">=3.9"**

**files = [**

**{file = "pre\_commit-3.7.1-py2.py3-none-any.whl", hash = "sha256:fae36fd1d7ad7d6a5a1c0b0d5adb2ed1a3bda5a21bf6c3e5372073d7a11cd4c5"},**

**{file = "pre\_commit-3.7.1.tar.gz", hash = "sha256:8ca3ad567bc78a4972a3f1a477e94a79d4597e8140a6e0b651c5e33899c3654a"},**

**]**

**[package.dependencies]**

**cfgv = ">=2.0.0"**

**identify = ">=1.0.0"**

**nodeenv = ">=0.11.1"**

**pyyaml = ">=5.1"**

**virtualenv = ">=20.10.0"**

**[[package]]**

**name = "ptyprocess"**

**version = "0.7.0"**

**description = "Run a subprocess in a pseudo terminal"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "ptyprocess-0.7.0-py2.py3-none-any.whl", hash = "sha256:4b41f3967fce3af57cc7e94b888626c18bf37a083e3651ca8feeb66d492fef35"},**

**{file = "ptyprocess-0.7.0.tar.gz", hash = "sha256:5c5d0a3b48ceee0b48485e0c26037c0acd7d29765ca3fbb5cb3831d347423220"},**

**]**

**[[package]]**

**name = "pycardano"**

**version = "0.10.0"**

**description = "A Cardano library in Python"**

**optional = false**

**python-versions = "^3.7"**

**files = []**

**develop = false**

**[package.dependencies]**

**blockfrost-python = "0.5.3"**

**cachetools = "^5.3.0"**

**cbor2 = "^5.4.3"**

**cose = "0.9.dev8"**

**docker = "^6.1.3"**

**ECPy = "^1.2.5"**

**frozendict = "^2.3.8"**

**frozenlist = "^1.3.3"**

**mnemonic = "^0.20"**

**pprintpp = "^0.4.0"**

**PyNaCl = "^1.5.0"**

**typeguard = "^2.13.3"**

**websocket-client = "^1.4.1"**

**[package.source]**

**type = "git"**

**url = "https://github.com/steak-dev/pycardano"**

**reference = "running\_fixes"**

**resolved\_reference = "bdf8e9ce061c1a1c46585233207480b025fd0993"**

**[[package]]**

**name = "pycparser"**

**version = "2.22"**

**description = "C parser in Python"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "pycparser-2.22-py3-none-any.whl", hash = "sha256:c3702b6d3dd8c7abc1afa565d7e63d53a1d0bd86cdc24edd75470f4de499cfcc"},**

**{file = "pycparser-2.22.tar.gz", hash = "sha256:491c8be9c040f5390f5bf44a5b07752bd07f56edf992381b05c701439eec10f6"},**

**]**

**[[package]]**

**name = "pydantic"**

**version = "2.7.1"**

**description = "Data validation using Python type hints"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "pydantic-2.7.1-py3-none-any.whl", hash = "sha256:e029badca45266732a9a79898a15ae2e8b14840b1eabbb25844be28f0b33f3d5"},**

**{file = "pydantic-2.7.1.tar.gz", hash = "sha256:e9dbb5eada8abe4d9ae5f46b9939aead650cd2b68f249bb3a8139dbe125803cc"},**

**]**

**[package.dependencies]**

**annotated-types = ">=0.4.0"**

**pydantic-core = "2.18.2"**

**typing-extensions = ">=4.6.1"**

**[package.extras]**

**email = ["email-validator (>=2.0.0)"]**

**[[package]]**

**name = "pydantic-core"**

**version = "2.18.2"**

**description = "Core functionality for Pydantic validation and serialization"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "pydantic\_core-2.18.2-cp310-cp310-macosx\_10\_12\_x86\_64.whl", hash = "sha256:9e08e867b306f525802df7cd16c44ff5ebbe747ff0ca6cf3fde7f36c05a59a81"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:f0a21cbaa69900cbe1a2e7cad2aa74ac3cf21b10c3efb0fa0b80305274c0e8a2"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:0680b1f1f11fda801397de52c36ce38ef1c1dc841a0927a94f226dea29c3ae3d"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-manylinux\_2\_17\_armv7l.manylinux2014\_armv7l.whl", hash = "sha256:95b9d5e72481d3780ba3442eac863eae92ae43a5f3adb5b4d0a1de89d42bb250"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:c4fcf5cd9c4b655ad666ca332b9a081112cd7a58a8b5a6ca7a3104bc950f2038"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:9b5155ff768083cb1d62f3e143b49a8a3432e6789a3abee8acd005c3c7af1c74"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:553ef617b6836fc7e4df130bb851e32fe357ce36336d897fd6646d6058d980af"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:b89ed9eb7d616ef5714e5590e6cf7f23b02d0d539767d33561e3675d6f9e3857"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:75f7e9488238e920ab6204399ded280dc4c307d034f3924cd7f90a38b1829563"},**

**{file = "pydantic\_core-2.18.2-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:ef26c9e94a8c04a1b2924149a9cb081836913818e55681722d7f29af88fe7b38"},**

**{file = "pydantic\_core-2.18.2-cp310-none-win32.whl", hash = "sha256:182245ff6b0039e82b6bb585ed55a64d7c81c560715d1bad0cbad6dfa07b4027"},**

**{file = "pydantic\_core-2.18.2-cp310-none-win\_amd64.whl", hash = "sha256:e23ec367a948b6d812301afc1b13f8094ab7b2c280af66ef450efc357d2ae543"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-macosx\_10\_12\_x86\_64.whl", hash = "sha256:219da3f096d50a157f33645a1cf31c0ad1fe829a92181dd1311022f986e5fbe3"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:cc1cfd88a64e012b74e94cd00bbe0f9c6df57049c97f02bb07d39e9c852e19a4"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:05b7133a6e6aeb8df37d6f413f7705a37ab4031597f64ab56384c94d98fa0e90"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-manylinux\_2\_17\_armv7l.manylinux2014\_armv7l.whl", hash = "sha256:224c421235f6102e8737032483f43c1a8cfb1d2f45740c44166219599358c2cd"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:b14d82cdb934e99dda6d9d60dc84a24379820176cc4a0d123f88df319ae9c150"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:2728b01246a3bba6de144f9e3115b532ee44bd6cf39795194fb75491824a1413"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:470b94480bb5ee929f5acba6995251ada5e059a5ef3e0dfc63cca287283ebfa6"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:997abc4df705d1295a42f95b4eec4950a37ad8ae46d913caeee117b6b198811c"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:75250dbc5290e3f1a0f4618db35e51a165186f9034eff158f3d490b3fed9f8a0"},**

**{file = "pydantic\_core-2.18.2-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:4456f2dca97c425231d7315737d45239b2b51a50dc2b6f0c2bb181fce6207664"},**

**{file = "pydantic\_core-2.18.2-cp311-none-win32.whl", hash = "sha256:269322dcc3d8bdb69f054681edff86276b2ff972447863cf34c8b860f5188e2e"},**

**{file = "pydantic\_core-2.18.2-cp311-none-win\_amd64.whl", hash = "sha256:800d60565aec896f25bc3cfa56d2277d52d5182af08162f7954f938c06dc4ee3"},**

**{file = "pydantic\_core-2.18.2-cp311-none-win\_arm64.whl", hash = "sha256:1404c69d6a676245199767ba4f633cce5f4ad4181f9d0ccb0577e1f66cf4c46d"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-macosx\_10\_12\_x86\_64.whl", hash = "sha256:fb2bd7be70c0fe4dfd32c951bc813d9fe6ebcbfdd15a07527796c8204bd36242"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:6132dd3bd52838acddca05a72aafb6eab6536aa145e923bb50f45e78b7251043"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:d7d904828195733c183d20a54230c0df0eb46ec746ea1a666730787353e87182"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-manylinux\_2\_17\_armv7l.manylinux2014\_armv7l.whl", hash = "sha256:c9bd70772c720142be1020eac55f8143a34ec9f82d75a8e7a07852023e46617f"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:2b8ed04b3582771764538f7ee7001b02e1170223cf9b75dff0bc698fadb00cf3"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:e6dac87ddb34aaec85f873d737e9d06a3555a1cc1a8e0c44b7f8d5daeb89d86f"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:7ca4ae5a27ad7a4ee5170aebce1574b375de390bc01284f87b18d43a3984df72"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:886eec03591b7cf058467a70a87733b35f44707bd86cf64a615584fd72488b7c"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:ca7b0c1f1c983e064caa85f3792dd2fe3526b3505378874afa84baf662e12241"},**

**{file = "pydantic\_core-2.18.2-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:4b4356d3538c3649337df4074e81b85f0616b79731fe22dd11b99499b2ebbdf3"},**

**{file = "pydantic\_core-2.18.2-cp312-none-win32.whl", hash = "sha256:8b172601454f2d7701121bbec3425dd71efcb787a027edf49724c9cefc14c038"},**

**{file = "pydantic\_core-2.18.2-cp312-none-win\_amd64.whl", hash = "sha256:b1bd7e47b1558ea872bd16c8502c414f9e90dcf12f1395129d7bb42a09a95438"},**

**{file = "pydantic\_core-2.18.2-cp312-none-win\_arm64.whl", hash = "sha256:98758d627ff397e752bc339272c14c98199c613f922d4a384ddc07526c86a2ec"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-macosx\_10\_12\_x86\_64.whl", hash = "sha256:9fdad8e35f278b2c3eb77cbdc5c0a49dada440657bf738d6905ce106dc1de439"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:1d90c3265ae107f91a4f279f4d6f6f1d4907ac76c6868b27dc7fb33688cfb347"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:390193c770399861d8df9670fb0d1874f330c79caaca4642332df7c682bf6b91"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-manylinux\_2\_17\_armv7l.manylinux2014\_armv7l.whl", hash = "sha256:82d5d4d78e4448683cb467897fe24e2b74bb7b973a541ea1dcfec1d3cbce39fb"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:4774f3184d2ef3e14e8693194f661dea5a4d6ca4e3dc8e39786d33a94865cefd"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:d4d938ec0adf5167cb335acb25a4ee69a8107e4984f8fbd2e897021d9e4ca21b"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:e0e8b1be28239fc64a88a8189d1df7fad8be8c1ae47fcc33e43d4be15f99cc70"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:868649da93e5a3d5eacc2b5b3b9235c98ccdbfd443832f31e075f54419e1b96b"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:78363590ef93d5d226ba21a90a03ea89a20738ee5b7da83d771d283fd8a56761"},**

**{file = "pydantic\_core-2.18.2-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:852e966fbd035a6468fc0a3496589b45e2208ec7ca95c26470a54daed82a0788"},**

**{file = "pydantic\_core-2.18.2-cp38-none-win32.whl", hash = "sha256:6a46e22a707e7ad4484ac9ee9f290f9d501df45954184e23fc29408dfad61350"},**

**{file = "pydantic\_core-2.18.2-cp38-none-win\_amd64.whl", hash = "sha256:d91cb5ea8b11607cc757675051f61b3d93f15eca3cefb3e6c704a5d6e8440f4e"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-macosx\_10\_12\_x86\_64.whl", hash = "sha256:ae0a8a797a5e56c053610fa7be147993fe50960fa43609ff2a9552b0e07013e8"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:042473b6280246b1dbf530559246f6842b56119c2926d1e52b631bdc46075f2a"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:1a388a77e629b9ec814c1b1e6b3b595fe521d2cdc625fcca26fbc2d44c816804"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-manylinux\_2\_17\_armv7l.manylinux2014\_armv7l.whl", hash = "sha256:e25add29b8f3b233ae90ccef2d902d0ae0432eb0d45370fe315d1a5cf231004b"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:f459a5ce8434614dfd39bbebf1041952ae01da6bed9855008cb33b875cb024c0"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:eff2de745698eb46eeb51193a9f41d67d834d50e424aef27df2fcdee1b153845"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:a8309f67285bdfe65c372ea3722b7a5642680f3dba538566340a9d36e920b5f0"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:f93a8a2e3938ff656a7c1bc57193b1319960ac015b6e87d76c76bf14fe0244b4"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:22057013c8c1e272eb8d0eebc796701167d8377441ec894a8fed1af64a0bf399"},**

**{file = "pydantic\_core-2.18.2-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:cfeecd1ac6cc1fb2692c3d5110781c965aabd4ec5d32799773ca7b1456ac636b"},**

**{file = "pydantic\_core-2.18.2-cp39-none-win32.whl", hash = "sha256:0d69b4c2f6bb3e130dba60d34c0845ba31b69babdd3f78f7c0c8fae5021a253e"},**

**{file = "pydantic\_core-2.18.2-cp39-none-win\_amd64.whl", hash = "sha256:d9319e499827271b09b4e411905b24a426b8fb69464dfa1696258f53a3334641"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:a1874c6dd4113308bd0eb568418e6114b252afe44319ead2b4081e9b9521fe75"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-macosx\_11\_0\_arm64.whl", hash = "sha256:ccdd111c03bfd3666bd2472b674c6899550e09e9f298954cfc896ab92b5b0e6d"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:e18609ceaa6eed63753037fc06ebb16041d17d28199ae5aba0052c51449650a9"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:6e5c584d357c4e2baf0ff7baf44f4994be121e16a2c88918a5817331fc7599d7"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:43f0f463cf89ace478de71a318b1b4f05ebc456a9b9300d027b4b57c1a2064fb"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-musllinux\_1\_1\_aarch64.whl", hash = "sha256:e1b395e58b10b73b07b7cf740d728dd4ff9365ac46c18751bf8b3d8cca8f625a"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:0098300eebb1c837271d3d1a2cd2911e7c11b396eac9661655ee524a7f10587b"},**

**{file = "pydantic\_core-2.18.2-pp310-pypy310\_pp73-win\_amd64.whl", hash = "sha256:36789b70d613fbac0a25bb07ab3d9dba4d2e38af609c020cf4d888d165ee0bf3"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-macosx\_10\_12\_x86\_64.whl", hash = "sha256:3f9a801e7c8f1ef8718da265bba008fa121243dfe37c1cea17840b0944dfd72c"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-macosx\_11\_0\_arm64.whl", hash = "sha256:3a6515ebc6e69d85502b4951d89131ca4e036078ea35533bb76327f8424531ce"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:20aca1e2298c56ececfd8ed159ae4dde2df0781988c97ef77d5c16ff4bd5b400"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:223ee893d77a310a0391dca6df00f70bbc2f36a71a895cecd9a0e762dc37b349"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-manylinux\_2\_5\_i686.manylinux1\_i686.whl", hash = "sha256:2334ce8c673ee93a1d6a65bd90327588387ba073c17e61bf19b4fd97d688d63c"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-musllinux\_1\_1\_aarch64.whl", hash = "sha256:cbca948f2d14b09d20268cda7b0367723d79063f26c4ffc523af9042cad95592"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:b3ef08e20ec49e02d5c6717a91bb5af9b20f1805583cb0adfe9ba2c6b505b5ae"},**

**{file = "pydantic\_core-2.18.2-pp39-pypy39\_pp73-win\_amd64.whl", hash = "sha256:c6fdc8627910eed0c01aed6a390a252fe3ea6d472ee70fdde56273f198938374"},**

**{file = "pydantic\_core-2.18.2.tar.gz", hash = "sha256:2e29d20810dfc3043ee13ac7d9e25105799817683348823f305ab3f349b9386e"},**

**]**

**[package.dependencies]**

**typing-extensions = ">=4.6.0,<4.7.0 || >4.7.0"**

**[[package]]**

**name = "pynacl"**

**version = "1.5.0"**

**description = "Python binding to the Networking and Cryptography (NaCl) library"**

**optional = false**

**python-versions = ">=3.6"**

**files = [**

**{file = "PyNaCl-1.5.0-cp36-abi3-macosx\_10\_10\_universal2.whl", hash = "sha256:401002a4aaa07c9414132aaed7f6836ff98f59277a234704ff66878c2ee4a0d1"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.manylinux\_2\_24\_aarch64.whl", hash = "sha256:52cb72a79269189d4e0dc537556f4740f7f0a9ec41c1322598799b0bdad4ef92"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a36d4a9dda1f19ce6e03c9a784a2921a4b726b02e1c736600ca9c22029474394"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.manylinux\_2\_24\_x86\_64.whl", hash = "sha256:0c84947a22519e013607c9be43706dd42513f9e6ae5d39d3613ca1e142fba44d"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:06b8f6fa7f5de8d5d2f7573fe8c863c051225a27b61e6860fd047b1775807858"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-musllinux\_1\_1\_aarch64.whl", hash = "sha256:a422368fc821589c228f4c49438a368831cb5bbc0eab5ebe1d7fac9dded6567b"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:61f642bf2378713e2c2e1de73444a3778e5f0a38be6fee0fe532fe30060282ff"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-win32.whl", hash = "sha256:e46dae94e34b085175f8abb3b0aaa7da40767865ac82c928eeb9e57e1ea8a543"},**

**{file = "PyNaCl-1.5.0-cp36-abi3-win\_amd64.whl", hash = "sha256:20f42270d27e1b6a29f54032090b972d97f0a1b0948cc52392041ef7831fee93"},**

**{file = "PyNaCl-1.5.0.tar.gz", hash = "sha256:8ac7448f09ab85811607bdd21ec2464495ac8b7c66d146bf545b0f08fb9220ba"},**

**]**

**[package.dependencies]**

**cffi = ">=1.4.1"**

**[package.extras]**

**docs = ["sphinx (>=1.6.5)", "sphinx-rtd-theme"]**

**tests = ["hypothesis (>=3.27.0)", "pytest (>=3.2.1,!=3.3.0)"]**

**[[package]]**

**name = "pyreadline3"**

**version = "3.4.1"**

**description = "A python implementation of GNU readline."**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "pyreadline3-3.4.1-py3-none-any.whl", hash = "sha256:b0efb6516fd4fb07b45949053826a62fa4cb353db5be2bbb4a7aa1fdd1e345fb"},**

**{file = "pyreadline3-3.4.1.tar.gz", hash = "sha256:6f3d1f7b8a31ba32b73917cefc1f28cc660562f39aea8646d30bd6eff21f7bae"},**

**]**

**[[package]]**

**name = "pytest"**

**version = "7.4.4"**

**description = "pytest: simple powerful testing with Python"**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "pytest-7.4.4-py3-none-any.whl", hash = "sha256:b090cdf5ed60bf4c45261be03239c2c1c22df034fbffe691abe93cd80cea01d8"},**

**{file = "pytest-7.4.4.tar.gz", hash = "sha256:2cf0005922c6ace4a3e2ec8b4080eb0d9753fdc93107415332f50ce9e7994280"},**

**]**

**[package.dependencies]**

**colorama = {version = "\*", markers = "sys\_platform == \"win32\""}**

**exceptiongroup = {version = ">=1.0.0rc8", markers = "python\_version < \"3.11\""}**

**iniconfig = "\*"**

**packaging = "\*"**

**pluggy = ">=0.12,<2.0"**

**tomli = {version = ">=1.0.0", markers = "python\_version < \"3.11\""}**

**[package.extras]**

**testing = ["argcomplete", "attrs (>=19.2.0)", "hypothesis (>=3.56)", "mock", "nose", "pygments (>=2.7.2)", "requests", "setuptools", "xmlschema"]**

**[[package]]**

**name = "python-dateutil"**

**version = "2.9.0.post0"**

**description = "Extensions to the standard Python datetime module"**

**optional = false**

**python-versions = "!=3.0.\*,!=3.1.\*,!=3.2.\*,>=2.7"**

**files = [**

**{file = "python-dateutil-2.9.0.post0.tar.gz", hash = "sha256:37dd54208da7e1cd875388217d5e00ebd4179249f90fb72437e91a35459a0ad3"},**

**{file = "python\_dateutil-2.9.0.post0-py2.py3-none-any.whl", hash = "sha256:a8b2bc7bffae282281c8140a97d3aa9c14da0b136dfe83f850eea9a5f7470427"},**

**]**

**[package.dependencies]**

**six = ">=1.5"**

**[[package]]**

**name = "python-secp256k1-cardano"**

**version = "0.2.3"**

**description = "Ctypes Python3 FFI bindings for libsecp256k1 at commit hash ac83be33"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "python-secp256k1-cardano-0.2.3.tar.gz", hash = "sha256:429b33cc1815c9942f3e6e4861d03aa9dbd73dbb7cc3dbc818e121f41d9f4c39"},**

**{file = "python\_secp256k1\_cardano-0.2.3-py3-none-any.whl", hash = "sha256:b4c664a007eb339c509615ad54bd2ee05564879f17b0fef9a5cc619aedf3bfb8"},**

**]**

**[package.extras]**

**typing-extensions = ["typing-extensions (>=3.6)"]**

**[[package]]**

**name = "pywin32"**

**version = "306"**

**description = "Python for Window Extensions"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "pywin32-306-cp310-cp310-win32.whl", hash = "sha256:06d3420a5155ba65f0b72f2699b5bacf3109f36acbe8923765c22938a69dfc8d"},**

**{file = "pywin32-306-cp310-cp310-win\_amd64.whl", hash = "sha256:84f4471dbca1887ea3803d8848a1616429ac94a4a8d05f4bc9c5dcfd42ca99c8"},**

**{file = "pywin32-306-cp311-cp311-win32.whl", hash = "sha256:e65028133d15b64d2ed8f06dd9fbc268352478d4f9289e69c190ecd6818b6407"},**

**{file = "pywin32-306-cp311-cp311-win\_amd64.whl", hash = "sha256:a7639f51c184c0272e93f244eb24dafca9b1855707d94c192d4a0b4c01e1100e"},**

**{file = "pywin32-306-cp311-cp311-win\_arm64.whl", hash = "sha256:70dba0c913d19f942a2db25217d9a1b726c278f483a919f1abfed79c9cf64d3a"},**

**{file = "pywin32-306-cp312-cp312-win32.whl", hash = "sha256:383229d515657f4e3ed1343da8be101000562bf514591ff383ae940cad65458b"},**

**{file = "pywin32-306-cp312-cp312-win\_amd64.whl", hash = "sha256:37257794c1ad39ee9be652da0462dc2e394c8159dfd913a8a4e8eb6fd346da0e"},**

**{file = "pywin32-306-cp312-cp312-win\_arm64.whl", hash = "sha256:5821ec52f6d321aa59e2db7e0a35b997de60c201943557d108af9d4ae1ec7040"},**

**{file = "pywin32-306-cp37-cp37m-win32.whl", hash = "sha256:1c73ea9a0d2283d889001998059f5eaaba3b6238f767c9cf2833b13e6a685f65"},**

**{file = "pywin32-306-cp37-cp37m-win\_amd64.whl", hash = "sha256:72c5f621542d7bdd4fdb716227be0dd3f8565c11b280be6315b06ace35487d36"},**

**{file = "pywin32-306-cp38-cp38-win32.whl", hash = "sha256:e4c092e2589b5cf0d365849e73e02c391c1349958c5ac3e9d5ccb9a28e017b3a"},**

**{file = "pywin32-306-cp38-cp38-win\_amd64.whl", hash = "sha256:e8ac1ae3601bee6ca9f7cb4b5363bf1c0badb935ef243c4733ff9a393b1690c0"},**

**{file = "pywin32-306-cp39-cp39-win32.whl", hash = "sha256:e25fd5b485b55ac9c057f67d94bc203f3f6595078d1fb3b458c9c28b7153a802"},**

**{file = "pywin32-306-cp39-cp39-win\_amd64.whl", hash = "sha256:39b61c15272833b5c329a2989999dcae836b1eed650252ab1b7bfbe1d59f30f4"},**

**]**

**[[package]]**

**name = "pyyaml"**

**version = "6.0.1"**

**description = "YAML parser and emitter for Python"**

**optional = false**

**python-versions = ">=3.6"**

**files = [**

**{file = "PyYAML-6.0.1-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:d858aa552c999bc8a8d57426ed01e40bef403cd8ccdd0fc5f6f04a00414cac2a"},**

**{file = "PyYAML-6.0.1-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:fd66fc5d0da6d9815ba2cebeb4205f95818ff4b79c3ebe268e75d961704af52f"},**

**{file = "PyYAML-6.0.1-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:69b023b2b4daa7548bcfbd4aa3da05b3a74b772db9e23b982788168117739938"},**

**{file = "PyYAML-6.0.1-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:81e0b275a9ecc9c0c0c07b4b90ba548307583c125f54d5b6946cfee6360c733d"},**

**{file = "PyYAML-6.0.1-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:ba336e390cd8e4d1739f42dfe9bb83a3cc2e80f567d8805e11b46f4a943f5515"},**

**{file = "PyYAML-6.0.1-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:326c013efe8048858a6d312ddd31d56e468118ad4cdeda36c719bf5bb6192290"},**

**{file = "PyYAML-6.0.1-cp310-cp310-win32.whl", hash = "sha256:bd4af7373a854424dabd882decdc5579653d7868b8fb26dc7d0e99f823aa5924"},**

**{file = "PyYAML-6.0.1-cp310-cp310-win\_amd64.whl", hash = "sha256:fd1592b3fdf65fff2ad0004b5e363300ef59ced41c2e6b3a99d4089fa8c5435d"},**

**{file = "PyYAML-6.0.1-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:6965a7bc3cf88e5a1c3bd2e0b5c22f8d677dc88a455344035f03399034eb3007"},**

**{file = "PyYAML-6.0.1-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:f003ed9ad21d6a4713f0a9b5a7a0a79e08dd0f221aff4525a2be4c346ee60aab"},**

**{file = "PyYAML-6.0.1-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:42f8152b8dbc4fe7d96729ec2b99c7097d656dc1213a3229ca5383f973a5ed6d"},**

**{file = "PyYAML-6.0.1-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:062582fca9fabdd2c8b54a3ef1c978d786e0f6b3a1510e0ac93ef59e0ddae2bc"},**

**{file = "PyYAML-6.0.1-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:d2b04aac4d386b172d5b9692e2d2da8de7bfb6c387fa4f801fbf6fb2e6ba4673"},**

**{file = "PyYAML-6.0.1-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:e7d73685e87afe9f3b36c799222440d6cf362062f78be1013661b00c5c6f678b"},**

**{file = "PyYAML-6.0.1-cp311-cp311-win32.whl", hash = "sha256:1635fd110e8d85d55237ab316b5b011de701ea0f29d07611174a1b42f1444741"},**

**{file = "PyYAML-6.0.1-cp311-cp311-win\_amd64.whl", hash = "sha256:bf07ee2fef7014951eeb99f56f39c9bb4af143d8aa3c21b1677805985307da34"},**

**{file = "PyYAML-6.0.1-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:855fb52b0dc35af121542a76b9a84f8d1cd886ea97c84703eaa6d88e37a2ad28"},**

**{file = "PyYAML-6.0.1-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:40df9b996c2b73138957fe23a16a4f0ba614f4c0efce1e9406a184b6d07fa3a9"},**

**{file = "PyYAML-6.0.1-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a08c6f0fe150303c1c6b71ebcd7213c2858041a7e01975da3a99aed1e7a378ef"},**

**{file = "PyYAML-6.0.1-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:6c22bec3fbe2524cde73d7ada88f6566758a8f7227bfbf93a408a9d86bcc12a0"},**

**{file = "PyYAML-6.0.1-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:8d4e9c88387b0f5c7d5f281e55304de64cf7f9c0021a3525bd3b1c542da3b0e4"},**

**{file = "PyYAML-6.0.1-cp312-cp312-win32.whl", hash = "sha256:d483d2cdf104e7c9fa60c544d92981f12ad66a457afae824d146093b8c294c54"},**

**{file = "PyYAML-6.0.1-cp312-cp312-win\_amd64.whl", hash = "sha256:0d3304d8c0adc42be59c5f8a4d9e3d7379e6955ad754aa9d6ab7a398b59dd1df"},**

**{file = "PyYAML-6.0.1-cp36-cp36m-macosx\_10\_9\_x86\_64.whl", hash = "sha256:50550eb667afee136e9a77d6dc71ae76a44df8b3e51e41b77f6de2932bfe0f47"},**

**{file = "PyYAML-6.0.1-cp36-cp36m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:1fe35611261b29bd1de0070f0b2f47cb6ff71fa6595c077e42bd0c419fa27b98"},**

**{file = "PyYAML-6.0.1-cp36-cp36m-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:704219a11b772aea0d8ecd7058d0082713c3562b4e271b849ad7dc4a5c90c13c"},**

**{file = "PyYAML-6.0.1-cp36-cp36m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:afd7e57eddb1a54f0f1a974bc4391af8bcce0b444685d936840f125cf046d5bd"},**

**{file = "PyYAML-6.0.1-cp36-cp36m-win32.whl", hash = "sha256:fca0e3a251908a499833aa292323f32437106001d436eca0e6e7833256674585"},**

**{file = "PyYAML-6.0.1-cp36-cp36m-win\_amd64.whl", hash = "sha256:f22ac1c3cac4dbc50079e965eba2c1058622631e526bd9afd45fedd49ba781fa"},**

**{file = "PyYAML-6.0.1-cp37-cp37m-macosx\_10\_9\_x86\_64.whl", hash = "sha256:b1275ad35a5d18c62a7220633c913e1b42d44b46ee12554e5fd39c70a243d6a3"},**

**{file = "PyYAML-6.0.1-cp37-cp37m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:18aeb1bf9a78867dc38b259769503436b7c72f7a1f1f4c93ff9a17de54319b27"},**

**{file = "PyYAML-6.0.1-cp37-cp37m-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:596106435fa6ad000c2991a98fa58eeb8656ef2325d7e158344fb33864ed87e3"},**

**{file = "PyYAML-6.0.1-cp37-cp37m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:baa90d3f661d43131ca170712d903e6295d1f7a0f595074f151c0aed377c9b9c"},**

**{file = "PyYAML-6.0.1-cp37-cp37m-win32.whl", hash = "sha256:9046c58c4395dff28dd494285c82ba00b546adfc7ef001486fbf0324bc174fba"},**

**{file = "PyYAML-6.0.1-cp37-cp37m-win\_amd64.whl", hash = "sha256:4fb147e7a67ef577a588a0e2c17b6db51dda102c71de36f8549b6816a96e1867"},**

**{file = "PyYAML-6.0.1-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:1d4c7e777c441b20e32f52bd377e0c409713e8bb1386e1099c2415f26e479595"},**

**{file = "PyYAML-6.0.1-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a0cd17c15d3bb3fa06978b4e8958dcdc6e0174ccea823003a106c7d4d7899ac5"},**

**{file = "PyYAML-6.0.1-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:28c119d996beec18c05208a8bd78cbe4007878c6dd15091efb73a30e90539696"},**

**{file = "PyYAML-6.0.1-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:7e07cbde391ba96ab58e532ff4803f79c4129397514e1413a7dc761ccd755735"},**

**{file = "PyYAML-6.0.1-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:49a183be227561de579b4a36efbb21b3eab9651dd81b1858589f796549873dd6"},**

**{file = "PyYAML-6.0.1-cp38-cp38-win32.whl", hash = "sha256:184c5108a2aca3c5b3d3bf9395d50893a7ab82a38004c8f61c258d4428e80206"},**

**{file = "PyYAML-6.0.1-cp38-cp38-win\_amd64.whl", hash = "sha256:1e2722cc9fbb45d9b87631ac70924c11d3a401b2d7f410cc0e3bbf249f2dca62"},**

**{file = "PyYAML-6.0.1-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:9eb6caa9a297fc2c2fb8862bc5370d0303ddba53ba97e71f08023b6cd73d16a8"},**

**{file = "PyYAML-6.0.1-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:c8098ddcc2a85b61647b2590f825f3db38891662cfc2fc776415143f599bb859"},**

**{file = "PyYAML-6.0.1-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:5773183b6446b2c99bb77e77595dd486303b4faab2b086e7b17bc6bef28865f6"},**

**{file = "PyYAML-6.0.1-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:b786eecbdf8499b9ca1d697215862083bd6d2a99965554781d0d8d1ad31e13a0"},**

**{file = "PyYAML-6.0.1-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:bc1bf2925a1ecd43da378f4db9e4f799775d6367bdb94671027b73b393a7c42c"},**

**{file = "PyYAML-6.0.1-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:04ac92ad1925b2cff1db0cfebffb6ffc43457495c9b3c39d3fcae417d7125dc5"},**

**{file = "PyYAML-6.0.1-cp39-cp39-win32.whl", hash = "sha256:faca3bdcf85b2fc05d06ff3fbc1f83e1391b3e724afa3feba7d13eeab355484c"},**

**{file = "PyYAML-6.0.1-cp39-cp39-win\_amd64.whl", hash = "sha256:510c9deebc5c0225e8c96813043e62b680ba2f9c50a08d3724c7f28a747d1486"},**

**{file = "PyYAML-6.0.1.tar.gz", hash = "sha256:bfdf460b1736c775f2ba9f6a92bca30bc2095067b8a9d77876d1fad6cc3b4a43"},**

**]**

**[[package]]**

**name = "requests"**

**version = "2.31.0"**

**description = "Python HTTP for Humans."**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "requests-2.31.0-py3-none-any.whl", hash = "sha256:58cd2187c01e70e6e26505bca751777aa9f2ee0b7f4300988b709f44e013003f"},**

**{file = "requests-2.31.0.tar.gz", hash = "sha256:942c5a758f98d790eaed1a29cb6eefc7ffb0d1cf7af05c3d2791656dbd6ad1e1"},**

**]**

**[package.dependencies]**

**certifi = ">=2017.4.17"**

**charset-normalizer = ">=2,<4"**

**idna = ">=2.5,<4"**

**urllib3 = ">=1.21.1,<3"**

**[package.extras]**

**socks = ["PySocks (>=1.5.6,!=1.5.7)"]**

**use-chardet-on-py3 = ["chardet (>=3.0.2,<6)"]**

**[[package]]**

**name = "rply"**

**version = "0.7.8"**

**description = "A pure Python Lex/Yacc that works with RPython"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "rply-0.7.8-py2.py3-none-any.whl", hash = "sha256:28ffd11d656c48aeb8c508eb382acd6a0bd906662624b34388751732a27807e7"},**

**{file = "rply-0.7.8.tar.gz", hash = "sha256:2a808ac25a4580a9991fc304d64434e299a8fc75760574492f242cbb5bb301c9"},**

**]**

**[package.dependencies]**

**appdirs = "\*"**

**[[package]]**

**name = "setuptools"**

**version = "69.5.1"**

**description = "Easily download, build, install, upgrade, and uninstall Python packages"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "setuptools-69.5.1-py3-none-any.whl", hash = "sha256:c636ac361bc47580504644275c9ad802c50415c7522212252c033bd15f301f32"},**

**{file = "setuptools-69.5.1.tar.gz", hash = "sha256:6c1fccdac05a97e598fb0ae3bbed5904ccb317337a51139dcd51453611bbb987"},**

**]**

**[package.extras]**

**docs = ["furo", "jaraco.packaging (>=9.3)", "jaraco.tidelift (>=1.4)", "pygments-github-lexers (==0.0.5)", "rst.linker (>=1.9)", "sphinx (>=3.5)", "sphinx-favicon", "sphinx-inline-tabs", "sphinx-lint", "sphinx-notfound-page (>=1,<2)", "sphinx-reredirects", "sphinxcontrib-towncrier"]**

**testing = ["build[virtualenv]", "filelock (>=3.4.0)", "importlib-metadata", "ini2toml[lite] (>=0.9)", "jaraco.develop (>=7.21)", "jaraco.envs (>=2.2)", "jaraco.path (>=3.2.0)", "mypy (==1.9)", "packaging (>=23.2)", "pip (>=19.1)", "pytest (>=6,!=8.1.1)", "pytest-checkdocs (>=2.4)", "pytest-cov", "pytest-enabler (>=2.2)", "pytest-home (>=0.5)", "pytest-mypy", "pytest-perf", "pytest-ruff (>=0.2.1)", "pytest-timeout", "pytest-xdist (>=3)", "tomli", "tomli-w (>=1.0.0)", "virtualenv (>=13.0.0)", "wheel"]**

**testing-integration = ["build[virtualenv] (>=1.0.3)", "filelock (>=3.4.0)", "jaraco.envs (>=2.2)", "jaraco.path (>=3.2.0)", "packaging (>=23.2)", "pytest", "pytest-enabler", "pytest-xdist", "tomli", "virtualenv (>=13.0.0)", "wheel"]**

**[[package]]**

**name = "six"**

**version = "1.16.0"**

**description = "Python 2 and 3 compatibility utilities"**

**optional = false**

**python-versions = ">=2.7, !=3.0.\*, !=3.1.\*, !=3.2.\*"**

**files = [**

**{file = "six-1.16.0-py2.py3-none-any.whl", hash = "sha256:8abb2f1d86890a2dfb989f9a77cfcfd3e47c2a354b01111771326f8aa26e0254"},**

**{file = "six-1.16.0.tar.gz", hash = "sha256:1e61c37477a1626458e36f7b1d82aa5c9b094fa4802892072e49de9c60c4c926"},**

**]**

**[[package]]**

**name = "sniffio"**

**version = "1.3.1"**

**description = "Sniff out which async library your code is running under"**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "sniffio-1.3.1-py3-none-any.whl", hash = "sha256:2f6da418d1f1e0fddd844478f41680e794e6051915791a034ff65e5f100525a2"},**

**{file = "sniffio-1.3.1.tar.gz", hash = "sha256:f4324edc670a0f49750a81b895f35c3adb843cca46f0530f79fc1babb23789dc"},**

**]**

**[[package]]**

**name = "sortedcontainers"**

**version = "2.4.0"**

**description = "Sorted Containers -- Sorted List, Sorted Dict, Sorted Set"**

**optional = false**

**python-versions = "\*"**

**files = [**

**{file = "sortedcontainers-2.4.0-py2.py3-none-any.whl", hash = "sha256:a163dcaede0f1c021485e957a39245190e74249897e2ae4b2aa38595db237ee0"},**

**{file = "sortedcontainers-2.4.0.tar.gz", hash = "sha256:25caa5a06cc30b6b83d11423433f65d1f9d76c4c6a0c90e3379eaa43b9bfdb88"},**

**]**

**[[package]]**

**name = "starlette"**

**version = "0.36.3"**

**description = "The little ASGI library that shines."**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "starlette-0.36.3-py3-none-any.whl", hash = "sha256:13d429aa93a61dc40bf503e8c801db1f1bca3dc706b10ef2434a36123568f044"},**

**{file = "starlette-0.36.3.tar.gz", hash = "sha256:90a671733cfb35771d8cc605e0b679d23b992f8dcfad48cc60b38cb29aeb7080"},**

**]**

**[package.dependencies]**

**anyio = ">=3.4.0,<5"**

**[package.extras]**

**full = ["httpx (>=0.22.0)", "itsdangerous", "jinja2", "python-multipart (>=0.0.7)", "pyyaml"]**

**[[package]]**

**name = "termcolor"**

**version = "2.4.0"**

**description = "ANSI color formatting for output in terminal"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "termcolor-2.4.0-py3-none-any.whl", hash = "sha256:9297c0df9c99445c2412e832e882a7884038a25617c60cea2ad69488d4040d63"},**

**{file = "termcolor-2.4.0.tar.gz", hash = "sha256:aab9e56047c8ac41ed798fa36d892a37aca6b3e9159f3e0c24bc64a9b3ac7b7a"},**

**]**

**[package.extras]**

**tests = ["pytest", "pytest-cov"]**

**[[package]]**

**name = "tomli"**

**version = "2.0.1"**

**description = "A lil' TOML parser"**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "tomli-2.0.1-py3-none-any.whl", hash = "sha256:939de3e7a6161af0c887ef91b7d41a53e7c5a1ca976325f429cb46ea9bc30ecc"},**

**{file = "tomli-2.0.1.tar.gz", hash = "sha256:de526c12914f0c550d15924c62d72abc48d6fe7364aa87328337a31007fe8a4f"},**

**]**

**[[package]]**

**name = "typeguard"**

**version = "2.13.3"**

**description = "Run-time type checker for Python"**

**optional = false**

**python-versions = ">=3.5.3"**

**files = [**

**{file = "typeguard-2.13.3-py3-none-any.whl", hash = "sha256:5e3e3be01e887e7eafae5af63d1f36c849aaa94e3a0112097312aabfa16284f1"},**

**{file = "typeguard-2.13.3.tar.gz", hash = "sha256:00edaa8da3a133674796cf5ea87d9f4b4c367d77476e185e80251cc13dfbb8c4"},**

**]**

**[package.extras]**

**doc = ["sphinx-autodoc-typehints (>=1.2.0)", "sphinx-rtd-theme"]**

**test = ["mypy", "pytest", "typing-extensions"]**

**[[package]]**

**name = "typing-extensions"**

**version = "4.11.0"**

**description = "Backported and Experimental Type Hints for Python 3.8+"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "typing\_extensions-4.11.0-py3-none-any.whl", hash = "sha256:c1f94d72897edaf4ce775bb7558d5b79d8126906a14ea5ed1635921406c0387a"},**

**{file = "typing\_extensions-4.11.0.tar.gz", hash = "sha256:83f085bd5ca59c80295fc2a82ab5dac679cbe02b9f33f7d83af68e241bea51b0"},**

**]**

**[[package]]**

**name = "tzdata"**

**version = "2024.1"**

**description = "Provider of IANA time zone data"**

**optional = false**

**python-versions = ">=2"**

**files = [**

**{file = "tzdata-2024.1-py2.py3-none-any.whl", hash = "sha256:9068bc196136463f5245e51efda838afa15aaeca9903f49050dfa2679db4d252"},**

**{file = "tzdata-2024.1.tar.gz", hash = "sha256:2674120f8d891909751c38abcdfd386ac0a5a1127954fbc332af6b5ceae07efd"},**

**]**

**[[package]]**

**name = "uplc"**

**version = "0.6.9"**

**description = "Python implementation of untyped plutus language core"**

**optional = false**

**python-versions = ">=3.8,<3.12"**

**files = [**

**{file = "uplc-0.6.9-py3-none-any.whl", hash = "sha256:1815aef89ce5f169ef7ebc2ea4090e488fb4d84785a41c16e1114a0a73cea996"},**

**{file = "uplc-0.6.9.tar.gz", hash = "sha256:a4f3990d9a009718e719d33d03e448946b195b0c31d9bd4700df85c13b384616"},**

**]**

**[package.dependencies]**

**cbor2 = ">=5.4.6,<6.0.0"**

**frozendict = ">=2.3.8,<3.0.0"**

**frozenlist = ">=1.3.3,<2.0.0"**

**pycardano = ">=0.10.0,<0.11.0"**

**python-secp256k1-cardano = ">=0.2.3,<0.3.0"**

**rply = ">=0.7.8,<0.8.0"**

**[[package]]**

**name = "urllib3"**

**version = "2.2.1"**

**description = "HTTP library with thread-safe connection pooling, file post, and more."**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "urllib3-2.2.1-py3-none-any.whl", hash = "sha256:450b20ec296a467077128bff42b73080516e71b56ff59a60a02bef2232c4fa9d"},**

**{file = "urllib3-2.2.1.tar.gz", hash = "sha256:d0570876c61ab9e520d776c38acbbb5b05a776d3f9ff98a5c8fd5162a444cf19"},**

**]**

**[package.extras]**

**brotli = ["brotli (>=1.0.9)", "brotlicffi (>=0.8.0)"]**

**h2 = ["h2 (>=4,<5)"]**

**socks = ["pysocks (>=1.5.6,!=1.5.7,<2.0)"]**

**zstd = ["zstandard (>=0.18.0)"]**

**[[package]]**

**name = "uvicorn"**

**version = "0.27.1"**

**description = "The lightning-fast ASGI server."**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "uvicorn-0.27.1-py3-none-any.whl", hash = "sha256:5c89da2f3895767472a35556e539fd59f7edbe9b1e9c0e1c99eebeadc61838e4"},**

**{file = "uvicorn-0.27.1.tar.gz", hash = "sha256:3d9a267296243532db80c83a959a3400502165ade2c1338dea4e67915fd4745a"},**

**]**

**[package.dependencies]**

**click = ">=7.0"**

**h11 = ">=0.8"**

**typing-extensions = {version = ">=4.0", markers = "python\_version < \"3.11\""}**

**[package.extras]**

**standard = ["colorama (>=0.4)", "httptools (>=0.5.0)", "python-dotenv (>=0.13)", "pyyaml (>=5.1)", "uvloop (>=0.14.0,!=0.15.0,!=0.15.1)", "watchfiles (>=0.13)", "websockets (>=10.4)"]**

**[[package]]**

**name = "virtualenv"**

**version = "20.26.2"**

**description = "Virtual Python Environment builder"**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "virtualenv-20.26.2-py3-none-any.whl", hash = "sha256:a624db5e94f01ad993d476b9ee5346fdf7b9de43ccaee0e0197012dc838a0e9b"},**

**{file = "virtualenv-20.26.2.tar.gz", hash = "sha256:82bf0f4eebbb78d36ddaee0283d43fe5736b53880b8a8cdcd37390a07ac3741c"},**

**]**

**[package.dependencies]**

**distlib = ">=0.3.7,<1"**

**filelock = ">=3.12.2,<4"**

**platformdirs = ">=3.9.1,<5"**

**[package.extras]**

**docs = ["furo (>=2023.7.26)", "proselint (>=0.13)", "sphinx (>=7.1.2,!=7.3)", "sphinx-argparse (>=0.4)", "sphinxcontrib-towncrier (>=0.2.1a0)", "towncrier (>=23.6)"]**

**test = ["covdefaults (>=2.3)", "coverage (>=7.2.7)", "coverage-enable-subprocess (>=1)", "flaky (>=3.7)", "packaging (>=23.1)", "pytest (>=7.4)", "pytest-env (>=0.8.2)", "pytest-freezer (>=0.4.8)", "pytest-mock (>=3.11.1)", "pytest-randomly (>=3.12)", "pytest-timeout (>=2.1)", "setuptools (>=68)", "time-machine (>=2.10)"]**

**[[package]]**

**name = "websocket-client"**

**version = "1.8.0"**

**description = "WebSocket client for Python with low level API options"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "websocket\_client-1.8.0-py3-none-any.whl", hash = "sha256:17b44cc997f5c498e809b22cdf2d9c7a9e71c02c8cc2b6c56e7c2d1239bfa526"},**

**{file = "websocket\_client-1.8.0.tar.gz", hash = "sha256:3239df9f44da632f96012472805d40a23281a991027ce11d2f45a6f24ac4c3da"},**

**]**

**[package.extras]**

**docs = ["Sphinx (>=6.0)", "myst-parser (>=2.0.0)", "sphinx-rtd-theme (>=1.1.0)"]**

**optional = ["python-socks", "wsaccel"]**

**test = ["websockets"]**

**[[package]]**

**name = "websockets"**

**version = "12.0"**

**description = "An implementation of the WebSocket Protocol (RFC 6455 & 7692)"**

**optional = false**

**python-versions = ">=3.8"**

**files = [**

**{file = "websockets-12.0-cp310-cp310-macosx\_10\_9\_universal2.whl", hash = "sha256:d554236b2a2006e0ce16315c16eaa0d628dab009c33b63ea03f41c6107958374"},**

**{file = "websockets-12.0-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:2d225bb6886591b1746b17c0573e29804619c8f755b5598d875bb4235ea639be"},**

**{file = "websockets-12.0-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:eb809e816916a3b210bed3c82fb88eaf16e8afcf9c115ebb2bacede1797d2547"},**

**{file = "websockets-12.0-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:c588f6abc13f78a67044c6b1273a99e1cf31038ad51815b3b016ce699f0d75c2"},**

**{file = "websockets-12.0-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:5aa9348186d79a5f232115ed3fa9020eab66d6c3437d72f9d2c8ac0c6858c558"},**

**{file = "websockets-12.0-cp310-cp310-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:6350b14a40c95ddd53e775dbdbbbc59b124a5c8ecd6fbb09c2e52029f7a9f480"},**

**{file = "websockets-12.0-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:70ec754cc2a769bcd218ed8d7209055667b30860ffecb8633a834dde27d6307c"},**

**{file = "websockets-12.0-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:6e96f5ed1b83a8ddb07909b45bd94833b0710f738115751cdaa9da1fb0cb66e8"},**

**{file = "websockets-12.0-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:4d87be612cbef86f994178d5186add3d94e9f31cc3cb499a0482b866ec477603"},**

**{file = "websockets-12.0-cp310-cp310-win32.whl", hash = "sha256:befe90632d66caaf72e8b2ed4d7f02b348913813c8b0a32fae1cc5fe3730902f"},**

**{file = "websockets-12.0-cp310-cp310-win\_amd64.whl", hash = "sha256:363f57ca8bc8576195d0540c648aa58ac18cf85b76ad5202b9f976918f4219cf"},**

**{file = "websockets-12.0-cp311-cp311-macosx\_10\_9\_universal2.whl", hash = "sha256:5d873c7de42dea355d73f170be0f23788cf3fa9f7bed718fd2830eefedce01b4"},**

**{file = "websockets-12.0-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:3f61726cae9f65b872502ff3c1496abc93ffbe31b278455c418492016e2afc8f"},**

**{file = "websockets-12.0-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:ed2fcf7a07334c77fc8a230755c2209223a7cc44fc27597729b8ef5425aa61a3"},**

**{file = "websockets-12.0-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:8e332c210b14b57904869ca9f9bf4ca32f5427a03eeb625da9b616c85a3a506c"},**

**{file = "websockets-12.0-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:5693ef74233122f8ebab026817b1b37fe25c411ecfca084b29bc7d6efc548f45"},**

**{file = "websockets-12.0-cp311-cp311-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:6e9e7db18b4539a29cc5ad8c8b252738a30e2b13f033c2d6e9d0549b45841c04"},**

**{file = "websockets-12.0-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:6e2df67b8014767d0f785baa98393725739287684b9f8d8a1001eb2839031447"},**

**{file = "websockets-12.0-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:bea88d71630c5900690fcb03161ab18f8f244805c59e2e0dc4ffadae0a7ee0ca"},**

**{file = "websockets-12.0-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:dff6cdf35e31d1315790149fee351f9e52978130cef6c87c4b6c9b3baf78bc53"},**

**{file = "websockets-12.0-cp311-cp311-win32.whl", hash = "sha256:3e3aa8c468af01d70332a382350ee95f6986db479ce7af14d5e81ec52aa2b402"},**

**{file = "websockets-12.0-cp311-cp311-win\_amd64.whl", hash = "sha256:25eb766c8ad27da0f79420b2af4b85d29914ba0edf69f547cc4f06ca6f1d403b"},**

**{file = "websockets-12.0-cp312-cp312-macosx\_10\_9\_universal2.whl", hash = "sha256:0e6e2711d5a8e6e482cacb927a49a3d432345dfe7dea8ace7b5790df5932e4df"},**

**{file = "websockets-12.0-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:dbcf72a37f0b3316e993e13ecf32f10c0e1259c28ffd0a85cee26e8549595fbc"},**

**{file = "websockets-12.0-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:12743ab88ab2af1d17dd4acb4645677cb7063ef4db93abffbf164218a5d54c6b"},**

**{file = "websockets-12.0-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:7b645f491f3c48d3f8a00d1fce07445fab7347fec54a3e65f0725d730d5b99cb"},**

**{file = "websockets-12.0-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:9893d1aa45a7f8b3bc4510f6ccf8db8c3b62120917af15e3de247f0780294b92"},**

**{file = "websockets-12.0-cp312-cp312-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:1f38a7b376117ef7aff996e737583172bdf535932c9ca021746573bce40165ed"},**

**{file = "websockets-12.0-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:f764ba54e33daf20e167915edc443b6f88956f37fb606449b4a5b10ba42235a5"},**

**{file = "websockets-12.0-cp312-cp312-musllinux\_1\_1\_i686.whl", hash = "sha256:1e4b3f8ea6a9cfa8be8484c9221ec0257508e3a1ec43c36acdefb2a9c3b00aa2"},**

**{file = "websockets-12.0-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:9fdf06fd06c32205a07e47328ab49c40fc1407cdec801d698a7c41167ea45113"},**

**{file = "websockets-12.0-cp312-cp312-win32.whl", hash = "sha256:baa386875b70cbd81798fa9f71be689c1bf484f65fd6fb08d051a0ee4e79924d"},**

**{file = "websockets-12.0-cp312-cp312-win\_amd64.whl", hash = "sha256:ae0a5da8f35a5be197f328d4727dbcfafa53d1824fac3d96cdd3a642fe09394f"},**

**{file = "websockets-12.0-cp38-cp38-macosx\_10\_9\_universal2.whl", hash = "sha256:5f6ffe2c6598f7f7207eef9a1228b6f5c818f9f4d53ee920aacd35cec8110438"},**

**{file = "websockets-12.0-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:9edf3fc590cc2ec20dc9d7a45108b5bbaf21c0d89f9fd3fd1685e223771dc0b2"},**

**{file = "websockets-12.0-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:8572132c7be52632201a35f5e08348137f658e5ffd21f51f94572ca6c05ea81d"},**

**{file = "websockets-12.0-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:604428d1b87edbf02b233e2c207d7d528460fa978f9e391bd8aaf9c8311de137"},**

**{file = "websockets-12.0-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:1a9d160fd080c6285e202327aba140fc9a0d910b09e423afff4ae5cbbf1c7205"},**

**{file = "websockets-12.0-cp38-cp38-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:87b4aafed34653e465eb77b7c93ef058516cb5acf3eb21e42f33928616172def"},**

**{file = "websockets-12.0-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:b2ee7288b85959797970114deae81ab41b731f19ebcd3bd499ae9ca0e3f1d2c8"},**

**{file = "websockets-12.0-cp38-cp38-musllinux\_1\_1\_i686.whl", hash = "sha256:7fa3d25e81bfe6a89718e9791128398a50dec6d57faf23770787ff441d851967"},**

**{file = "websockets-12.0-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:a571f035a47212288e3b3519944f6bf4ac7bc7553243e41eac50dd48552b6df7"},**

**{file = "websockets-12.0-cp38-cp38-win32.whl", hash = "sha256:3c6cc1360c10c17463aadd29dd3af332d4a1adaa8796f6b0e9f9df1fdb0bad62"},**

**{file = "websockets-12.0-cp38-cp38-win\_amd64.whl", hash = "sha256:1bf386089178ea69d720f8db6199a0504a406209a0fc23e603b27b300fdd6892"},**

**{file = "websockets-12.0-cp39-cp39-macosx\_10\_9\_universal2.whl", hash = "sha256:ab3d732ad50a4fbd04a4490ef08acd0517b6ae6b77eb967251f4c263011a990d"},**

**{file = "websockets-12.0-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:a1d9697f3337a89691e3bd8dc56dea45a6f6d975f92e7d5f773bc715c15dde28"},**

**{file = "websockets-12.0-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:1df2fbd2c8a98d38a66f5238484405b8d1d16f929bb7a33ed73e4801222a6f53"},**

**{file = "websockets-12.0-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:23509452b3bc38e3a057382c2e941d5ac2e01e251acce7adc74011d7d8de434c"},**

**{file = "websockets-12.0-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:2e5fc14ec6ea568200ea4ef46545073da81900a2b67b3e666f04adf53ad452ec"},**

**{file = "websockets-12.0-cp39-cp39-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:46e71dbbd12850224243f5d2aeec90f0aaa0f2dde5aeeb8fc8df21e04d99eff9"},**

**{file = "websockets-12.0-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:b81f90dcc6c85a9b7f29873beb56c94c85d6f0dac2ea8b60d995bd18bf3e2aae"},**

**{file = "websockets-12.0-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:a02413bc474feda2849c59ed2dfb2cddb4cd3d2f03a2fedec51d6e959d9b608b"},**

**{file = "websockets-12.0-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:bbe6013f9f791944ed31ca08b077e26249309639313fff132bfbf3ba105673b9"},**

**{file = "websockets-12.0-cp39-cp39-win32.whl", hash = "sha256:cbe83a6bbdf207ff0541de01e11904827540aa069293696dd528a6640bd6a5f6"},**

**{file = "websockets-12.0-cp39-cp39-win\_amd64.whl", hash = "sha256:fc4e7fa5414512b481a2483775a8e8be7803a35b30ca805afa4998a84f9fd9e8"},**

**{file = "websockets-12.0-pp310-pypy310\_pp73-macosx\_10\_9\_x86\_64.whl", hash = "sha256:248d8e2446e13c1d4326e0a6a4e9629cb13a11195051a73acf414812700badbd"},**

**{file = "websockets-12.0-pp310-pypy310\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:f44069528d45a933997a6fef143030d8ca8042f0dfaad753e2906398290e2870"},**

**{file = "websockets-12.0-pp310-pypy310\_pp73-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:c4e37d36f0d19f0a4413d3e18c0d03d0c268ada2061868c1e6f5ab1a6d575077"},**

**{file = "websockets-12.0-pp310-pypy310\_pp73-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:3d829f975fc2e527a3ef2f9c8f25e553eb7bc779c6665e8e1d52aa22800bb38b"},**

**{file = "websockets-12.0-pp310-pypy310\_pp73-win\_amd64.whl", hash = "sha256:2c71bd45a777433dd9113847af751aae36e448bc6b8c361a566cb043eda6ec30"},**

**{file = "websockets-12.0-pp38-pypy38\_pp73-macosx\_10\_9\_x86\_64.whl", hash = "sha256:0bee75f400895aef54157b36ed6d3b308fcab62e5260703add87f44cee9c82a6"},**

**{file = "websockets-12.0-pp38-pypy38\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:423fc1ed29f7512fceb727e2d2aecb952c46aa34895e9ed96071821309951123"},**

**{file = "websockets-12.0-pp38-pypy38\_pp73-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:27a5e9964ef509016759f2ef3f2c1e13f403725a5e6a1775555994966a66e931"},**

**{file = "websockets-12.0-pp38-pypy38\_pp73-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:c3181df4583c4d3994d31fb235dc681d2aaad744fbdbf94c4802485ececdecf2"},**

**{file = "websockets-12.0-pp38-pypy38\_pp73-win\_amd64.whl", hash = "sha256:b067cb952ce8bf40115f6c19f478dc71c5e719b7fbaa511359795dfd9d1a6468"},**

**{file = "websockets-12.0-pp39-pypy39\_pp73-macosx\_10\_9\_x86\_64.whl", hash = "sha256:00700340c6c7ab788f176d118775202aadea7602c5cc6be6ae127761c16d6b0b"},**

**{file = "websockets-12.0-pp39-pypy39\_pp73-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:e469d01137942849cff40517c97a30a93ae79917752b34029f0ec72df6b46399"},**

**{file = "websockets-12.0-pp39-pypy39\_pp73-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:ffefa1374cd508d633646d51a8e9277763a9b78ae71324183693959cf94635a7"},**

**{file = "websockets-12.0-pp39-pypy39\_pp73-manylinux\_2\_5\_x86\_64.manylinux1\_x86\_64.manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:ba0cab91b3956dfa9f512147860783a1829a8d905ee218a9837c18f683239611"},**

**{file = "websockets-12.0-pp39-pypy39\_pp73-win\_amd64.whl", hash = "sha256:2cb388a5bfb56df4d9a406783b7f9dbefb888c09b71629351cc6b036e9259370"},**

**{file = "websockets-12.0-py3-none-any.whl", hash = "sha256:dc284bbc8d7c78a6c69e0c7325ab46ee5e40bb4d50e494d8131a07ef47500e9e"},**

**{file = "websockets-12.0.tar.gz", hash = "sha256:81df9cbcbb6c260de1e007e58c011bfebe2dafc8435107b0537f393dd38c8b1b"},**

**]**

**[[package]]**

**name = "yarl"**

**version = "1.9.4"**

**description = "Yet another URL library"**

**optional = false**

**python-versions = ">=3.7"**

**files = [**

**{file = "yarl-1.9.4-cp310-cp310-macosx\_10\_9\_universal2.whl", hash = "sha256:a8c1df72eb746f4136fe9a2e72b0c9dc1da1cbd23b5372f94b5820ff8ae30e0e"},**

**{file = "yarl-1.9.4-cp310-cp310-macosx\_10\_9\_x86\_64.whl", hash = "sha256:a3a6ed1d525bfb91b3fc9b690c5a21bb52de28c018530ad85093cc488bee2dd2"},**

**{file = "yarl-1.9.4-cp310-cp310-macosx\_11\_0\_arm64.whl", hash = "sha256:c38c9ddb6103ceae4e4498f9c08fac9b590c5c71b0370f98714768e22ac6fa66"},**

**{file = "yarl-1.9.4-cp310-cp310-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:d9e09c9d74f4566e905a0b8fa668c58109f7624db96a2171f21747abc7524234"},**

**{file = "yarl-1.9.4-cp310-cp310-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:b8477c1ee4bd47c57d49621a062121c3023609f7a13b8a46953eb6c9716ca392"},**

**{file = "yarl-1.9.4-cp310-cp310-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:d5ff2c858f5f6a42c2a8e751100f237c5e869cbde669a724f2062d4c4ef93551"},**

**{file = "yarl-1.9.4-cp310-cp310-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:357495293086c5b6d34ca9616a43d329317feab7917518bc97a08f9e55648455"},**

**{file = "yarl-1.9.4-cp310-cp310-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:54525ae423d7b7a8ee81ba189f131054defdb122cde31ff17477951464c1691c"},**

**{file = "yarl-1.9.4-cp310-cp310-musllinux\_1\_1\_aarch64.whl", hash = "sha256:801e9264d19643548651b9db361ce3287176671fb0117f96b5ac0ee1c3530d53"},**

**{file = "yarl-1.9.4-cp310-cp310-musllinux\_1\_1\_i686.whl", hash = "sha256:e516dc8baf7b380e6c1c26792610230f37147bb754d6426462ab115a02944385"},**

**{file = "yarl-1.9.4-cp310-cp310-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:7d5aaac37d19b2904bb9dfe12cdb08c8443e7ba7d2852894ad448d4b8f442863"},**

**{file = "yarl-1.9.4-cp310-cp310-musllinux\_1\_1\_s390x.whl", hash = "sha256:54beabb809ffcacbd9d28ac57b0db46e42a6e341a030293fb3185c409e626b8b"},**

**{file = "yarl-1.9.4-cp310-cp310-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:bac8d525a8dbc2a1507ec731d2867025d11ceadcb4dd421423a5d42c56818541"},**

**{file = "yarl-1.9.4-cp310-cp310-win32.whl", hash = "sha256:7855426dfbddac81896b6e533ebefc0af2f132d4a47340cee6d22cac7190022d"},**

**{file = "yarl-1.9.4-cp310-cp310-win\_amd64.whl", hash = "sha256:848cd2a1df56ddbffeb375535fb62c9d1645dde33ca4d51341378b3f5954429b"},**

**{file = "yarl-1.9.4-cp311-cp311-macosx\_10\_9\_universal2.whl", hash = "sha256:35a2b9396879ce32754bd457d31a51ff0a9d426fd9e0e3c33394bf4b9036b099"},**

**{file = "yarl-1.9.4-cp311-cp311-macosx\_10\_9\_x86\_64.whl", hash = "sha256:4c7d56b293cc071e82532f70adcbd8b61909eec973ae9d2d1f9b233f3d943f2c"},**

**{file = "yarl-1.9.4-cp311-cp311-macosx\_11\_0\_arm64.whl", hash = "sha256:d8a1c6c0be645c745a081c192e747c5de06e944a0d21245f4cf7c05e457c36e0"},**

**{file = "yarl-1.9.4-cp311-cp311-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:4b3c1ffe10069f655ea2d731808e76e0f452fc6c749bea04781daf18e6039525"},**

**{file = "yarl-1.9.4-cp311-cp311-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:549d19c84c55d11687ddbd47eeb348a89df9cb30e1993f1b128f4685cd0ebbf8"},**

**{file = "yarl-1.9.4-cp311-cp311-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:a7409f968456111140c1c95301cadf071bd30a81cbd7ab829169fb9e3d72eae9"},**

**{file = "yarl-1.9.4-cp311-cp311-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:e23a6d84d9d1738dbc6e38167776107e63307dfc8ad108e580548d1f2c587f42"},**

**{file = "yarl-1.9.4-cp311-cp311-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:d8b889777de69897406c9fb0b76cdf2fd0f31267861ae7501d93003d55f54fbe"},**

**{file = "yarl-1.9.4-cp311-cp311-musllinux\_1\_1\_aarch64.whl", hash = "sha256:03caa9507d3d3c83bca08650678e25364e1843b484f19986a527630ca376ecce"},**

**{file = "yarl-1.9.4-cp311-cp311-musllinux\_1\_1\_i686.whl", hash = "sha256:4e9035df8d0880b2f1c7f5031f33f69e071dfe72ee9310cfc76f7b605958ceb9"},**

**{file = "yarl-1.9.4-cp311-cp311-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:c0ec0ed476f77db9fb29bca17f0a8fcc7bc97ad4c6c1d8959c507decb22e8572"},**

**{file = "yarl-1.9.4-cp311-cp311-musllinux\_1\_1\_s390x.whl", hash = "sha256:ee04010f26d5102399bd17f8df8bc38dc7ccd7701dc77f4a68c5b8d733406958"},**

**{file = "yarl-1.9.4-cp311-cp311-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:49a180c2e0743d5d6e0b4d1a9e5f633c62eca3f8a86ba5dd3c471060e352ca98"},**

**{file = "yarl-1.9.4-cp311-cp311-win32.whl", hash = "sha256:81eb57278deb6098a5b62e88ad8281b2ba09f2f1147c4767522353eaa6260b31"},**

**{file = "yarl-1.9.4-cp311-cp311-win\_amd64.whl", hash = "sha256:d1d2532b340b692880261c15aee4dc94dd22ca5d61b9db9a8a361953d36410b1"},**

**{file = "yarl-1.9.4-cp312-cp312-macosx\_10\_9\_universal2.whl", hash = "sha256:0d2454f0aef65ea81037759be5ca9947539667eecebca092733b2eb43c965a81"},**

**{file = "yarl-1.9.4-cp312-cp312-macosx\_10\_9\_x86\_64.whl", hash = "sha256:44d8ffbb9c06e5a7f529f38f53eda23e50d1ed33c6c869e01481d3fafa6b8142"},**

**{file = "yarl-1.9.4-cp312-cp312-macosx\_11\_0\_arm64.whl", hash = "sha256:aaaea1e536f98754a6e5c56091baa1b6ce2f2700cc4a00b0d49eca8dea471074"},**

**{file = "yarl-1.9.4-cp312-cp312-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:3777ce5536d17989c91696db1d459574e9a9bd37660ea7ee4d3344579bb6f129"},**

**{file = "yarl-1.9.4-cp312-cp312-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:9fc5fc1eeb029757349ad26bbc5880557389a03fa6ada41703db5e068881e5f2"},**

**{file = "yarl-1.9.4-cp312-cp312-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:ea65804b5dc88dacd4a40279af0cdadcfe74b3e5b4c897aa0d81cf86927fee78"},**

**{file = "yarl-1.9.4-cp312-cp312-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:aa102d6d280a5455ad6a0f9e6d769989638718e938a6a0a2ff3f4a7ff8c62cc4"},**

**{file = "yarl-1.9.4-cp312-cp312-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:09efe4615ada057ba2d30df871d2f668af661e971dfeedf0c159927d48bbeff0"},**

**{file = "yarl-1.9.4-cp312-cp312-musllinux\_1\_1\_aarch64.whl", hash = "sha256:008d3e808d03ef28542372d01057fd09168419cdc8f848efe2804f894ae03e51"},**

**{file = "yarl-1.9.4-cp312-cp312-musllinux\_1\_1\_i686.whl", hash = "sha256:6f5cb257bc2ec58f437da2b37a8cd48f666db96d47b8a3115c29f316313654ff"},**

**{file = "yarl-1.9.4-cp312-cp312-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:992f18e0ea248ee03b5a6e8b3b4738850ae7dbb172cc41c966462801cbf62cf7"},**

**{file = "yarl-1.9.4-cp312-cp312-musllinux\_1\_1\_s390x.whl", hash = "sha256:0e9d124c191d5b881060a9e5060627694c3bdd1fe24c5eecc8d5d7d0eb6faabc"},**

**{file = "yarl-1.9.4-cp312-cp312-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:3986b6f41ad22988e53d5778f91855dc0399b043fc8946d4f2e68af22ee9ff10"},**

**{file = "yarl-1.9.4-cp312-cp312-win32.whl", hash = "sha256:4b21516d181cd77ebd06ce160ef8cc2a5e9ad35fb1c5930882baff5ac865eee7"},**

**{file = "yarl-1.9.4-cp312-cp312-win\_amd64.whl", hash = "sha256:a9bd00dc3bc395a662900f33f74feb3e757429e545d831eef5bb280252631984"},**

**{file = "yarl-1.9.4-cp37-cp37m-macosx\_10\_9\_x86\_64.whl", hash = "sha256:63b20738b5aac74e239622d2fe30df4fca4942a86e31bf47a81a0e94c14df94f"},**

**{file = "yarl-1.9.4-cp37-cp37m-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:d7d7f7de27b8944f1fee2c26a88b4dabc2409d2fea7a9ed3df79b67277644e17"},**

**{file = "yarl-1.9.4-cp37-cp37m-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:c74018551e31269d56fab81a728f683667e7c28c04e807ba08f8c9e3bba32f14"},**

**{file = "yarl-1.9.4-cp37-cp37m-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:ca06675212f94e7a610e85ca36948bb8fc023e458dd6c63ef71abfd482481aa5"},**

**{file = "yarl-1.9.4-cp37-cp37m-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:5aef935237d60a51a62b86249839b51345f47564208c6ee615ed2a40878dccdd"},**

**{file = "yarl-1.9.4-cp37-cp37m-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:2b134fd795e2322b7684155b7855cc99409d10b2e408056db2b93b51a52accc7"},**

**{file = "yarl-1.9.4-cp37-cp37m-musllinux\_1\_1\_aarch64.whl", hash = "sha256:d25039a474c4c72a5ad4b52495056f843a7ff07b632c1b92ea9043a3d9950f6e"},**

**{file = "yarl-1.9.4-cp37-cp37m-musllinux\_1\_1\_i686.whl", hash = "sha256:f7d6b36dd2e029b6bcb8a13cf19664c7b8e19ab3a58e0fefbb5b8461447ed5ec"},**

**{file = "yarl-1.9.4-cp37-cp37m-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:957b4774373cf6f709359e5c8c4a0af9f6d7875db657adb0feaf8d6cb3c3964c"},**

**{file = "yarl-1.9.4-cp37-cp37m-musllinux\_1\_1\_s390x.whl", hash = "sha256:d7eeb6d22331e2fd42fce928a81c697c9ee2d51400bd1a28803965883e13cead"},**

**{file = "yarl-1.9.4-cp37-cp37m-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:6a962e04b8f91f8c4e5917e518d17958e3bdee71fd1d8b88cdce74dd0ebbf434"},**

**{file = "yarl-1.9.4-cp37-cp37m-win32.whl", hash = "sha256:f3bc6af6e2b8f92eced34ef6a96ffb248e863af20ef4fde9448cc8c9b858b749"},**

**{file = "yarl-1.9.4-cp37-cp37m-win\_amd64.whl", hash = "sha256:ad4d7a90a92e528aadf4965d685c17dacff3df282db1121136c382dc0b6014d2"},**

**{file = "yarl-1.9.4-cp38-cp38-macosx\_10\_9\_universal2.whl", hash = "sha256:ec61d826d80fc293ed46c9dd26995921e3a82146feacd952ef0757236fc137be"},**

**{file = "yarl-1.9.4-cp38-cp38-macosx\_10\_9\_x86\_64.whl", hash = "sha256:8be9e837ea9113676e5754b43b940b50cce76d9ed7d2461df1af39a8ee674d9f"},**

**{file = "yarl-1.9.4-cp38-cp38-macosx\_11\_0\_arm64.whl", hash = "sha256:bef596fdaa8f26e3d66af846bbe77057237cb6e8efff8cd7cc8dff9a62278bbf"},**

**{file = "yarl-1.9.4-cp38-cp38-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:2d47552b6e52c3319fede1b60b3de120fe83bde9b7bddad11a69fb0af7db32f1"},**

**{file = "yarl-1.9.4-cp38-cp38-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:84fc30f71689d7fc9168b92788abc977dc8cefa806909565fc2951d02f6b7d57"},**

**{file = "yarl-1.9.4-cp38-cp38-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:4aa9741085f635934f3a2583e16fcf62ba835719a8b2b28fb2917bb0537c1dfa"},**

**{file = "yarl-1.9.4-cp38-cp38-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:206a55215e6d05dbc6c98ce598a59e6fbd0c493e2de4ea6cc2f4934d5a18d130"},**

**{file = "yarl-1.9.4-cp38-cp38-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:07574b007ee20e5c375a8fe4a0789fad26db905f9813be0f9fef5a68080de559"},**

**{file = "yarl-1.9.4-cp38-cp38-musllinux\_1\_1\_aarch64.whl", hash = "sha256:5a2e2433eb9344a163aced6a5f6c9222c0786e5a9e9cac2c89f0b28433f56e23"},**

**{file = "yarl-1.9.4-cp38-cp38-musllinux\_1\_1\_i686.whl", hash = "sha256:6ad6d10ed9b67a382b45f29ea028f92d25bc0bc1daf6c5b801b90b5aa70fb9ec"},**

**{file = "yarl-1.9.4-cp38-cp38-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:6fe79f998a4052d79e1c30eeb7d6c1c1056ad33300f682465e1b4e9b5a188b78"},**

**{file = "yarl-1.9.4-cp38-cp38-musllinux\_1\_1\_s390x.whl", hash = "sha256:a825ec844298c791fd28ed14ed1bffc56a98d15b8c58a20e0e08c1f5f2bea1be"},**

**{file = "yarl-1.9.4-cp38-cp38-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:8619d6915b3b0b34420cf9b2bb6d81ef59d984cb0fde7544e9ece32b4b3043c3"},**

**{file = "yarl-1.9.4-cp38-cp38-win32.whl", hash = "sha256:686a0c2f85f83463272ddffd4deb5e591c98aac1897d65e92319f729c320eece"},**

**{file = "yarl-1.9.4-cp38-cp38-win\_amd64.whl", hash = "sha256:a00862fb23195b6b8322f7d781b0dc1d82cb3bcac346d1e38689370cc1cc398b"},**

**{file = "yarl-1.9.4-cp39-cp39-macosx\_10\_9\_universal2.whl", hash = "sha256:604f31d97fa493083ea21bd9b92c419012531c4e17ea6da0f65cacdcf5d0bd27"},**

**{file = "yarl-1.9.4-cp39-cp39-macosx\_10\_9\_x86\_64.whl", hash = "sha256:8a854227cf581330ffa2c4824d96e52ee621dd571078a252c25e3a3b3d94a1b1"},**

**{file = "yarl-1.9.4-cp39-cp39-macosx\_11\_0\_arm64.whl", hash = "sha256:ba6f52cbc7809cd8d74604cce9c14868306ae4aa0282016b641c661f981a6e91"},**

**{file = "yarl-1.9.4-cp39-cp39-manylinux\_2\_17\_aarch64.manylinux2014\_aarch64.whl", hash = "sha256:a6327976c7c2f4ee6816eff196e25385ccc02cb81427952414a64811037bbc8b"},**

**{file = "yarl-1.9.4-cp39-cp39-manylinux\_2\_17\_ppc64le.manylinux2014\_ppc64le.whl", hash = "sha256:8397a3817d7dcdd14bb266283cd1d6fc7264a48c186b986f32e86d86d35fbac5"},**

**{file = "yarl-1.9.4-cp39-cp39-manylinux\_2\_17\_s390x.manylinux2014\_s390x.whl", hash = "sha256:e0381b4ce23ff92f8170080c97678040fc5b08da85e9e292292aba67fdac6c34"},**

**{file = "yarl-1.9.4-cp39-cp39-manylinux\_2\_17\_x86\_64.manylinux2014\_x86\_64.whl", hash = "sha256:23d32a2594cb5d565d358a92e151315d1b2268bc10f4610d098f96b147370136"},**

**{file = "yarl-1.9.4-cp39-cp39-manylinux\_2\_5\_i686.manylinux1\_i686.manylinux\_2\_17\_i686.manylinux2014\_i686.whl", hash = "sha256:ddb2a5c08a4eaaba605340fdee8fc08e406c56617566d9643ad8bf6852778fc7"},**

**{file = "yarl-1.9.4-cp39-cp39-musllinux\_1\_1\_aarch64.whl", hash = "sha256:26a1dc6285e03f3cc9e839a2da83bcbf31dcb0d004c72d0730e755b33466c30e"},**

**{file = "yarl-1.9.4-cp39-cp39-musllinux\_1\_1\_i686.whl", hash = "sha256:18580f672e44ce1238b82f7fb87d727c4a131f3a9d33a5e0e82b793362bf18b4"},**

**{file = "yarl-1.9.4-cp39-cp39-musllinux\_1\_1\_ppc64le.whl", hash = "sha256:29e0f83f37610f173eb7e7b5562dd71467993495e568e708d99e9d1944f561ec"},**

**{file = "yarl-1.9.4-cp39-cp39-musllinux\_1\_1\_s390x.whl", hash = "sha256:1f23e4fe1e8794f74b6027d7cf19dc25f8b63af1483d91d595d4a07eca1fb26c"},**

**{file = "yarl-1.9.4-cp39-cp39-musllinux\_1\_1\_x86\_64.whl", hash = "sha256:db8e58b9d79200c76956cefd14d5c90af54416ff5353c5bfd7cbe58818e26ef0"},**

**{file = "yarl-1.9.4-cp39-cp39-win32.whl", hash = "sha256:c7224cab95645c7ab53791022ae77a4509472613e839dab722a72abe5a684575"},**

**{file = "yarl-1.9.4-cp39-cp39-win\_amd64.whl", hash = "sha256:824d6c50492add5da9374875ce72db7a0733b29c2394890aef23d533106e2b15"},**

**{file = "yarl-1.9.4-py3-none-any.whl", hash = "sha256:928cecb0ef9d5a7946eb6ff58417ad2fe9375762382f1bf5c55e61645f2c43ad"},**

**{file = "yarl-1.9.4.tar.gz", hash = "sha256:566db86717cf8080b99b58b083b773a908ae40f06681e87e589a976faf8246bf"},**

**]**

**[package.dependencies]**

**idna = ">=2.0"**

**multidict = ">=4.0"**

**[metadata]**

**lock-version = "2.0"**

**python-versions = ">=3.10, <3.12"**

**content-hash = "338842123758aeeb285666152378459eda5edc3b225d1b5d00b3fe780f9b4641"**

### **-Pass/Fail**

#### **Issues**

Based on the rules of the STEAK protocol a simple side chain should be added to allow easy consumption by third parties

* Minting policy only allows minting for specific token with latest random hash
* Accompanied by datum that describes the latest slot for which this hash is valid
* Locked at script that only allows burning of token after some time (i.e. 1 day)

This can then be operated as a service for any chain consumer and allows for easy consumption of the random hash until 1 day after generation

## **2. Test Case Name: Aiken STD Library**

### **2.1 rational.ak** rational.ak

Results:  
pub fn mul(left: Rational, right: Rational) -> Rational {

let Rational { numerator: a\_n, denominator: a\_d } = left

let Rational { numerator: b\_n, denominator: b\_d } = right

Rational { numerator: a\_n \* b\_n, denominator: a\_d \* b\_d }

}

test mul\_1() {

mul(ratio(2, 3), ratio(3, 4)) == ratio(6, 12)

}

test mul\_2() {

mul(ratio(-2, 3), ratio(-3, 4)) == ratio(6, 12)

}

test mul\_3() {

let result =

ratio(2, 5)

|> mul(ratio(1, 8))

|> mul(ratio(3, 10))

|> mul(ratio(21, 100))

|> mul(ratio(3, 5))

|> mul(ratio(2, 8))

|> mul(ratio(4, 10))

|> mul(ratio(22, 100))

|> reduce

result == ratio(2079, 50000000)

}

Tests:  
test geometric\_mean1() {

expect Some(x) = new(1, 2)

expect Some(y) = new(1, 2)

geometric\_mean(x, y) == new(1, 2)

}

test geometric\_mean2() {

expect Some(x) = new(-1, 2)

expect Some(y) = new(1, 2)

geometric\_mean(x, y) == None

}

test geometric\_mean3() {

expect Some(x) = new(1, 2)

expect Some(y) = new(-1, 2)

geometric\_mean(x, y) == None

}

test geometric\_mean4() {

expect Some(x) = new(1, 3)

expect Some(y) = new(1, 6)

geometric\_mean(x, y) == new(1, 4)

}

test geometric\_mean5() {

expect Some(x) = new(67, 2500)

expect Some(y) = new(35331, 1000)

expect Some(yi) = reciprocal(y)

geometric\_mean(x, yi) == new(258, 9398)

}

### 2.2. Transaction Function

#### #### certificate.ak

use aiken/hash.{Blake2b\_224, Hash}

use aiken/transaction/credential.{PoolId, StakeCredential, VerificationKey}

/// An on-chain certificate attesting of some operation. Publishing

/// certificates / triggers different kind of rules; most of the time,

/// they require signatures from / specific keys.

pub type Certificate {

CredentialRegistration { delegator: StakeCredential }

CredentialDeregistration { delegator: StakeCredential }

CredentialDelegation { delegator: StakeCredential, delegatee: PoolId }

PoolRegistration { pool\_id: PoolId, vrf: Hash<Blake2b\_224, VerificationKey> }

PoolDeregistration { pool\_id: PoolId, epoch: Int }

Governance

TreasuryMovement

}

#### #### credential.ak

use aiken/builtin

use aiken/hash.{Blake2b\_224, Hash}

/// A general structure for representing an on-chain `Credential`.

///

/// Credentials are always one of two kinds: a direct public/private key

/// pair, or a script (native or Plutus).

pub type Credential {

VerificationKeyCredential(Hash<Blake2b\_224, VerificationKey>)

ScriptCredential(Hash<Blake2b\_224, Script>)

}

/// A Cardano `Address` typically holding one or two credential references.

///

/// Note that legacy bootstrap addresses (a.k.a. 'Byron addresses') are

/// completely excluded from Plutus contexts. Thus, from an on-chain

/// perspective only exists addresses of type 00, 01, ..., 07 as detailed

/// in [CIP-0019 :: Shelley Addresses](https://github.com/cardano-foundation/CIPs/tree/master/CIP-0019/#shelley-addresses).

pub type Address {

payment\_credential: PaymentCredential,

stake\_credential: Option<StakeCredential>,

}

/// Smart-constructor for an [Address](#Address) from a [verification key](#VerificationKey) hash. The resulting address has no delegation rights whatsoever.

pub fn from\_verification\_key(vk: Hash<Blake2b\_224, VerificationKey>) -> Address {

Address {

payment\_credential: VerificationKeyCredential(vk),

stake\_credential: None,

}

}

/// Smart-constructor for an [Address](#Address) from a [script](#Script) hash. The address has no delegation rights whatsoever.

pub fn from\_script(script: Hash<Blake2b\_224, Script>) -> Address {

Address {

payment\_credential: ScriptCredential(script),

stake\_credential: None,

}

}

/// Set (or reset) the delegation part of an [Address](#Address) using a [verification key](#VerificationKey) hash. This is useful when combined with [`from\_verification\_key`](#from\_verification\_key) and/or [`from\_script`](#from\_script).

pub fn with\_delegation\_key(

self: Address,

vk: Hash<Blake2b\_224, VerificationKey>,

) -> Address {

Address {

payment\_credential: self.payment\_credential,

stake\_credential: Some(Inline(VerificationKeyCredential(vk))),

}

}

/// Set (or reset) the delegation part of an [Address](#Address) using a [script](#Script) hash. This is useful when combined with [`from\_verification\_key`](#from\_verification\_key) and/or [`from\_script`](#from\_script).

pub fn with\_delegation\_script(

self: Address,

script: Hash<Blake2b\_224, Script>,

) -> Address {

Address {

payment\_credential: self.payment\_credential,

stake\_credential: Some(Inline(ScriptCredential(script))),

}

}

/// Represent a type of object that can be represented either inline (by hash)

/// or via a reference (i.e. a pointer to an on-chain location).

///

/// This is mainly use for capturing pointers to a stake credential

/// registration certificate in the case of so-called pointer addresses.

pub type Referenced<a> {

Inline(a)

Pointer { slot\_number: Int, transaction\_index: Int, certificate\_index: Int }

}

pub type VerificationKey =

ByteArray

pub type Script =

ByteArray

pub type Signature =

ByteArray

/// Verify an Ed25519 signature using the given verification key.

/// Returns `True` when the signature is valid.

pub fn verify\_signature(

key: VerificationKey,

msg: ByteArray,

sig: Signature,

) -> Bool {

builtin.verify\_ed25519\_signature(key, msg, sig)

}

/// A `StakeCredential` represents the delegation and rewards withdrawal conditions

/// associated with some stake address / account.

///

/// A `StakeCredential` is either provided inline, or, by reference using an

/// on-chain pointer.

///

/// Read more about pointers in [CIP-0019 :: Pointers](https://github.com/cardano-foundation/CIPs/tree/master/CIP-0019/#pointers).

pub type StakeCredential =

Referenced<Credential>

/// A 'PaymentCredential' represents the spending conditions associated with

/// some output. Hence,

///

/// - a `VerificationKeyCredential` captures an output locked by a public/private key pair;

/// - and a `ScriptCredential` captures an output locked by a native or Plutus script.

///

pub type PaymentCredential =

Credential

/// A unique stake pool identifier, as a hash of its owner verification key.

pub type PoolId =

Hash<Blake2b\_224, VerificationKey>

#### #### value.ak

use aiken/dict.{Dict, from\_ascending\_pairs\_with}

use aiken/hash.{Blake2b\_224, Hash}

use aiken/list

use aiken/option

use aiken/transaction/credential.{Script}

/// A type-alias for a `PolicyId`. A `PolicyId` is always 28-byte long

pub type PolicyId =

Hash<Blake2b\_224, Script>

/// Ada, the native currency, isn't associated with any `PolicyId` (it's not

/// possible to mint Ada!).

///

/// By convention, it is an empty `ByteArray`.

pub const ada\_policy\_id = #""

/// A type-alias for 'AssetName`, which are free-form byte-arrays between

/// 0 and 32 bytes.

pub type AssetName =

ByteArray

/// Ada, the native currency, isn't associated with any `AssetName` (it's not

/// possible to mint Ada!).

///

/// By convention, it is an empty `ByteArray`.

pub const ada\_asset\_name = #""

/// A multi-asset output `Value`. Contains tokens indexed by [PolicyId](#PolicyId) and [AssetName](#AssetName).

///

/// This type maintain some invariants by construction; in particular, a `Value` will never contain a

/// zero quantity of a particular token.

pub opaque type Value {

inner: Dict<PolicyId, Dict<AssetName, Int>>,

}

/// Construct an empty `Value` with nothing in it.

pub fn zero() -> Value {

Value { inner: dict.new() }

}

/// Check is a `Value` is zero. That is, it has no assets and holds no Ada/Lovelace.

pub fn is\_zero(self: Value) -> Bool {

self == zero()

}

/// Construct a `Value` from an asset identifier (i.e. `PolicyId` + `AssetName`)

/// and a given quantity.

pub fn from\_asset(

policy\_id: PolicyId,

asset\_name: AssetName,

quantity: Int,

) -> Value {

if quantity == 0 {

dict.new()

|> Value

} else {

let asset =

dict.new()

|> dict.insert(asset\_name, quantity)

dict.new()

|> dict.insert(policy\_id, asset)

|> Value

}

}

/// Construct a `Value` from a lovelace quantity.

///

/// Friendly reminder: 1 Ada = 1.000.000 Lovelace

pub fn from\_lovelace(quantity: Int) -> Value {

from\_asset(ada\_policy\_id, ada\_asset\_name, quantity)

}

/// Get a `Value` excluding Ada.

pub fn without\_lovelace(self: Value) -> Value {

dict.delete(self.inner, ada\_policy\_id)

|> Value

}

test without\_lovelace\_1() {

let v = from\_lovelace(1000000)

without\_lovelace(v) == zero()

}

test without\_lovelace\_2() {

let v = from\_lovelace(1000000)

let v2 = from\_lovelace(50000000)

without\_lovelace(v) == without\_lovelace(v2)

}

test without\_lovelace\_3() {

let v =

from\_asset(#"010203", #"040506", 100)

|> add(ada\_policy\_id, ada\_asset\_name, 100000000)

let v2 = from\_asset(#"010203", #"040506", 100)

without\_lovelace(v) == without\_lovelace(v2) && without\_lovelace(v) == v2

}

/// Negates quantities of all tokens (including Ada) in that `Value`.

///

/// ```

/// v1

/// |> value.negate

/// |> value.merge(v1)

/// |> value.is\_zero

/// // True

/// ```

pub fn negate(self: Value) -> Value {

dict.map(self.inner, fn(\_, a) { dict.map(a, fn(\_, q) { 0 - q }) })

|> Value

}

/// Combine two `Value` together.

pub fn merge(left v0: Value, right v1: Value) -> Value {

Value(

dict.union\_with(

v0.inner,

v1.inner,

fn(\_, a0, a1) {

let result =

dict.union\_with(

a0,

a1,

fn(\_, q0, q1) {

let q = q0 + q1

if q == 0 {

None

} else {

Some(q)

}

},

)

if dict.is\_empty(result) {

None

} else {

Some(result)

}

},

),

)

}

test merge\_1() {

let v1 = from\_lovelace(1)

let v2 = from\_lovelace(-1)

merge(v1, v2) == zero()

}

test merge\_2() {

let v1 = from\_asset(#"00", #"", 1)

let v2 = from\_asset(#"01", #"", 2)

let v3 = from\_asset(#"02", #"", 3)

let v =

from\_lovelace(42)

|> merge(v3)

|> merge(v1)

|> merge(v2)

flatten(v) == [

(#"", #"", 42),

(#"00", #"", 1),

(#"01", #"", 2),

(#"02", #"", 3),

]

}

test merge\_3() {

let v1 = from\_asset(#"00", #"", 1)

let v2 = from\_asset(#"00", #"", -1)

let v3 = from\_asset(#"01", #"", 1)

let v =

zero()

|> merge(v1)

|> merge(v2)

|> merge(v3)

flatten(v) == [(#"01", #"", 1)]

}

test merge\_4() {

let v1 = from\_asset(#"00", #"", 1)

let v2 = from\_asset(#"00", #"", -1)

merge(v1, v2) == zero()

}

test merge\_5() {

let v =

zero()

|> add(#"acab", #"beef", 0)

merge(zero(), v) == zero()

}

/// Add a (positive or negative) quantity of a single token to a value.

/// This is more efficient than [`merge`](#merge) for a single asset.

pub fn add(

self: Value,

policy\_id: PolicyId,

asset\_name: AssetName,

quantity: Int,

) -> Value {

if quantity == 0 {

self

} else {

let helper =

fn(\_, left, \_right) {

let inner\_result =

dict.insert\_with(

left,

asset\_name,

quantity,

fn(\_k, ql, qr) {

let q = ql + qr

if q == 0 {

None

} else {

Some(q)

}

},

)

if dict.is\_empty(inner\_result) {

None

} else {

Some(inner\_result)

}

}

Value(

dict.insert\_with(

self.inner,

policy\_id,

dict.from\_ascending\_pairs([Pair(asset\_name, quantity)]),

helper,

),

)

}

}

test add\_1() {

let v =

zero()

|> add(#"acab", #"beef", 321)

|> add(#"acab", #"beef", -321)

v == zero()

}

test add\_2() {

let v =

from\_lovelace(123)

|> add(#"acab", #"beef", 321)

|> add(#"acab", #"beef", -1 \* 321)

v == from\_lovelace(123)

}

test add\_3() {

let v =

from\_lovelace(1)

|> add(ada\_policy\_id, ada\_asset\_name, 2)

|> add(ada\_policy\_id, ada\_asset\_name, 3)

v == from\_lovelace(6)

}

test add\_4() {

let v =

zero()

|> add(#"acab", #"beef", 0)

v == zero()

}

test add\_5() {

let v =

zero()

|> add(#"acab", #"beef", 0)

|> add(#"acab", #"beef", 0)

v == zero()

}

/// Extract the quantity of a given asset.

pub fn quantity\_of(

self: Value,

policy\_id: PolicyId,

asset\_name: AssetName,

) -> Int {

self.inner

|> dict.get(policy\_id)

|> option.and\_then(dict.get(\_, asset\_name))

|> option.or\_else(0)

}

/// A specialized version of `quantity\_of` for the Ada currency.

pub fn lovelace\_of(self: Value) -> Int {

quantity\_of(self, ada\_policy\_id, ada\_asset\_name)

}

/// Get all tokens associated with a given policy.

pub fn tokens(self: Value, policy\_id: PolicyId) -> Dict<AssetName, Int> {

self.inner

|> dict.get(policy\_id)

|> option.or\_else(dict.new())

}

/// A list of all token policies in that Value with non-zero tokens.

pub fn policies(self: Value) -> List<PolicyId> {

dict.keys(self.inner)

}

/// Flatten a value as list of 3-tuple (PolicyId, AssetName, Quantity).

///

/// Handy to manipulate values as uniform lists.

pub fn flatten(self: Value) -> List<(PolicyId, AssetName, Int)> {

dict.foldr(

self.inner,

[],

fn(policy\_id, asset\_list, value) {

dict.foldr(

asset\_list,

value,

fn(asset\_name, quantity, xs) {

[(policy\_id, asset\_name, quantity), ..xs]

},

)

},

)

}

/// Flatten a value as a list of results, possibly discarding some along the way.

///

/// When the transform function returns `None`, the result is discarded altogether.

pub fn flatten\_with(

self: Value,

with: fn(PolicyId, AssetName, Int) -> Option<result>,

) -> List<result> {

dict.foldr(

self.inner,

[],

fn(policy\_id, asset\_list, value) {

dict.foldr(

asset\_list,

value,

fn(asset\_name, quantity, xs) {

when with(policy\_id, asset\_name, quantity) is {

None -> xs

Some(x) ->

[x, ..xs]

}

},

)

},

)

}

test flatten\_with\_1() {

flatten\_with(zero(), fn(p, a, q) { Some((p, a, q)) }) == []

}

test flatten\_with\_2() {

let v =

zero()

|> add("a", "1", 14)

|> add("b", "", 42)

|> add("a", "2", 42)

flatten\_with(

v,

fn(p, a, q) {

if q == 42 {

Some((p, a))

} else {

None

}

},

) == [("a", "2"), ("b", "")]

}

/// Reduce a value into a single result

///

/// ```

/// value.zero()

/// |> value.add("a", "1", 10)

/// |> value.add("b", "2", 20)

/// |> value.reduce(v, 0, fn(\_, \_, quantity, acc) { acc + quantity })

/// // 30

/// ```

pub fn reduce(

self: Value,

start: acc,

with: fn(PolicyId, AssetName, Int, acc) -> acc,

) -> acc {

dict.foldr(

self.inner,

start,

fn(policy\_id, asset\_list, result) {

dict.foldr(asset\_list, result, with(policy\_id, \_, \_, \_))

},

)

}

test reduce\_1() {

let v =

zero()

|> add("a", "1", 10)

|> add("b", "2", 20)

let result = reduce(v, 0, fn(\_, \_, quantity, acc) { acc + quantity })

result == 30

}

test reduce\_2() {

let v =

zero()

|> add("a", "1", 5)

|> add("a", "2", 15)

|> add("b", "", 10)

let result =

reduce(

v,

[],

fn(policy\_id, asset\_name, \_, acc) { [(policy\_id, asset\_name), ..acc] },

)

result == [("a", "1"), ("a", "2"), ("b", "")]

}

test reduce\_3() {

let v = zero()

let result = reduce(v, 1, fn(\_, \_, quantity, acc) { acc + quantity })

result == 1

}

/// Promote an arbitrary list of assets into a `Value`. This function fails

/// (i.e. halt the program execution) if:

///

/// - there's any duplicate amongst `PolicyId`;

/// - there's any duplicate amongst `AssetName`;

/// - the `AssetName` aren't sorted in ascending lexicographic order; or

/// - any asset quantity is null.

///

/// This function is meant to turn arbitrary user-defined `Data` into safe `Value`,

/// while checking for internal invariants.

pub fn from\_asset\_list(xs: Pairs<PolicyId, Pairs<AssetName, Int>>) -> Value {

xs

|> list.foldr(

dict.new(),

fn(inner, acc) {

expect Pair(p, [\_, ..] as x) = inner

x

|> from\_ascending\_pairs\_with(fn(v) { v != 0 })

|> dict.insert\_with(

acc,

p,

\_,

fn(\_, \_, \_) {

fail @"Duplicate policy in the asset list."

},

)

},

)

|> Value

}

test from\_asset\_list\_1() {

let v = from\_asset\_list([])

v == zero()

}

test from\_asset\_list\_2() fail {

let v = from\_asset\_list([Pair(#"33", [])])

v == zero()

}

test from\_asset\_list\_3() fail {

let v = from\_asset\_list([Pair(#"33", [Pair(#"", 0)])])

v != zero()

}

test from\_asset\_list\_4() {

let v = from\_asset\_list([Pair(#"33", [Pair(#"", 1)])])

flatten(v) == [(#"33", #"", 1)]

}

test from\_asset\_list\_5() {

let v = from\_asset\_list([Pair(#"33", [Pair(#"", 1), Pair(#"33", 1)])])

flatten(v) == [(#"33", #"", 1), (#"33", #"33", 1)]

}

test from\_asset\_list\_6() fail {

let v =

from\_asset\_list(

[

Pair(#"33", [Pair(#"", 1), Pair(#"33", 1)]),

Pair(#"33", [Pair(#"", 1), Pair(#"33", 1)]),

],

)

v != zero()

}

test from\_asset\_list\_7() fail {

let v =

from\_asset\_list(

[

Pair(#"33", [Pair(#"", 1), Pair(#"33", 1)]),

Pair(#"34", [Pair(#"", 1), Pair(#"", 1)]),

],

)

v != zero()

}

test from\_asset\_list\_8() {

let v =

from\_asset\_list(

[

Pair(#"33", [Pair(#"", 1), Pair(#"33", 1)]),

Pair(#"34", [Pair(#"31", 1)]),

Pair(#"35", [Pair(#"", 1)]),

],

)

flatten(v) == [

(#"33", #"", 1),

(#"33", #"33", 1),

(#"34", #"31", 1),

(#"35", #"", 1),

]

}

test from\_asset\_list\_9() {

let v =

from\_asset\_list(

[

Pair(#"35", [Pair(#"", 1)]),

Pair(#"33", [Pair(#"", 1), Pair(#"33", 1)]),

Pair(#"34", [Pair(#"31", 1)]),

],

)

flatten(v) == [

(#"33", #"", 1),

(#"33", #"33", 1),

(#"34", #"31", 1),

(#"35", #"", 1),

]

}

/// Convert the value into a dictionary of dictionaries.

pub fn to\_dict(self: Value) -> Dict<PolicyId, Dict<AssetName, Int>> {

self.inner

}

/// A multi-asset value that can be found when minting transaction. It always holds

/// a null quantity of \_Ada\_. Note that because of historical reasons, this is slightly

/// different from `Value` found in transaction outputs.

///

/// Note that you're never expected to construct a `MintedValue` yourself. If you need to

/// manipulate multi-asset values, use [Value](#Value)

///

/// See also [`from\_minted\_value`](#from\_minted\_value).

pub opaque type MintedValue {

inner: Dict<PolicyId, Dict<AssetName, Int>>,

}

/// Convert minted value into a dictionary of dictionaries.

pub fn minted\_to\_dict(self: MintedValue) -> Dict<PolicyId, Dict<AssetName, Int>> {

self.inner

}

/// Convert a [`MintedValue`](#MintedValue) into a [`Value`](#Value).

pub fn from\_minted\_value(self: MintedValue) -> Value {

self.inner |> dict.delete(ada\_policy\_id) |> Value

}

test from\_minted\_value\_1() {

flatten(from\_minted\_value(from\_internal\_list([]))) == []

}

test from\_minted\_value\_2() {

flatten(from\_minted\_value(from\_internal\_list([("p0", "a0", 1)]))) == [

("p0", "a0", 1),

]

}

test from\_minted\_value\_3() {

let assets =

[("p0", "a0", 1), ("p1", "a0", 1), ("p0", "a0", 1), ("p1", "a1", 1)]

let result =

[("p0", "a0", 2), ("p1", "a0", 1), ("p1", "a1", 1)]

flatten(from\_minted\_value(from\_internal\_list(assets))) == result

}

test from\_minted\_value\_4() {

let assets =

[

("", "", 0),

("p0", "a0", 1),

("p1", "a0", 1),

("p0", "a0", 1),

("p1", "a1", 1),

]

let result =

[("p0", "a0", 2), ("p1", "a0", 1), ("p1", "a1", 1)]

flatten(from\_minted\_value(from\_internal\_list(assets))) == result

}

test from\_minted\_value\_5() {

let assets =

[

("p0", "a0", 1),

("p0", "a1", 1),

("p1", "a0", 1),

("p1", "a1", 1),

("p1", "a2", 1),

("p2", "a0", 1),

("p2", "a1", 1),

("p3", "a0", 1),

("p3", "a1", 1),

("p3", "a2", 1),

("p3", "a3", 1),

("p3", "a4", 1),

("p3", "a5", 1),

("p3", "a6", 1),

("p3", "a7", 1),

]

flatten(from\_minted\_value(from\_internal\_list(assets))) == assets

}

/// Convert a [`Value`](#Value) into a [`MintedValue`](#MintedValue).

pub fn to\_minted\_value(self: Value) -> MintedValue {

self.inner

|> dict.insert(ada\_policy\_id, dict.insert(dict.new(), ada\_asset\_name, 0))

|> MintedValue

}

test to\_minted\_value\_1() {

let minted\_value = to\_minted\_value(zero())

( minted\_value.inner |> dict.to\_pairs |> list.length ) == 1

}

test to\_minted\_value\_2() {

let minted\_value = to\_minted\_value(from\_lovelace(42))

(

minted\_value.inner

|> dict.get(ada\_policy\_id)

|> option.and\_then(dict.get(\_, ada\_asset\_name))

) == Some(0)

}

/// Convert a list of tokens into a `MintedValue`.

///

/// NOTE: Not exposed because we do not want people to construct `MintedValue`. Only

/// get them from the script context.

fn from\_internal\_list(xs: List<(PolicyId, AssetName, Int)>) -> MintedValue {

list.foldr(

xs,

MintedValue(dict.new()),

fn(elem, st) {

let (policy\_id, asset\_name, quantity) = elem

unchecked\_add(st, policy\_id, asset\_name, quantity)

},

)

}

fn unchecked\_add(

self: MintedValue,

policy\_id: PolicyId,

asset\_name: AssetName,

quantity: Int,

) -> MintedValue {

MintedValue(

dict.insert\_with(

self.inner,

policy\_id,

dict.from\_ascending\_pairs([Pair(asset\_name, quantity)]),

fn(\_, left, \_right) {

Some(

dict.insert\_with(

left,

asset\_name,

quantity,

fn(\_k, ql, qr) { Some(ql + qr) },

),

)

},

),

)

}

### 2.3 bytearray

use aiken/builtin

use aiken/math

use aiken/option

/// Compare two bytearrays lexicographically.

///

/// ```aiken

/// bytearray.compare(#"00", #"FF") == Less

/// bytearray.compare(#"42", #"42") == Equal

/// bytearray.compare(#"FF", #"00") == Greater

/// ```

pub fn compare(left: ByteArray, right: ByteArray) -> Ordering {

if builtin.less\_than\_bytearray(left, right) {

Less

} else if builtin.equals\_bytearray(left, right) {

Equal

} else {

Greater

}

}

/// Combine two `ByteArray` together.

///

/// ```aiken

/// bytearray.concat(left: #[1, 2, 3], right: #[4, 5, 6]) == #[1, 2, 3, 4, 5, 6]

/// ```

pub fn concat(left: ByteArray, right: ByteArray) -> ByteArray {

builtin.append\_bytearray(left, right)

}

test concat\_1() {

concat(#"", #"") == #""

}

test concat\_2() {

concat(#"", #"01") == #"01"

}

test concat\_3() {

concat(#"0102", #"") == #"0102"

}

test concat\_4() {

concat(#"0102", #"0304") == #"01020304"

}

/// Returns the suffix of a `ByteArray` after `n` elements.

///

/// ```aiken

/// bytearray.drop(#[1, 2, 3], n: 2) == #[3]

/// ```

pub fn drop(self: ByteArray, n: Int) -> ByteArray {

builtin.slice\_bytearray(n, builtin.length\_of\_bytearray(self) - n, self)

}

test drop\_1() {

let x = #"01020304050607"

drop(x, 2) == #"0304050607"

}

test drop\_2() {

let x = #"01020304050607"

drop(x, 0) == x

}

test drop\_3() {

let x = #"01"

drop(x, 1) == #""

}

test drop\_4() {

let x = #""

drop(x, 2) == #""

}

/// Left-fold over bytes of a [`ByteArray`](https://aiken-lang.github.io/prelude/aiken.html#ByteArray). Note that every byte given to the callback function is comprised between 0 and 255.

///

/// ```aiken

/// bytearray.foldl(#"acab", 0, fn(byte, acc) { acc \* 256 + byte }) == 44203

/// bytearray.foldl(#[1, 2, 3], #"", flip(bytearray.push)) == #[3, 2, 1]

/// ```

pub fn foldl(

self: ByteArray,

zero: result,

with: fn(Int, result) -> result,

) -> result {

do\_foldl(self, zero, builtin.length\_of\_bytearray(self), 0, with)

}

fn do\_foldl(

self: ByteArray,

zero: result,

len: Int,

cursor: Int,

with: fn(Int, result) -> result,

) -> result {

if cursor == len {

zero

} else {

do\_foldl(

self,

with(builtin.index\_bytearray(self, cursor), zero),

len,

cursor + 1,

with,

)

}

}

test foldl\_1() {

foldl(#[], 42, fn(byte, acc) { byte + acc }) == 42

}

test foldl\_2() {

foldl(#"acab", 0, fn(byte, acc) { acc \* 256 + byte }) == 44203

}

test foldl\_3() {

foldl(

#"356cf088720a169dae0ce0bb1df8588944389fa43322f0d6ef4ed8c069bfd405",

0,

fn(byte, acc) { acc \* 256 + byte },

) == 24165060555594911913195642527692216679757672038384202527929620681761931383813

}

test foldl\_4() {

foldl(#[1, 2, 3, 4, 5], #"", flip(push)) == #[5, 4, 3, 2, 1]

}

/// Right-fold over bytes of a [`ByteArray`](https://aiken-lang.github.io/prelude/aiken.html#ByteArray). Note that every byte given to the callback function is comprised between 0 and 255.

///

/// ```aiken

/// bytearray.foldr(#"acab", 0, fn(byte, acc) { acc \* 256 + byte }) == 43948

/// bytearray.foldl(#[1, 2, 3], #"", flip(bytearray.push)) == #[1, 2, 3]

/// ```

pub fn foldr(

self: ByteArray,

zero: result,

with: fn(Int, result) -> result,

) -> result {

do\_foldr(self, zero, builtin.length\_of\_bytearray(self) - 1, with)

}

fn do\_foldr(

self: ByteArray,

zero: result,

cursor: Int,

with: fn(Int, result) -> result,

) -> result {

if cursor < 0 {

zero

} else {

do\_foldr(

self,

with(builtin.index\_bytearray(self, cursor), zero),

cursor - 1,

with,

)

}

}

test foldr\_1() {

foldr(#[], 42, fn(byte, acc) { byte + acc }) == 42

}

test foldr\_2() {

foldr(#"acab", 0, fn(byte, acc) { acc \* 256 + byte }) == 43948

}

test foldr\_3() {

foldr(#[1, 2, 3, 4, 5], #"", flip(push)) == #[1, 2, 3, 4, 5]

}

/// Search the start and end positions of a sub-array in a `ByteArray`.

///

/// ```aiken

/// bytearray.index\_of("Hello, World!", "World") == Some((7, 11))

/// bytearray.index\_of("Hello, World!", "foo") == None

/// bytearray.index\_of("Hello, World!", "!") == Some((12, 12))

/// bytearray.index\_of("Hello, World!", "o") == Some((4, 4))

/// bytearray.index\_of("Hello, World!", "Hello, World!") == Some((0, 12))

/// ```

pub fn index\_of(self: ByteArray, bytes: ByteArray) -> Option<(Int, Int)> {

let offset = length(bytes)

do\_index\_of(self, bytes, 0, offset, length(self))

|> option.map(fn(ix) { (ix, ix + offset - 1) })

}

fn do\_index\_of(

self: ByteArray,

bytes: ByteArray,

cursor: Int,

offset: Int,

size: Int,

) -> Option<Int> {

if cursor + offset > size {

None

} else {

if builtin.slice\_bytearray(cursor, offset, self) == bytes {

Some(cursor)

} else {

do\_index\_of(self, bytes, cursor + 1, offset, size)

}

}

}

test index\_of\_1() {

index\_of("Hello, World!", "World") == Some((7, 11))

}

test index\_of\_2() {

index\_of("Hello, World!", "foo") == None

}

test index\_of\_3() {

index\_of("Hello, World!", "!") == Some((12, 12))

}

test index\_of\_4() {

index\_of("Hello, World!", "o") == Some((4, 4))

}

test index\_of\_5() {

index\_of("Hello, World!", "Hello, World!") == Some((0, 12))

}

/// Returns the number of bytes in a `ByteArray`.

///

/// ```aiken

/// bytearray.length(#[1, 2, 3]) == 3

/// ```

pub fn length(self: ByteArray) -> Int {

builtin.length\_of\_bytearray(self)

}

test length\_1() {

length(#"") == 0

}

test length\_2() {

length(#"010203") == 3

}

/// Returns `True` when the given `ByteArray` is empty.

///

/// ```aiken

/// bytearray.is\_empty(#"") == True

/// bytearray.is\_empty(#"00ff") == False

/// ```

pub fn is\_empty(self: ByteArray) -> Bool {

builtin.length\_of\_bytearray(self) == 0

}

test is\_empty\_1() {

is\_empty(#"") == True

}

test is\_empty\_2() {

is\_empty(#"01") == False

}

/// Convert a `String` into a `ByteArray`.

///

/// ```aiken

/// bytearray.from\_string(@"ABC") == #"414243"

/// ```

pub fn from\_string(str: String) -> ByteArray {

builtin.encode\_utf8(str)

}

test from\_string\_1() {

from\_string(@"") == ""

}

test from\_string\_2() {

from\_string(@"ABC") == #"414243"

}

/// Add a byte element in front of a `ByteArray`. When the given byte is

/// greater than 255, it wraps-around. \*\*PlutusV2 behavior\*\* So 256 is mapped to 0, 257 to 1, and so

/// forth.

/// In PlutusV3 this will error instead of wrapping around.

///

/// ```aiken

/// bytearray.push(#"", 0) == #"00"

/// bytearray.push(#"0203", 1) == #"010203"

/// bytearray.push(#"0203", 257) == #"010203"

/// ```

pub fn push(self: ByteArray, byte: Int) -> ByteArray {

builtin.cons\_bytearray(byte, self)

}

test push\_1() {

push(#[], 0) == #[0]

}

test push\_2() {

push(#[2, 3], 1) == #[1, 2, 3]

}

test push\_3() {

let x = 257

push(#[2, 3], x) == #[1, 2, 3]

}

/// Reduce bytes in a ByteArray from left to right using the accumulator as left operand.

/// Said differently, this is [`foldl`](#foldl) with callback arguments swapped.

///

/// ```aiken

/// bytearray.reduce(#[1,2,3], #[], bytearray.push) == #[3, 2, 1]

/// ```

pub fn reduce(

self: ByteArray,

zero: result,

with: fn(result, Int) -> result,

) -> result {

foldl(self, zero, flip(with))

}

test reduce\_1() {

reduce(#[], #[], push) == #[]

}

test reduce\_2() {

reduce(#[1, 2, 3], #[], push) == #[3, 2, 1]

}

/// Extract a `ByteArray` as a slice of another `ByteArray`.

///

/// Indexes are 0-based and inclusive.

///

/// ```aiken

/// bytearray.slice(#[0, 1, 2, 3, 4, 5, 6], start: 1, end: 3) == #[1, 2, 3]

/// ```

pub fn slice(self: ByteArray, start: Int, end: Int) -> ByteArray {

builtin.slice\_bytearray(start, end - start + 1, self)

}

test slice\_1() {

slice(#"", 1, 2) == #""

}

test slice\_2() {

slice(#"010203", 1, 2) == #"0203"

}

test slice\_3() {

slice(#"010203", 0, 42) == #"010203"

}

test slice\_4() {

slice(#[0, 1, 2, 3, 4], 0, 3) == #[0, 1, 2, 3]

}

test slice\_5() {

slice(#[0, 1, 2, 3, 4], 1, 2) == #[1, 2]

}

/// Returns the n-length prefix of a `ByteArray`.

///

/// ```aiken

/// bytearray.take(#[1, 2, 3], n: 2) == #[1, 2]

/// ```

pub fn take(self: ByteArray, n: Int) -> ByteArray {

builtin.slice\_bytearray(0, n, self)

}

test take\_1() {

let x = #"01020304050607"

take(x, 2) == #"0102"

}

test take\_2() {

let x = #"01020304050607"

take(x, 0) == #""

}

test take\_3() {

let x = #"01"

take(x, 1) == x

}

test take\_4() {

let x = #"010203"

take(x, 0) == #""

}

/// Convert a `ByteArray` into a `String`.

///

/// <br/>⚠️<pre>WARNING</pre> | This functions fails if the underlying `ByteArray` isn't UTF-8-encoded. <br/>In particular, you cannot convert arbitrary hash digests using this function. <br/>For converting arbitrary `ByteArray`s, use [bytearray.to\_hex](#to\_hex).

/// --- | ---

///

///

/// ```aiken

/// bytearray.to\_string(#"414243") == "ABC"

///

/// bytearray.to\_string(some\_hash) -> fail

/// ```

pub fn to\_string(self: ByteArray) -> String {

builtin.decode\_utf8(self)

}

test to\_string\_1() {

to\_string("") == @""

}

test to\_string\_2() {

to\_string("ABC") == @"ABC"

}

/// Encode a `ByteArray` as a hexidecimal `String`.

///

/// ```aiken

/// use aiken/bytearray

///

/// bytearray.to\_hex("Hello world!") == @"48656c6c6f20776f726c6421"

/// ```

pub fn to\_hex(self: ByteArray) -> String {

self

|> encode\_base16(builtin.length\_of\_bytearray(self) - 1, "")

|> builtin.decode\_utf8

}

// Construct an hex string in reverse order, from the back. The 'builder' is an

// accumulator. It works fast because `index\_bytearray` follows a constant-time cost

// model

fn encode\_base16(bytes: ByteArray, ix: Int, builder: ByteArray) -> ByteArray {

if ix < 0 {

builder

} else {

let byte = builtin.index\_bytearray(bytes, ix)

let msb = byte / 16

let lsb = byte % 16

encode\_base16(

bytes,

ix - 1,

builtin.cons\_bytearray(

msb + if msb < 10 {

48

} else {

87

},

builtin.cons\_bytearray(

lsb + if lsb < 10 {

48

} else {

87

},

builder,

),

),

)

}

}

test to\_hex\_1() {

to\_hex("Hello world!") == @"48656c6c6f20776f726c6421"

}

test to\_hex\_2() {

to\_hex("The quick brown fox jumps over the lazy dog") == @"54686520717569636b2062726f776e20666f78206a756d7073206f76657220746865206c617a7920646f67"

}

/// Checks whether a bit (Most-Significant-Bit first) is set in the given 'ByteArray'.

///

/// For example, consider the following bytearray: `#"8b765f"`. It can also be written as the

/// following bits sequence:

///

/// `8` | `b` | `7` | `6` | `5` | `f`

/// --- | --- | --- | --- | --- | ---

/// `1000` | `1011` | `0111` | `0110` | `0101` | `1111`

///

/// And thus, we have:

///

/// ```aiken

/// test\_bit(#"8b765f", 0) == True

/// test\_bit(#"8b765f", 1) == False

/// test\_bit(#"8b765f", 2) == False

/// test\_bit(#"8b765f", 3) == False

/// test\_bit(#"8b765f", 7) == True

/// test\_bit(#"8b765f", 8) == False

/// test\_bit(#"8b765f", 20) == True

/// test\_bit(#"8b765f", 21) == True

/// test\_bit(#"8b765f", 22) == True

/// test\_bit(#"8b765f", 23) == True

/// ```

pub fn test\_bit(self: ByteArray, ix: Int) -> Bool {

builtin.less\_than\_equals\_bytearray(

#[128],

builtin.cons\_bytearray(

builtin.index\_bytearray(self, ix / 8) \* math.pow2(ix % 8),

"",

),

)

}

test test\_bit\_0() {

test\_bit(#"8b765f", 0)

}

test test\_bit\_1() {

!test\_bit(#"8b765f", 1)

}

test test\_bit\_2() {

!test\_bit(#"8b765f", 2)

}

test test\_bit\_3() {

!test\_bit(#"8b765f", 3)

}

test test\_bit\_7() {

test\_bit(#"8b765f", 7)

}

test test\_bit\_8() {

!test\_bit(#"8b765f", 8)

}

test test\_bit\_20\_21\_22\_23() {

and {

test\_bit(#"8b765f", 20),

test\_bit(#"8b765f", 21),

test\_bit(#"8b765f", 22),

test\_bit(#"8b765f", 23),

}

}

### 

### 2.4. Cbor

2.4. cbor.ak

use aiken/builtin.{

append\_bytearray, choose\_data, cons\_bytearray, decode\_utf8, index\_bytearray,

length\_of\_bytearray, quotient\_integer, remainder\_integer, serialise\_data,

un\_b\_data, un\_constr\_data, un\_i\_data, un\_list\_data, un\_map\_data,

}

use aiken/list

/// Serialise any value to binary, encoding using [CBOR](https://www.rfc-editor.org/rfc/rfc8949).

///

/// This is particularly useful in combination with hashing functions, as a way

/// to obtain a byte representation that matches the serialised representation

/// used by the ledger in the context of on-chain code.

///

/// Note that the output matches the output of [`diagnostic`](#diagnostic),

/// though with a different encoding. [`diagnostic`](#diagnostic) is merely a

/// textual representation of the CBOR encoding that is human friendly and

/// useful for debugging.

///

/// ```aiken

/// serialise(42) == #"182a"

/// serialise(#"a1b2") == #"42a1b2"

/// serialise([]) == #"80"

/// serialise((1, 2)) == #"9f0102ff"

/// serialise((1, #"ff", 3)) == #"9f0141ff03ff"

/// serialise([(1, #"ff")]) == #"a10141ff"

/// serialise(Some(42)) == #"d8799f182aff"

/// serialise(None) == #"d87a80"

/// ```

pub fn serialise(self: Data) -> ByteArray {

serialise\_data(self)

}

test serialise\_1() {

serialise(42) == #"182a"

}

test serialise\_2() {

serialise(#"a1b2") == #"42a1b2"

}

test serialise\_3() {

serialise([]) == #"80"

}

test serialise\_4() {

serialise((1, 2)) == #"9f0102ff"

}

test serialise\_5() {

serialise((1, #"ff", 3)) == #"9f0141ff03ff"

}

test serialise\_6() {

serialise([(1, #"ff")]) == #"9f9f0141ffffff"

}

test serialise\_7() {

serialise(Some(42)) == #"d8799f182aff"

}

test serialise\_8() {

serialise(None) == #"d87a80"

}

test serialise\_9() {

serialise([Pair(1, #"ff")]) == #"a10141ff"

}

/// Obtain a String representation of \_anything\_. This is particularly (and only) useful for tracing

/// and debugging. This function is expensive and should not be used in any production code as it

/// will very likely explodes the validator's budget.

///

/// The output is a [CBOR diagnostic](https://www.rfc-editor.org/rfc/rfc8949#name-diagnostic-notation)

/// of the underlying on-chain binary representation of the data. It's not as

/// easy to read as plain Aiken code, but it is handy for troubleshooting values

/// \_at runtime\_. Incidentally, getting familiar with reading CBOR diagnostic is

/// a good idea in the Cardano world.

///

/// ```aiken

/// diagnostic(42) == "42"

/// diagnostic(#"a1b2") == "h'A1B2'"

/// diagnostic([1, 2, 3]) == "[\_ 1, 2, 3]"

/// diagnostic([]) == "[]"

/// diagnostic((1, 2)) == "[\_ 1, 2]"

/// diagnostic((1, #"ff", 3)) == "[\_ 1, h'FF', 3]"

/// diagnostic([(1, #"ff")]) == "{\_ 1: h'FF' }"

/// diagnostic(Some(42)) == "121([\_ 42])"

/// diagnostic(None) == "122([])"

/// ```

pub fn diagnostic(self: Data) -> String {

do\_diagnostic(self, #"")

|> decode\_utf8

}

/// UTF-8 lookup table. Comes in handy to decipher the code below.

///

/// | Symbol | Decimal | Hex |

/// | --- | --- | --- |

/// | | 32 | 0x20 |

/// | ' | 39 | 0x27 |

/// | ( | 40 | 0x28 |

/// | ) | 41 | 0x29 |

/// | , | 44 | 0x2c |

/// | 0 | 48 | 0x30 |

/// | : | 58 | 0x3a |

/// | A | 65 | 0x41 |

/// | [ | 91 | 0x5b |

/// | ] | 93 | 0x5d |

/// | \_ | 95 | 0x5f |

/// | h | 104 | 0x68 |

/// | { | 123 | 0x7b |

/// | } | 125 | 0x7d |

fn do\_diagnostic(self: Data, builder: ByteArray) -> ByteArray {

choose\_data(

self,

{

// -------- Constr

let Pair(constr, fields) = un\_constr\_data(self)

// NOTE: This is fundamentally the same logic for serializing list. However, the compiler

// doesn't support mutual recursion just yet, so we can't extract that logic in a separate

// function.

//

// See [aiken-lang/aiken#389](https://github.com/aiken-lang/aiken/pull/389)

let builder =

when fields is {

[] -> append\_bytearray(#"5b5d29", builder)

\_ -> {

let (\_, bytes) =

list.foldr(

fields,

(#"5d", append\_bytearray(#"29", builder)),

fn(e: Data, st: (ByteArray, ByteArray)) {

(#"2c20", do\_diagnostic(e, append\_bytearray(st.1st, st.2nd)))

},

)

append\_bytearray(#"5b5f20", bytes)

}

}

let constr\_tag =

if constr < 7 {

121 + constr

} else if constr < 128 {

1280 + constr - 7

} else {

fail @"What are you doing? No I mean, seriously."

}

builder

|> append\_bytearray(#"28", \_)

|> from\_int(constr\_tag, \_)

},

{

// -------- Map

let elems = un\_map\_data(self)

when elems is {

[] -> append\_bytearray(#"7b7d", builder)

\_ -> {

let (\_, bytes) =

list.foldr(

elems,

(#"207d", builder),

fn(e: Pair<Data, Data>, st: (ByteArray, ByteArray)) {

let value =

do\_diagnostic(e.2nd, append\_bytearray(st.1st, st.2nd))

(

#"2c20",

do\_diagnostic(e.1st, append\_bytearray(#"3a20", value)),

)

},

)

append\_bytearray(#"7b5f20", bytes)

}

}

},

{

// -------- List

let elems = un\_list\_data(self)

when elems is {

[] -> append\_bytearray(#"5b5d", builder)

\_ -> {

let (\_, bytes) =

list.foldr(

elems,

(#"5d", builder),

fn(e: Data, st: (ByteArray, ByteArray)) {

(#"2c20", do\_diagnostic(e, append\_bytearray(st.1st, st.2nd)))

},

)

append\_bytearray(#"5b5f20", bytes)

}

}

},

// -------- Integer

self

|> un\_i\_data

|> from\_int(builder),

{

// -------- ByteArray

let bytes = un\_b\_data(self)

bytes

|> encode\_base16(

length\_of\_bytearray(bytes) - 1,

append\_bytearray(#"27", builder),

)

|> append\_bytearray(#"6827", \_)

},

)

}

fn encode\_base16(bytes: ByteArray, ix: Int, builder: ByteArray) -> ByteArray {

if ix < 0 {

builder

} else {

let byte = index\_bytearray(bytes, ix)

let msb = byte / 16

let lsb = byte % 16

let builder =

cons\_bytearray(

msb + if msb < 10 {

48

} else {

55

},

cons\_bytearray(

lsb + if lsb < 10 {

48

} else {

55

},

builder,

),

)

encode\_base16(bytes, ix - 1, builder)

}

}

fn from\_int(i: Int, digits: ByteArray) -> ByteArray {

if i == 0 {

append\_bytearray(#"30", digits)

} else if i < 0 {

append\_bytearray(#"2d", from\_int(-i, digits))

} else {

do\_from\_int(

quotient\_integer(i, 10),

cons\_bytearray(remainder\_integer(i, 10) + 48, digits),

)

}

}

fn do\_from\_int(i: Int, digits: ByteArray) -> ByteArray {

if i <= 0 {

digits

} else {

do\_from\_int(

quotient\_integer(i, 10),

cons\_bytearray(remainder\_integer(i, 10) + 48, digits),

)

}

}

test diagnostic\_1() {

diagnostic(42) == @"42"

}

test diagnostic\_2() {

diagnostic(#"a1b2") == @"h'A1B2'"

}

test diagnostic\_3() {

diagnostic([1, 2, 3]) == @"[\_ 1, 2, 3]"

}

test diagnostic\_4() {

diagnostic([]) == @"[]"

}

test diagnostic\_5() {

diagnostic((1, 2)) == @"[\_ 1, 2]"

}

test diagnostic\_6() {

diagnostic((1, #"ff", 3)) == @"[\_ 1, h'FF', 3]"

}

test diagnostic\_7() {

diagnostic([(1, #"ff")]) == @"[\_ [\_ 1, h'FF']]"

}

test diagnostic\_7\_alt() {

diagnostic([Pair(1, #"ff")]) == @"{\_ 1: h'FF' }"

}

test diagnostic\_8() {

diagnostic(Some(42)) == @"121([\_ 42])"

}

test diagnostic\_9() {

diagnostic(None) == @"122([])"

}

test diagnostic\_10() {

let xs: List<(Int, Int)> =

[]

diagnostic(xs) == @"[]"

}

test diagnostic\_10\_alt() {

let xs: Pairs<Int, Int> =

[]

diagnostic(xs) == @"{}"

}

type Foo {

foo: Bar,

}

type Bar {

A

B(Int)

}

test diagnostic\_11() {

diagnostic(Foo { foo: A }) == @"121([\_ 121([])])"

}

test diagnostic\_12() {

diagnostic(Foo { foo: B(42) }) == @"121([\_ 122([\_ 42])])"

}

type Baz {

a0: Int,

b0: ByteArray,

}

test diagnostic\_13() {

diagnostic(Baz { a0: 14, b0: #"ff" }) == @"121([\_ 14, h'FF'])"

}

test diagnostic\_14() {

diagnostic([0]) == @"[\_ 0]"

}

test diagnostic\_15() {

diagnostic(-42) == @"-42"

}

test diagnostic\_16() {

diagnostic([-1, 0, 1]) == @"[\_ -1, 0, 1]"

}

### 2.5. Dict

//// A module for working with bytearray dictionaries.

////

//// ### Important

////

//// Dictionaries are \*\*ordered sets\*\* of key-value pairs, which thus

//// preserve some invariants. Specifically, each key is only present once in

//// the dictionary and all keys are stored in ascending lexicographic order.

////

//// These invariants allow for more optimized functions to operate on `Dict`,

//// but as a trade-offs, prevent `Dict` from being serializable. To recover a `Dict`

//// from an unknown `Data`, you must first recover an `Pairs<k, v>` and use

//// `dict.from\_ascending\_list`.

use aiken/builtin

/// An opaque `Dict`. The type is opaque because the module maintains some

/// invariant, namely: there's only one occurrence of a given key in the dictionary.

///

/// Note that the `key` parameter is a phantom-type, and only present as a

/// means of documentation. Keys can be any type, yet will need to comparable

/// to use functions like `insert`.

///

/// See for example:

///

/// ```aiken

/// pub type Value =

/// Dict<PolicyId, Dict<AssetName, Int>>

/// ```

pub opaque type Dict<key, value> {

inner: Pairs<ByteArray, value>,

}

/// Create a new empty Dict

/// ```aiken

/// dict.to\_pairs(dict.new()) == []

/// ```

pub fn new() -> Dict<key, value> {

Dict { inner: [] }

}

const foo = #"666f6f"

const bar = #"626172"

const baz = #"62617a"

fn fixture\_1() {

new()

|> insert(foo, 42)

|> insert(bar, 14)

}

/// Remove a key-value pair from the dictionary. If the key is not found, no changes are made.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: 100)

/// |> dict.insert(key: "b", value: 200)

/// |> dict.delete(key: "a")

/// |> dict.to\_pairs()

///

/// result == [Pair("b", 200)]

/// ```

pub fn delete(self: Dict<key, value>, key: ByteArray) -> Dict<key, value> {

Dict { inner: do\_delete(self.inner, key) }

}

fn do\_delete(

self: Pairs<ByteArray, value>,

key k: ByteArray,

) -> Pairs<ByteArray, value> {

when self is {

[] ->

[]

[Pair(k2, v2), ..rest] ->

if builtin.less\_than\_equals\_bytearray(k, k2) {

if k == k2 {

rest

} else {

self

}

} else {

[Pair(k2, v2), ..do\_delete(rest, k)]

}

}

}

test delete\_1() {

delete(new(), foo) == new()

}

test delete\_2() {

let m =

new()

|> insert(foo, 14)

delete(m, foo) == new()

}

test delete\_3() {

let m =

new()

|> insert(foo, 14)

delete(m, bar) == m

}

test delete\_4() {

let m =

new()

|> insert(foo, 14)

|> insert(bar, 14)

!has\_key(delete(m, foo), foo)

}

test delete\_5() {

let m =

new()

|> insert(foo, 14)

|> insert(bar, 14)

has\_key(delete(m, bar), foo)

}

test delete\_6() {

let m =

new()

|> insert("aaa", 1)

|> insert("bbb", 2)

|> insert("ccc", 3)

|> insert("ddd", 4)

|> insert("eee", 5)

|> insert("fff", 6)

|> insert("ggg", 7)

|> insert("hhh", 8)

|> insert("iii", 9)

|> insert("jjj", 10)

delete(m, "bcd") == m

}

/// Keep only the key-value pairs that pass the given predicate.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: 100)

/// |> dict.insert(key: "b", value: 200)

/// |> dict.insert(key: "c", value: 300)

/// |> dict.filter(fn(k, \_v) { k != "a" })

/// |> dict.to\_pairs()

///

/// result == [Pair("b", 200), Pair("c", 300)]

/// ```

pub fn filter(

self: Dict<key, value>,

with: fn(ByteArray, value) -> Bool,

) -> Dict<key, value> {

Dict { inner: do\_filter(self.inner, with) }

}

fn do\_filter(

self: Pairs<ByteArray, value>,

with: fn(ByteArray, value) -> Bool,

) -> Pairs<ByteArray, value> {

when self is {

[] ->

[]

[Pair(k, v), ..rest] ->

if with(k, v) {

[Pair(k, v), ..do\_filter(rest, with)]

} else {

do\_filter(rest, with)

}

}

}

test filter\_1() {

filter(new(), fn(\_, \_) { True }) == new()

}

test filter\_2() {

let expected =

new()

|> insert(foo, 42)

filter(fixture\_1(), fn(\_, v) { v > 14 }) == expected

}

test filter\_3() {

let expected =

new()

|> insert(bar, 14)

filter(fixture\_1(), fn(k, \_) { k == bar }) == expected

}

/// Finds a value in the dictionary, and returns the first key found to have that value.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: 42)

/// |> dict.insert(key: "b", value: 14)

/// |> dict.insert(key: "c", value: 42)

/// |> dict.find(42)

///

/// result == Some("a")

/// ```

pub fn find(self: Dict<key, value>, value v: value) -> Option<ByteArray> {

do\_find(self.inner, v)

}

fn do\_find(self: Pairs<ByteArray, value>, value v: value) -> Option<ByteArray> {

when self is {

[] -> None

[Pair(k2, v2), ..rest] ->

if v == v2 {

Some(k2)

} else {

do\_find(rest, v)

}

}

}

test find\_1() {

find(new(), foo) == None

}

test find\_2() {

find(

new()

|> insert(foo, 14),

14,

) == Some(foo)

}

test find\_3() {

find(

new()

|> insert(foo, 14),

42,

) == None

}

test find\_4() {

find(

new()

|> insert(foo, 14)

|> insert(bar, 42)

|> insert(baz, 14),

14,

) == Some(baz)

}

/// Fold over the key-value pairs in a dictionary. The fold direction follows keys

/// in ascending order and is done from right-to-left.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: 100)

/// |> dict.insert(key: "b", value: 200)

/// |> dict.insert(key: "c", value: 300)

/// |> dict.foldr(0, fn(\_k, v, r) { v + r })

///

/// result == 600

/// ```

pub fn foldr(

self: Dict<key, value>,

zero: result,

with: fn(ByteArray, value, result) -> result,

) -> result {

do\_foldr(self.inner, zero, with)

}

fn do\_foldr(

self: Pairs<ByteArray, value>,

zero: result,

with: fn(ByteArray, value, result) -> result,

) -> result {

when self is {

[] -> zero

[Pair(k, v), ..rest] -> with(k, v, do\_foldr(rest, zero, with))

}

}

test foldr\_1() {

foldr(new(), 14, fn(\_, \_, \_) { 42 }) == 14

}

test foldr\_2() {

foldr(fixture\_1(), zero: 0, with: fn(\_, v, total) { v + total }) == 56

}

/// Fold over the key-value pairs in a dictionary. The fold direction follows keys

/// in ascending order and is done from left-to-right.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: 100)

/// |> dict.insert(key: "b", value: 200)

/// |> dict.insert(key: "c", value: 300)

/// |> dict.foldl(0, fn(\_k, v, r) { v + r })

///

/// result == 600

/// ```

pub fn foldl(

self: Dict<key, value>,

zero: result,

with: fn(ByteArray, value, result) -> result,

) -> result {

do\_foldl(self.inner, zero, with)

}

fn do\_foldl(

self: Pairs<ByteArray, value>,

zero: result,

with: fn(ByteArray, value, result) -> result,

) -> result {

when self is {

[] -> zero

[Pair(k, v), ..rest] -> do\_foldl(rest, with(k, v, zero), with)

}

}

test fold\_1() {

foldl(new(), 14, fn(\_, \_, \_) { 42 }) == 14

}

test fold\_2() {

foldl(fixture\_1(), zero: 0, with: fn(\_, v, total) { v + total }) == 56

}

/// Construct a dictionary from a list of key-value pairs. Note that when a key is present

/// multiple times, the first occurrence prevails.

///

/// ```aiken

/// let pairs = [Pair("a", 100), Pair("c", 300), Pair("b", 200)]

///

/// let result =

/// dict.from\_pairs(pairs)

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 100), Pair("b", 200), Pair("c", 300)]

/// ```

pub fn from\_pairs(self: Pairs<ByteArray, value>) -> Dict<key, value> {

Dict { inner: do\_from\_pairs(self) }

}

fn do\_from\_pairs(xs: Pairs<ByteArray, value>) -> Pairs<ByteArray, value> {

when xs is {

[] ->

[]

[Pair(k, v), ..rest] -> do\_insert(do\_from\_pairs(rest), k, v)

}

}

test from\_list\_1() {

from\_pairs([]) == new()

}

test from\_list\_2() {

from\_pairs([Pair(foo, 42), Pair(bar, 14)]) == from\_pairs(

[Pair(bar, 14), Pair(foo, 42)],

)

}

test from\_list\_3() {

from\_pairs([Pair(foo, 42), Pair(bar, 14)]) == fixture\_1()

}

test from\_list\_4() {

from\_pairs([Pair(foo, 42), Pair(bar, 14), Pair(foo, 1337)]) == fixture\_1()

}

test bench\_from\_pairs() {

let dict =

from\_pairs(

[

Pair("bbba", 8),

Pair("bbab", 12),

Pair("aabb", 13),

Pair("aaab", 9),

Pair("bbbb", 16),

Pair("aaaa", 1),

Pair("aaba", 5),

Pair("abab", 10),

Pair("baba", 7),

Pair("baab", 11),

Pair("abaa", 2),

Pair("baaa", 3),

Pair("bbaa", 4),

Pair("babb", 15),

Pair("abbb", 14),

Pair("abba", 6),

],

)

size(dict) == 16

}

/// Like ['from\_list'](from\_list), but from an already sorted list by ascending

/// keys. This function fails (i.e. halt the program execution) if the list isn't

/// sorted.

///

/// ```aiken

/// let pairs = [Pair("a", 100), Pair("b", 200), Pair("c", 300)]

///

/// let result =

/// dict.from\_ascending\_pairs(pairs)

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 100), Pair("b", 200), Pair("c", 300)]

/// ```

///

/// This is meant to be used to turn a list constructed off-chain into a `Dict`

/// which has taken care of maintaining interval invariants. This function still

/// performs a sanity check on all keys to avoid silly mistakes. It is, however,

/// considerably faster than ['from\_list'](from\_list)

pub fn from\_ascending\_pairs(xs: Pairs<ByteArray, value>) -> Dict<key, value> {

let Void = check\_ascending\_list(xs)

Dict { inner: xs }

}

fn check\_ascending\_list(xs: Pairs<ByteArray, value>) {

when xs is {

[] -> Void

[\_] -> Void

[Pair(x0, \_), Pair(x1, \_) as e, ..rest] ->

if builtin.less\_than\_bytearray(x0, x1) {

check\_ascending\_list([e, ..rest])

} else {

fail @"keys in associative list aren't in ascending order"

}

}

}

/// Like [`from\_ascending\_pairs`](#from\_ascending\_list) but fails if \*\*any\*\*

/// value doesn't satisfy the predicate.

///

/// ```aiken

/// let pairs = [Pair("a", 100), Pair("b", 200), Pair("c", 300)]

///

/// dict.from\_ascending\_pairs\_with(pairs, fn(x) { x <= 250 }) // fail

/// ```

pub fn from\_ascending\_pairs\_with(

xs: Pairs<ByteArray, value>,

predicate: fn(value) -> Bool,

) -> Dict<key, value> {

let Void = check\_ascending\_pairs\_with(xs, predicate)

Dict { inner: xs }

}

fn check\_ascending\_pairs\_with(

xs: Pairs<ByteArray, value>,

predicate: fn(value) -> Bool,

) {

when xs is {

[] -> Void

[Pair(\_, v)] ->

if predicate(v) {

Void

} else {

fail @"value doesn't satisfy predicate"

}

[Pair(x0, v0), Pair(x1, \_) as e, ..rest] ->

if builtin.less\_than\_bytearray(x0, x1) {

if predicate(v0) {

check\_ascending\_pairs\_with([e, ..rest], predicate)

} else {

fail @"value doesn't satisfy predicate"

}

} else {

fail @"keys in pairs aren't in ascending order"

}

}

}

test bench\_from\_ascending\_pairs() {

let dict =

from\_ascending\_pairs(

[

Pair("aaaa", 1),

Pair("aaab", 9),

Pair("aaba", 5),

Pair("aabb", 13),

Pair("abaa", 2),

Pair("abab", 10),

Pair("abba", 6),

Pair("abbb", 14),

Pair("baaa", 3),

Pair("baab", 11),

Pair("baba", 7),

Pair("babb", 15),

Pair("bbaa", 4),

Pair("bbab", 12),

Pair("bbba", 8),

Pair("bbbb", 16),

],

)

size(dict) == 16

}

/// Get a value in the dict by its key.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: "Aiken")

/// |> dict.get(key: "a")

///

/// result == Some("Aiken")

/// ```

pub fn get(self: Dict<key, value>, key: ByteArray) -> Option<value> {

do\_get(self.inner, key)

}

fn do\_get(self: Pairs<ByteArray, value>, key k: ByteArray) -> Option<value> {

when self is {

[] -> None

[Pair(k2, v), ..rest] ->

if builtin.less\_than\_equals\_bytearray(k, k2) {

if k == k2 {

Some(v)

} else {

None

}

} else {

do\_get(rest, k)

}

}

}

test get\_1() {

get(new(), foo) == None

}

test get\_2() {

let m =

new()

|> insert(foo, "Aiken")

|> insert(bar, "awesome")

get(m, key: foo) == Some("Aiken")

}

test get\_3() {

let m =

new()

|> insert(foo, "Aiken")

|> insert(bar, "awesome")

get(m, key: baz) == None

}

test get\_4() {

let m =

new()

|> insert("aaa", "1")

|> insert("bbb", "2")

|> insert("ccc", "3")

|> insert("ddd", "4")

|> insert("eee", "5")

|> insert("fff", "6")

|> insert("ggg", "7")

|> insert("hhh", "8")

|> insert("iii", "9")

|> insert("jjj", "10")

get(m, "bcd") == None

}

test get\_5() {

let m =

new()

|> insert("aaa", "1")

|> insert("bbb", "2")

|> insert("ccc", "3")

|> insert("ddd", "4")

|> insert("eee", "5")

|> insert("fff", "6")

|> insert("ggg", "7")

|> insert("hhh", "8")

|> insert("iii", "9")

|> insert("jjj", "10")

get(m, "kkk") == None

}

/// Check if a key exists in the dictionary.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: "Aiken")

/// |> dict.has\_key("a")

///

/// result == True

/// ```

pub fn has\_key(self: Dict<key, value>, key k: ByteArray) -> Bool {

do\_has\_key(self.inner, k)

}

fn do\_has\_key(self: Pairs<ByteArray, value>, key k: ByteArray) -> Bool {

when self is {

[] -> False

[Pair(k2, \_), ..rest] ->

if builtin.less\_than\_equals\_bytearray(k, k2) {

k == k2

} else {

do\_has\_key(rest, k)

}

}

}

test has\_key\_1() {

!has\_key(new(), foo)

}

test has\_key\_2() {

has\_key(

new()

|> insert(foo, 14),

foo,

)

}

test has\_key\_3() {

!has\_key(

new()

|> insert(foo, 14),

bar,

)

}

test has\_key\_4() {

has\_key(

new()

|> insert(foo, 14)

|> insert(bar, 42),

bar,

)

}

/// Insert a value in the dictionary at a given key. If the key already exists, its value is \*\*overridden\*\*. If you need ways to combine keys together, use (`insert\_with`)[#insert\_with].

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert(key: "a", value: 1)

/// |> dict.insert(key: "b", value: 2)

/// |> dict.insert(key: "a", value: 3)

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 3), Pair("b", 2)]

/// ```

pub fn insert(

self: Dict<key, value>,

key k: ByteArray,

value v: value,

) -> Dict<key, value> {

Dict { inner: do\_insert(self.inner, k, v) }

}

fn do\_insert(

self: Pairs<ByteArray, value>,

key k: ByteArray,

value v: value,

) -> Pairs<ByteArray, value> {

when self is {

[] ->

[Pair(k, v)]

[Pair(k2, v2), ..rest] ->

if builtin.less\_than\_bytearray(k, k2) {

[Pair(k, v), ..self]

} else {

if k == k2 {

[Pair(k, v), ..rest]

} else {

[Pair(k2, v2), ..do\_insert(rest, k, v)]

}

}

}

}

test insert\_1() {

let m1 =

new()

|> insert(foo, 42)

let m2 =

new()

|> insert(foo, 14)

insert(m1, foo, 14) == m2

}

test insert\_2() {

let m1 =

new()

|> insert(foo, 42)

let m2 =

new()

|> insert(bar, 14)

insert(m1, bar, 14) == insert(m2, foo, 42)

}

/// Insert a value in the dictionary at a given key. When the key already exist, the provided

/// merge function is called. The value existing in the dictionary is passed as the second argument

/// to the merge function, and the new value is passed as the third argument.

///

/// ```aiken

/// let sum =

/// fn (\_k, a, b) { Some(a + b) }

///

/// let result =

/// dict.new()

/// |> dict.insert\_with(key: "a", value: 1, with: sum)

/// |> dict.insert\_with(key: "b", value: 2, with: sum)

/// |> dict.insert\_with(key: "a", value: 3, with: sum)

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 4), Pair("b", 2)]

/// ```

pub fn insert\_with(

self: Dict<key, value>,

key k: ByteArray,

value v: value,

with: fn(ByteArray, value, value) -> Option<value>,

) -> Dict<key, value> {

Dict {

inner: do\_insert\_with(self.inner, k, v, fn(k, v1, v2) { with(k, v2, v1) }),

}

}

test insert\_with\_1() {

let sum =

fn(\_k, a, b) { Some(a + b) }

let result =

new()

|> insert\_with(key: "foo", value: 1, with: sum)

|> insert\_with(key: "bar", value: 2, with: sum)

|> to\_pairs()

result == [Pair("bar", 2), Pair("foo", 1)]

}

test insert\_with\_2() {

let sum =

fn(\_k, a, b) { Some(a + b) }

let result =

new()

|> insert\_with(key: "foo", value: 1, with: sum)

|> insert\_with(key: "bar", value: 2, with: sum)

|> insert\_with(key: "foo", value: 3, with: sum)

|> to\_pairs()

result == [Pair("bar", 2), Pair("foo", 4)]

}

test insert\_with\_3() {

let with =

fn(k, a, \_b) {

if k == "foo" {

Some(a)

} else {

None

}

}

let result =

new()

|> insert\_with(key: "foo", value: 1, with: with)

|> insert\_with(key: "bar", value: 2, with: with)

|> insert\_with(key: "foo", value: 3, with: with)

|> insert\_with(key: "bar", value: 4, with: with)

|> to\_pairs()

result == [Pair("foo", 1)]

}

/// Efficiently checks whether a dictionary is empty.

/// ```aiken

/// dict.is\_empty(dict.new()) == True

/// ```

pub fn is\_empty(self: Dict<key, value>) -> Bool {

when self.inner is {

[] -> True

\_ -> False

}

}

test is\_empty\_1() {

is\_empty(new())

}

/// Extract all the keys present in a given `Dict`.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert("a", 14)

/// |> dict.insert("b", 42)

/// |> dict.insert("a", 1337)

/// |> dict.keys()

///

/// result == ["a", "b"]

/// ```

pub fn keys(self: Dict<key, value>) -> List<ByteArray> {

do\_keys(self.inner)

}

fn do\_keys(self: Pairs<ByteArray, value>) -> List<ByteArray> {

when self is {

[] ->

[]

[Pair(k, \_), ..rest] ->

[k, ..do\_keys(rest)]

}

}

test keys\_1() {

keys(new()) == []

}

test keys\_2() {

keys(

new()

|> insert(foo, 0)

|> insert(bar, 0),

) == [bar, foo]

}

/// Apply a function to all key-value pairs in a Dict.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert("a", 100)

/// |> dict.insert("b", 200)

/// |> dict.insert("c", 300)

/// |> dict.map(fn(\_k, v) { v \* 2 })

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 200), Pair("b", 400), Pair("c", 600)]

/// ```

pub fn map(self: Dict<key, a>, with: fn(ByteArray, a) -> b) -> Dict<key, b> {

Dict { inner: do\_map(self.inner, with) }

}

fn do\_map(

self: Pairs<ByteArray, a>,

with: fn(ByteArray, a) -> b,

) -> Pairs<ByteArray, b> {

when self is {

[] ->

[]

[Pair(k, v), ..rest] ->

[Pair(k, with(k, v)), ..do\_map(rest, with)]

}

}

test map\_1() {

let result =

fixture\_1()

|> map(with: fn(k, \_) { k })

get(result, foo) == Some(foo)

}

test map\_2() {

let result =

fixture\_1()

|> map(with: fn(\_, v) { v + 1 })

get(result, foo) == Some(43) && size(result) == size(fixture\_1())

}

/// Get the inner list holding the dictionary data.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert("a", 100)

/// |> dict.insert("b", 200)

/// |> dict.insert("c", 300)

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 100), Pair("b", 200), Pair("c", 300)]

/// ```

pub fn to\_pairs(self: Dict<key, value>) -> Pairs<ByteArray, value> {

self.inner

}

test to\_list\_1() {

to\_pairs(new()) == []

}

test to\_list\_2() {

to\_pairs(fixture\_1()) == [Pair(bar, 14), Pair(foo, 42)]

}

/// Return the number of key-value pairs in the dictionary.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert("a", 100)

/// |> dict.insert("b", 200)

/// |> dict.insert("c", 300)

/// |> dict.size()

///

/// result == 3

/// ```

pub fn size(self: Dict<key, value>) -> Int {

do\_size(self.inner)

}

fn do\_size(self: Pairs<key, value>) -> Int {

when self is {

[] -> 0

[\_, ..rest] -> 1 + do\_size(rest)

}

}

test size\_1() {

size(new()) == 0

}

test size\_2() {

size(

new()

|> insert(foo, 14),

) == 1

}

test size\_3() {

size(

new()

|> insert(foo, 14)

|> insert(bar, 42),

) == 2

}

/// Combine two dictionaries. If the same key exist in both the left and

/// right dictionary, values from the left are preferred (i.e. left-biaised).

///

/// ```aiken

/// let left\_dict = dict.from\_pairs([Pair("a", 100), Pair("b", 200)])

/// let right\_dict = dict.from\_pairs([Pair("a", 150), Pair("c", 300)])

///

/// let result =

/// dict.union(left\_dict, right\_dict) |> dict.to\_pairs()

///

/// result == [Pair("a", 100), Pair("b", 200), Pair("c", 300)]

/// ```

pub fn union(

left: Dict<key, value>,

right: Dict<key, value>,

) -> Dict<key, value> {

Dict { inner: do\_union(left.inner, right.inner) }

}

fn do\_union(

left: Pairs<ByteArray, value>,

right: Pairs<ByteArray, value>,

) -> Pairs<ByteArray, value> {

when left is {

[] -> right

[Pair(k, v), ..rest] -> do\_union(rest, do\_insert(right, k, v))

}

}

test union\_1() {

union(fixture\_1(), new()) == fixture\_1()

}

test union\_2() {

union(new(), fixture\_1()) == fixture\_1()

}

test union\_3() {

let left =

new()

|> insert(foo, 14)

let right =

new()

|> insert(bar, 42)

|> insert(baz, 1337)

union(left, right) == from\_pairs(

[Pair(foo, 14), Pair(baz, 1337), Pair(bar, 42)],

)

}

test union\_4() {

let left =

new()

|> insert(foo, 14)

let right =

new()

|> insert(bar, 42)

|> insert(foo, 1337)

union(left, right) == from\_pairs([Pair(foo, 14), Pair(bar, 42)])

}

/// Like [`union`](#union) but allows specifying the behavior to adopt when a key is present

/// in both dictionaries. The first value received correspond to the value in the left

/// dictionnary, whereas the second argument corresponds to the value in the right dictionnary.

///

/// When passing `None`, the value is removed and not present in the union.

///

/// ```aiken

/// let left\_dict = dict.from\_pairs([Pair("a", 100), Pair("b", 200)])

/// let right\_dict = dict.from\_pairs([Pair("a", 150), Pair("c", 300)])

///

/// let result =

/// dict.union\_with(

/// left\_dict,

/// right\_dict,

/// fn(\_k, v1, v2) { Some(v1 + v2) },

/// )

/// |> dict.to\_pairs()

///

/// result == [Pair("a", 250), Pair("b", 200), Pair("c", 300)]

/// ```

pub fn union\_with(

left: Dict<key, value>,

right: Dict<key, value>,

with: fn(ByteArray, value, value) -> Option<value>,

) -> Dict<key, value> {

Dict { inner: do\_union\_with(left.inner, right.inner, with) }

}

fn do\_union\_with(

left: Pairs<ByteArray, value>,

right: Pairs<ByteArray, value>,

with: fn(ByteArray, value, value) -> Option<value>,

) -> Pairs<ByteArray, value> {

when left is {

[] -> right

[Pair(k, v), ..rest] ->

do\_union\_with(rest, do\_insert\_with(right, k, v, with), with)

}

}

fn do\_insert\_with(

self: Pairs<ByteArray, value>,

key k: ByteArray,

value v: value,

with: fn(ByteArray, value, value) -> Option<value>,

) -> Pairs<ByteArray, value> {

when self is {

[] ->

[Pair(k, v)]

[Pair(k2, v2), ..rest] ->

if builtin.less\_than\_bytearray(k, k2) {

[Pair(k, v), ..self]

} else {

if k == k2 {

when with(k, v, v2) is {

Some(combined) ->

[Pair(k, combined), ..rest]

None -> rest

}

} else {

[Pair(k2, v2), ..do\_insert\_with(rest, k, v, with)]

}

}

}

}

test union\_with\_1() {

let left =

new()

|> insert(foo, 14)

let right =

new()

|> insert(bar, 42)

|> insert(foo, 1337)

let result = union\_with(left, right, with: fn(\_, l, r) { Some(l + r) })

result == from\_pairs([Pair(foo, 1351), Pair(bar, 42)])

}

/// Extract all the values present in a given `Dict`.

///

/// ```aiken

/// let result =

/// dict.new()

/// |> dict.insert("a", 14)

/// |> dict.insert("b", 42)

/// |> dict.insert("c", 1337)

/// |> dict.values()

///

/// result == [1337, 42]

/// ```

pub fn values(self: Dict<key, value>) -> List<value> {

do\_values(self.inner)

}

fn do\_values(self: Pairs<key, value>) -> List<value> {

when self is {

[] ->

[]

[Pair(\_, v), ..rest] ->

[v, ..do\_values(rest)]

}

}

test values\_1() {

values(new()) == []

}

test values\_2() {

values(

new()

|> insert(foo, 3)

|> insert(bar, 4),

) == [4, 3]

}

### 2.6. Hash

//// This module defines `Hash`, a self-documenting type-alias with a

//// phantom-type for readability.

////

//// On-chain, any hash digest value is represented as a plain 'ByteArray'.

//// Though in practice, hashes come from different sources and have

//// different semantics.

////

//// Hence, while this type-alias doesn't provide any strong type-guarantees,

//// it helps writing functions signatures with more meaningful types than mere

//// 'ByteArray'.

////

//// Compare for example:

////

//// ```aiken

//// pub type Credential {

//// VerificationKeyCredential(ByteArray)

//// ScriptCredential(ByteArray)

//// }

//// ```

////

//// with

////

//// ```aiken

//// pub type Credential {

//// VerificationKeyCredential(Hash<Blake2b\_224, VerificationKey>)

//// ScriptCredential(Hash<Blake2b\_224, Script>)

//// }

//// ```

////

//// Both are strictly equivalent, but the second reads much better.

use aiken/builtin

/// A `Hash` is nothing more than a `ByteArray`, but it carries extra

/// information for readability.

pub type Hash<alg, a> =

ByteArray

/// A blake2b-224 hash algorithm.

///

/// Typically used for:

///

/// - [`Credential`](../aiken/transaction/credential.html#Credential)

/// - [`PolicyId`](../aiken/transaction/value.html#PolicyId)

///

/// Note: there's no function to calculate blake2b-224 hash digests on-chain.

pub opaque type Blake2b\_224 {

Blake2b\_224

}

/// A blake2b-256 hash algorithm.

///

/// Typically used for:

///

/// - [`TransactionId`](../aiken/transaction.html#TransactionId)

pub opaque type Blake2b\_256 {

Blake2b\_256

}

/// Compute the blake2b-256 hash digest of some data.

pub fn blake2b\_256(bytes: ByteArray) -> Hash<Blake2b\_256, a> {

builtin.blake2b\_256(bytes)

}

/// A SHA2-256 hash algorithm.

pub opaque type Sha2\_256 {

Sha2\_256

}

/// Compute the sha2-256 hash digest of some data.

pub fn sha2\_256(bytes: ByteArray) -> Hash<Sha2\_256, a> {

builtin.sha2\_256(bytes)

}

/// A SHA3-256 hash algorithm.

pub opaque type Sha3\_256 {

Sha3\_256

}

/// Compute the sha3-256 hash digest of some data.

pub fn sha3\_256(bytes: ByteArray) -> Hash<Sha3\_256, a> {

builtin.sha3\_256(bytes)

}

### 2.7 int

use aiken/bytearray

use aiken/math

use aiken/option

/// Compare two integers.

///

/// ```aiken

/// int.compare(14, 42) == Less

/// int.compare(14, 14) == Equal

/// int.compare(42, 14) == Greater

/// ```

pub fn compare(left: Int, right: Int) -> Ordering {

if left < right {

Less

} else if left > right {

Greater

} else {

Equal

}

}

/// Parse an integer from a utf-8 encoded 'ByteArray', when possible.

///

/// ```aiken

/// int.from\_utf8("14") == Some(14)

/// int.from\_utf8("-42") == Some(-42)

/// int.from\_utf8("007") == Some(7)

/// int.from\_utf8("foo") == None

/// int.from\_utf8("1.0") == None

/// int.from\_utf8("1-2") == None

/// ```

pub fn from\_utf8(bytes: ByteArray) -> Option<Int> {

bytes

|> bytearray.foldr(

Some((0, 0)),

fn(byte, st) {

when st is {

None -> None

Some((n, e)) ->

if byte < 48 || byte > 57 {

if byte == 45 {

Some((-n, 0))

} else {

None

}

} else if n < 0 {

None

} else {

let digit = byte - 48

Some((n + digit \* math.pow(10, e), e + 1))

}

}

},

)

|> option.map(fn(tuple) { tuple.1st })

}

test from\_utf8\_1() {

from\_utf8("0017") == Some(17)

}

test from\_utf8\_2() {

from\_utf8("42") == Some(42)

}

test from\_utf8\_3() {

from\_utf8("1337") == Some(1337)

}

test from\_utf8\_4() {

from\_utf8("-14") == Some(-14)

}

test from\_utf8\_5() {

from\_utf8("foo") == None

}

test from\_utf8\_6() {

from\_utf8("1-2") == None

}

### 2.8. Interval

//// In a eUTxO-based blockchain like Cardano, the management of time can be

//// finicky.

////

//// Indeed, in order to maintain a complete determinism in the execution of

//// scripts, it is impossible to introduce a notion of \_"current time"\_ since

//// the execution would then depend on factor that are external to the

//// transaction itself: the ineluctable stream of time flowing in our universe.

////

//// Hence, to work around that, we typically define time intervals, which gives

//// window -- a.k.a intervals -- within which the transaction can be executed.

//// From within a script, it isn't possible to know when exactly the script is

//// executed, but we can reason about the interval bounds to validate pieces of

//// logic.

/// A type to represent intervals of values. Interval are inhabited by a type

/// `a` which is useful for non-infinite intervals that have a finite

/// lower-bound and/or upper-bound.

///

/// This allows to represent all kind of mathematical intervals:

///

/// ```aiken

/// // [1; 10]

/// let i0: Interval<PosixTime> = Interval

/// { lower\_bound:

/// IntervalBound { bound\_type: Finite(1), is\_inclusive: True }

/// , upper\_bound:

/// IntervalBound { bound\_type: Finite(10), is\_inclusive: True }

/// }

/// ```

///

/// ```aiken

/// // (20; infinity)

/// let i1: Interval<PosixTime> = Interval

/// { lower\_bound:

/// IntervalBound { bound\_type: Finite(20), is\_inclusive: False }

/// , upper\_bound:

/// IntervalBound { bound\_type: PositiveInfinity, is\_inclusive: False }

/// }

/// ```

pub type Interval<a> {

lower\_bound: IntervalBound<a>,

upper\_bound: IntervalBound<a>,

}

/// An interval bound, either inclusive or exclusive.

pub type IntervalBound<a> {

bound\_type: IntervalBoundType<a>,

is\_inclusive: Bool,

}

/// Return the highest bound of the two.

///

/// ```aiken

/// let ib1 = IntervalBound { bound\_type: Finite(0), is\_inclusive: False }

/// let ib2 = IntervalBound { bound\_type: Finite(1), is\_inclusive: False }

///

/// interval.max(ib1, ib2) == ib2

/// ```

pub fn max(

left: IntervalBound<Int>,

right: IntervalBound<Int>,

) -> IntervalBound<Int> {

when compare\_bound(left, right) is {

Less -> right

Equal -> left

Greater -> left

}

}

/// Return the smallest bound of the two.

///

/// ```aiken

/// let ib1 = IntervalBound { bound\_type: Finite(0), is\_inclusive: False }

/// let ib2 = IntervalBound { bound\_type: Finite(1), is\_inclusive: False }

///

/// interval.min(ib1, ib2) == ib1

/// ```

pub fn min(

left: IntervalBound<Int>,

right: IntervalBound<Int>,

) -> IntervalBound<Int> {

when compare\_bound(left, right) is {

Less -> left

Equal -> left

Greater -> right

}

}

fn compare\_bound(

left: IntervalBound<Int>,

right: IntervalBound<Int>,

) -> Ordering {

when compare\_bound\_type(left.bound\_type, right.bound\_type) is {

Less -> Less

Greater -> Greater

Equal ->

if left.is\_inclusive == right.is\_inclusive {

Equal

} else if left.is\_inclusive {

Greater

} else {

Less

}

}

}

/// A type of interval bound. Where finite, a value of type `a` must be

/// provided. `a` will typically be an `Int`, representing a number of seconds or

/// milliseconds.

pub type IntervalBoundType<a> {

NegativeInfinity

Finite(a)

PositiveInfinity

}

fn compare\_bound\_type(

left: IntervalBoundType<Int>,

right: IntervalBoundType<Int>,

) -> Ordering {

when left is {

NegativeInfinity ->

when right is {

NegativeInfinity -> Equal

\_ -> Less

}

PositiveInfinity ->

when right is {

PositiveInfinity -> Equal

\_ -> Greater

}

Finite(left) ->

when right is {

NegativeInfinity -> Greater

PositiveInfinity -> Less

Finite(right) ->

if left < right {

Less

} else if left == right {

Equal

} else {

Greater

}

}

}

}

// TODO: Replace 'Int' with a generic 'a' once we have comparable traits.

/// Checks whether an element is contained within the interval.

///

/// ```aiken

/// let iv =

/// Interval {

/// lower\_bound: IntervalBound {

/// bound\_type: Finite(14),

/// is\_inclusive: True

/// },

/// upper\_bound: IntervalBound {

/// bound\_type: Finite(42),

/// is\_inclusive: False

/// },

/// }

///

/// interval.contains(iv, 25) == True

/// interval.contains(iv, 0) == False

/// interval.contains(iv, 14) == True

/// interval.contains(iv, 42) == False

/// ```

pub fn contains(self: Interval<Int>, elem: Int) -> Bool {

let is\_greater\_than\_lower\_bound =

when self.lower\_bound.bound\_type is {

NegativeInfinity -> True

Finite(lower\_bound) ->

if self.lower\_bound.is\_inclusive {

elem >= lower\_bound

} else {

elem > lower\_bound

}

PositiveInfinity -> False

}

let is\_smaller\_than\_upper\_bound =

when self.upper\_bound.bound\_type is {

NegativeInfinity -> False

Finite(upper\_bound) ->

if self.upper\_bound.is\_inclusive {

elem <= upper\_bound

} else {

elem < upper\_bound

}

PositiveInfinity -> True

}

is\_greater\_than\_lower\_bound && is\_smaller\_than\_upper\_bound

}

test contains\_1() {

let iv = everything()

contains(iv, 14)

}

test contains\_2() {

let iv = entirely\_before(15)

contains(iv, 14)

}

test contains\_3() {

let iv = before(14)

contains(iv, 14)

}

test contains\_4() {

let iv = entirely\_before(14)

!contains(iv, 14)

}

test contains\_5() {

let iv = entirely\_after(13)

contains(iv, 14)

}

test contains\_6() {

let iv = after(14)

contains(iv, 14)

}

test contains\_7() {

let iv = entirely\_after(14)

!contains(iv, 14)

}

test contains\_8() {

let iv = between(42, 1337)

!contains(iv, 14)

}

test contains\_9() {

let iv = between(0, 42)

contains(iv, 14)

}

test contains\_10() {

let iv = between(0, 42)

contains(iv, 42)

}

test contains\_11() {

let iv = entirely\_between(0, 42)

!contains(iv, 0)

}

test contains\_12() {

let iv = empty()

!contains(iv, 14)

}

/// Create an interval that contains every possible values. i.e. (-INF, +INF)

///

/// ```aiken

/// interval.contains(everything(), 0) == True

/// interval.contains(everything(), 1000) == True

/// ```

pub fn everything() -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: NegativeInfinity,

is\_inclusive: True,

},

upper\_bound: IntervalBound {

bound\_type: PositiveInfinity,

is\_inclusive: True,

},

}

}

/// Create an empty interval that contains no value.

///

/// ```aiken

/// interval.contains(empty(), 0) == False

/// interval.contains(empty(), 1000) == False

/// ```

pub fn empty() -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: PositiveInfinity,

is\_inclusive: True,

},

upper\_bound: IntervalBound {

bound\_type: NegativeInfinity,

is\_inclusive: True,

},

}

}

/// Create an interval that includes all values between two bounds, including the bounds. i.e. [lower\_bound, upper\_bound]

///

/// ```aiken

/// interval.between(10, 100) == Interval {

/// lower\_bound: IntervalBound { bound\_type: Finite(10), is\_inclusive: True },

/// upper\_bound: IntervalBound { bound\_type: Finite(100), is\_inclusive: True },

/// }

/// ```

pub fn between(lower\_bound: a, upper\_bound: a) -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: Finite(lower\_bound),

is\_inclusive: True,

},

upper\_bound: IntervalBound {

bound\_type: Finite(upper\_bound),

is\_inclusive: True,

},

}

}

/// Create an interval that includes all values between two bounds, excluding the bounds. i.e. (lower\_bound, upper\_bound)

///

/// ```aiken

/// interval.entirely\_between(10, 100) == Interval {

/// lower\_bound: IntervalBound { bound\_type: Finite(10), is\_inclusive: False },

/// upper\_bound: IntervalBound { bound\_type: Finite(100), is\_inclusive: False },

/// }

/// ```

pub fn entirely\_between(lower\_bound: a, upper\_bound: a) -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: Finite(lower\_bound),

is\_inclusive: False,

},

upper\_bound: IntervalBound {

bound\_type: Finite(upper\_bound),

is\_inclusive: False,

},

}

}

/// Create an interval that includes all values greater than the given bound. i.e [lower\_bound, +INF)

///

/// ```aiken

/// interval.after(10) == Interval {

/// lower\_bound: IntervalBound { bound\_type: Finite(10), is\_inclusive: True },

/// upper\_bound: IntervalBound { bound\_type: PositiveInfinity, is\_inclusive: True },

/// }

/// ```

pub fn after(lower\_bound: a) -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: Finite(lower\_bound),

is\_inclusive: True,

},

upper\_bound: IntervalBound {

bound\_type: PositiveInfinity,

is\_inclusive: True,

},

}

}

// TODO: Replace 'Int' with a generic 'a' once we have comparable traits.

/// Check whether the interval is entirely after the point "a"

///

/// ```aiken

/// interval.is\_entirely\_after(interval.after(10), 5) == True

/// interval.is\_entirely\_after(interval.after(10), 10) == False

/// interval.is\_entirely\_after(interval.after(10), 15) == False

/// interval.is\_entirely\_after(interval.between(10, 20), 30) == False

/// interval.is\_entirely\_after(interval.between(10, 20), 5) == True

pub fn is\_entirely\_after(self: Interval<Int>, point: Int) -> Bool {

when self.lower\_bound.bound\_type is {

Finite(low) ->

if self.lower\_bound.is\_inclusive {

point < low

} else {

point <= low

}

\_ -> False

}

}

test is\_entirely\_after\_1() {

is\_entirely\_after(after(10), 5)

}

test is\_entirely\_after\_2() {

!is\_entirely\_after(after(10), 10)

}

test is\_entirely\_after\_3() {

!is\_entirely\_after(after(10), 15)

}

test is\_entirely\_after\_4() {

!is\_entirely\_after(between(10, 20), 30)

}

test is\_entirely\_after\_5() {

is\_entirely\_after(between(10, 20), 5)

}

test is\_entirely\_after\_6() {

is\_entirely\_after(entirely\_after(10), 10)

}

test is\_entirely\_after\_7() {

!is\_entirely\_after(before(10), 5)

}

test is\_entirely\_after\_8() {

!is\_entirely\_after(before(10), 15)

}

test is\_entirely\_after\_9() {

!is\_entirely\_after(entirely\_before(10), 5)

}

// TODO: Replace 'Int' with a generic 'a' once we have comparable traits.

/// Check whether the interval is entirely before the point "a"

///

/// ```aiken

/// interval.is\_entirely\_before(interval.before(10), 15) == True

/// interval.is\_entirely\_before(interval.before(10), 10) == False

/// interval.is\_entirely\_before(interval.before(10), 5) == False

/// interval.is\_entirely\_before(interval.between(10, 20), 30) == True

/// interval.is\_entirely\_before(interval.between(10, 20), 5) == False

pub fn is\_entirely\_before(self: Interval<Int>, point: Int) -> Bool {

when self.upper\_bound.bound\_type is {

Finite(hi) ->

if self.upper\_bound.is\_inclusive {

hi < point

} else {

hi <= point

}

\_ -> False

}

}

test is\_entirely\_before\_1() {

is\_entirely\_before(before(10), 15)

}

test is\_entirely\_before\_2() {

!is\_entirely\_before(before(10), 10)

}

test is\_entirely\_before\_3() {

!is\_entirely\_before(before(10), 5)

}

test is\_entirely\_before\_4() {

is\_entirely\_before(between(10, 20), 30)

}

test is\_entirely\_before\_5() {

!is\_entirely\_before(between(10, 20), 5)

}

test is\_entirely\_before\_6() {

is\_entirely\_before(entirely\_before(10), 10)

}

test is\_entirely\_before\_7() {

!is\_entirely\_before(after(10), 15)

}

test is\_entirely\_before\_8() {

!is\_entirely\_before(after(10), 5)

}

test is\_entirely\_before\_9() {

!is\_entirely\_before(entirely\_after(10), 5)

}

/// Create an interval that includes all values after (and not including) the given bound. i.e (lower\_bound, +INF)

///

/// ```aiken

/// interval.entirely\_after(10) == Interval {

/// lower\_bound: IntervalBound { bound\_type: Finite(10), is\_inclusive: False },

/// upper\_bound: IntervalBound { bound\_type: PositiveInfinity, is\_inclusive: True },

/// }

/// ```

pub fn entirely\_after(lower\_bound: a) -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: Finite(lower\_bound),

is\_inclusive: False,

},

upper\_bound: IntervalBound {

bound\_type: PositiveInfinity,

is\_inclusive: True,

},

}

}

/// Create an interval that includes all values before (and including) the given bound. i.e (-INF, upper\_bound]

///

/// ```aiken

/// interval.before(100) == Interval {

/// lower\_bound: IntervalBound { bound\_type: NegativeInfinity, is\_inclusive: True },

/// upper\_bound: IntervalBound { bound\_type: Finite(100), is\_inclusive: True },

/// }

/// ```

pub fn before(upper\_bound: a) -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: NegativeInfinity,

is\_inclusive: True,

},

upper\_bound: IntervalBound {

bound\_type: Finite(upper\_bound),

is\_inclusive: True,

},

}

}

/// Create an interval that includes all values before (and not including) the given bound. i.e (-INF, upper\_bound)

///

/// ```aiken

/// interval.entirely\_before(10) == Interval {

/// lower\_bound: IntervalBound { bound\_type: NegativeInfinity, is\_inclusive: True },

/// upper\_bound: IntervalBound { bound\_type: Finite(10), is\_inclusive: False },

/// }

/// ```

pub fn entirely\_before(upper\_bound: a) -> Interval<a> {

Interval {

lower\_bound: IntervalBound {

bound\_type: NegativeInfinity,

is\_inclusive: True,

},

upper\_bound: IntervalBound {

bound\_type: Finite(upper\_bound),

is\_inclusive: False,

},

}

}

/// Tells whether an interval is empty; i.e. that is contains no value.

///

/// ```aiken

/// let iv1 = interval.empty()

///

/// let iv2 = Interval {

/// lower\_bound: IntervalBound { bound\_type: Finite(0), is\_inclusive: False },

/// upper\_bound: IntervalBound { bound\_type: Finite(0), is\_inclusive: False },

/// }

///

/// let iv3 = Interval {

/// lower\_bound: IntervalBound { bound\_type: Finite(0), is\_inclusive: False },

/// upper\_bound: IntervalBound { bound\_type: Finite(100), is\_inclusive: False },

/// }

///

/// interval.is\_empty(iv1) == True

/// interval.is\_empty(iv2) == True

/// interval.is\_empty(iv3) == False

///

/// // Note: Two empty intervals are not necessarily equal.

/// iv1 != iv2

/// ```

pub fn is\_empty(self: Interval<Int>) -> Bool {

let ordering =

compare\_bound\_type(self.lower\_bound.bound\_type, self.upper\_bound.bound\_type)

when ordering is {

Greater -> True

Equal -> !(self.lower\_bound.is\_inclusive && self.upper\_bound.is\_inclusive)

Less -> {

let is\_open\_interval =

!self.lower\_bound.is\_inclusive && !self.upper\_bound.is\_inclusive

if is\_open\_interval {

when (self.lower\_bound.bound\_type, self.upper\_bound.bound\_type) is {

(Finite(lower\_bound), Finite(upper\_bound)) ->

lower\_bound + 1 == upper\_bound

\_ -> False

}

} else {

False

}

}

}

}

/// Computes the largest interval contains in the two given intervals, if any.

///

/// ```aiken

/// let iv1 = interval.between(0, 10)

/// let iv2 = interval.between(2, 14)

/// interval.intersection(iv1, iv2) == interval.between(2, 10)

///

/// let iv1 = interval.entirely\_before(10)

/// let iv2 = interval.entirely\_after(0)

/// interval.intersection(iv1, iv2) == interval.entirely\_between(0, 10)

///

/// let iv1 = interval.between(0, 1)

/// let iv2 = interval.between(2, 3)

/// interval.intersection(iv1, iv2) |> interval.is\_empty

/// ```

pub fn intersection(iv1: Interval<Int>, iv2: Interval<Int>) -> Interval<Int> {

Interval {

lower\_bound: max(iv1.lower\_bound, iv2.lower\_bound),

upper\_bound: min(iv1.upper\_bound, iv2.upper\_bound),

}

}

test intersection\_1() {

let iv1 = between(0, 10)

let iv2 = between(2, 14)

intersection(iv1, iv2) == between(2, 10)

}

test intersection\_2() {

let iv1 = between(0, 1)

let iv2 = between(1, 2)

intersection(iv1, iv2) == between(1, 1)

}

test intersection\_3() {

let iv1 = between(0, 1)

let iv2 = entirely\_between(1, 2)

intersection(iv1, iv2)

|> is\_empty

}

test intersection\_4() {

let iv1 = entirely\_between(0, 1)

let iv2 = entirely\_between(1, 2)

intersection(iv1, iv2)

|> is\_empty

}

test intersection\_5() {

let iv1 = between(0, 10)

let iv2 = before(4)

intersection(iv1, iv2) == between(0, 4)

}

test intersection\_6() {

let iv1 = entirely\_before(10)

let iv2 = entirely\_after(0)

intersection(iv1, iv2) == entirely\_between(0, 10)

}

/// Computes the smallest interval containing the two given intervals, if any

///

/// ```aiken

/// let iv1 = between(0, 10)

/// let iv2 = between(2, 14)

/// hull(iv1, iv2) == between(0, 14)

///

/// let iv1 = between(5, 10)

/// let iv2 = before(0)

/// hull(iv1, iv2) == before(10)

///

/// let iv1 = entirely\_after(0)

/// let iv2 = between(10, 42)

/// hull(iv1, iv2) = entirely\_after(0)

/// ```

pub fn hull(iv1: Interval<Int>, iv2: Interval<Int>) -> Interval<Int> {

Interval {

lower\_bound: min(iv1.lower\_bound, iv2.lower\_bound),

upper\_bound: max(iv1.upper\_bound, iv2.upper\_bound),

}

}

test hull\_1() {

let iv1 = between(0, 10)

let iv2 = between(2, 14)

hull(iv1, iv2) == between(0, 14)

}

test hull\_2() {

let iv1 = between(5, 10)

let iv2 = before(0)

hull(iv1, iv2) == before(10)

}

test hull\_3() {

let iv1 = entirely\_after(0)

let iv2 = between(10, 42)

hull(iv1, iv2) == entirely\_after(0)

}

### 2.9. List

use aiken/builtin

use aiken/bytearray

use aiken/int

/// Determine if all elements of the list satisfy the given predicate.

///

/// Note: an empty list always satisfies the predicate.

///

/// ```aiken

/// list.all([], fn(n) { n > 0 }) == True

/// list.all([1, 2, 3], fn(n) { n > 0 }) == True

/// list.all([1, 2, 3], fn(n) { n == 2 }) == False

/// ```

pub fn all(self: List<a>, predicate: fn(a) -> Bool) -> Bool {

when self is {

[] -> True

[x, ..xs] -> predicate(x) && all(xs, predicate)

}

}

test all\_1() {

all([1, 2, 3], fn(n) { n > 0 }) == True

}

test all\_2() {

all([1, 2, 3], fn(n) { n > 42 }) == False

}

test all\_3() {

all([], fn(n) { n == 42 }) == True

}

/// Determine if at least one element of the list satisfies the given predicate.

///

/// Note: an empty list never satisfies the predicate.

///

/// ```aiken

/// list.any([], fn(n) { n > 2 }) == False

/// list.any([1, 2, 3], fn(n) { n > 0 }) == True

/// list.any([1, 2, 3], fn(n) { n == 2 }) == True

/// list.any([1, 2, 3], fn(n) { n < 0 }) == False

/// ```

pub fn any(self: List<a>, predicate: fn(a) -> Bool) -> Bool {

when self is {

[] -> False

[x, ..xs] -> predicate(x) || any(xs, predicate)

}

}

test any\_1() {

any([1, 2, 3], fn(n) { n > 0 }) == True

}

test any\_2() {

any([1, 2, 3], fn(n) { n > 42 }) == False

}

test any\_3() {

any([], fn(n) { n == 42 }) == False

}

/// Count how many items in the list satisfy the given predicate.

///

/// ```aiken

/// list.count([], fn(a) { a > 2}) == 0

/// list.count([1, 2, 3], fn(a) { n > 0 }) == 3

/// list.count([1, 2, 3], fn(a) { n >= 2 }) == 2

/// list.count([1, 2, 3], fn(a) { n > 5 }) == 0

/// ```

pub fn count(self: List<a>, predicate: fn(a) -> Bool) -> Int {

foldr(

self,

0,

fn(item, total) {

if predicate(item) {

total + 1

} else {

total

}

},

)

}

test count\_empty() {

count([], fn(a) { a > 2 }) == 0

}

test count\_all() {

count([1, 2, 3], fn(a) { a > 0 }) == 3

}

test count\_some() {

count([1, 2, 3], fn(a) { a >= 2 }) == 2

}

test count\_none() {

count([1, 2, 3], fn(a) { a > 5 }) == 0

}

/// Return Some(item) at the index or None if the index is out of range. The index is 0-based.

///

/// ```aiken

/// list.at([1, 2, 3], 1) == Some(2)

/// list.at([1, 2, 3], 42) == None

/// ```

pub fn at(self: List<a>, index: Int) -> Option<a> {

when self is {

[] -> None

[x, ..xs] ->

if index == 0 {

Some(x)

} else {

at(xs, index - 1)

}

}

}

test at\_1() {

at([1, 2, 3], -1) == None

}

test at\_2() {

at([], 0) == None

}

test at\_3() {

at([1, 2, 3], 3) == None

}

test at\_4() {

at([1], 0) == Some(1)

}

test at\_5() {

at([1, 2, 3], 2) == Some(3)

}

/// Merge two lists together.

///

/// ```aiken

/// list.concat([], []) == []

/// list.concat([], [1, 2, 3]) == [1, 2, 3]

/// list.concat([1, 2, 3], [4, 5, 6]) == [1, 2, 3, 4, 5, 6]

/// ```

pub fn concat(left: List<a>, right: List<a>) -> List<a> {

when left is {

[] -> right

[x, ..xs] ->

[x, ..concat(xs, right)]

}

}

test concat\_1() {

concat([1, 2, 3], [4, 5, 6]) == [1, 2, 3, 4, 5, 6]

}

test concat\_2() {

concat([1, 2, 3], []) == [1, 2, 3]

}

test concat\_3() {

concat([], [1, 2, 3]) == [1, 2, 3]

}

/// Remove the first occurrence of the given element from the list.

///

/// ```aiken

/// list.delete([1, 2, 3, 1], 1) == [2, 3, 1]

/// list.delete([1, 2, 3], 14) == [1, 2, 3]

/// ```

pub fn delete(self: List<a>, elem: a) -> List<a> {

when self is {

[] ->

[]

[x, ..xs] ->

if x == elem {

xs

} else {

[x, ..delete(xs, elem)]

}

}

}

test delete\_1() {

delete([], 42) == []

}

test delete\_2() {

delete([1, 2, 3, 1], 1) == [2, 3, 1]

}

test delete\_3() {

delete([1, 2, 3], 14) == [1, 2, 3]

}

test delete\_4() {

delete([2], 2) == []

}

/// Remove the first occurrence of each element of the second list from the first one.

///

/// ```

/// list.difference(["h", "e", "l", "l", "o"], ["l", "e", "l"]) == ["h", "o"]

/// list.difference([1, 2, 3, 4, 5], [1, 1, 2]) == [3, 4, 5]

/// list.difference([1, 2, 3], []) == [1, 2, 3]

/// ```

pub fn difference(self: List<a>, with: List<a>) -> List<a> {

when with is {

[] -> self

[x, ..xs] -> difference(delete(self, x), xs)

}

}

test difference\_1() {

difference(["h", "e", "l", "l", "o"], ["l", "e", "l"]) == ["h", "o"]

}

test difference\_2() {

difference([1, 2, 3, 4, 5], [1, 1, 2]) == [3, 4, 5]

}

test difference\_3() {

difference([1, 2, 3], []) == [1, 2, 3]

}

test difference\_4() {

difference([], [1, 2, 3]) == []

}

/// Drop the first `n` elements of a list.

///

/// ```aiken

/// list.drop([1, 2, 3], 2) == [3]

/// list.drop([], 42) == []

/// list.drop([1, 2, 3], 42) == []

/// ```

pub fn drop(self: List<a>, n: Int) -> List<a> {

if n <= 0 {

self

} else {

when self is {

[] ->

[]

[\_x, ..xs] -> drop(xs, n - 1)

}

}

}

test drop\_1() {

drop([], 42) == []

}

test drop\_2() {

drop([1, 2, 3], 2) == [3]

}

/// Returns the suffix of the given list after removing all elements that satisfy the predicate.

///

/// ```aiken

/// list.drop\_while([1, 2, 3], fn(x) { x < 2 }) == [2, 3]

/// list.drop\_while([], fn(x) { x > 2 }) == []

/// list.drop\_while([1, 2, 3], fn(x) { x == 3 }) == [1, 2, 3]

/// ```

pub fn drop\_while(self: List<a>, predicate: fn(a) -> Bool) -> List<a> {

when self is {

[] ->

[]

[x, ..xs] ->

if predicate(x) {

drop\_while(xs, predicate)

} else {

self

}

}

}

test drop\_while\_1() {

drop\_while([], fn(x) { x > 2 }) == []

}

test drop\_while\_2() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

drop\_while(xs, fn(x) { x > 5 }) == [5, 4, 3, 2, 1]

}

test drop\_while\_3() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

drop\_while(xs, fn(x) { x == 42 }) == xs

}

test drop\_while\_4() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

drop\_while(xs, fn(x) { x < 42 }) == []

}

/// Produce a list of elements that satisfy a predicate.

///

/// ```aiken

/// list.filter([1, 2, 3], fn(x) { x >= 2 }) == [2, 3]

/// list.filter([], fn(x) { x > 2 }) == []

/// list.filter([1, 2, 3], fn(x) { x == 3 }) == [3]

/// ```

pub fn filter(self: List<a>, predicate: fn(a) -> Bool) -> List<a> {

when self is {

[] ->

[]

[x, ..xs] ->

if predicate(x) {

[x, ..filter(xs, predicate)]

} else {

filter(xs, predicate)

}

}

}

test filter\_1() {

filter([], fn(x) { x > 0 }) == []

}

test filter\_2() {

let xs =

[1, 2, 3, 4, 5, 6]

filter(xs, fn(x) { builtin.mod\_integer(x, 2) == 0 }) == [2, 4, 6]

}

test filter\_3() {

let filter\_foldr =

fn(xs, f) {

foldr(

xs,

[],

fn(x, ys) {

if f(x) {

[x, ..ys]

} else {

ys

}

},

)

}

let is\_odd =

fn(n) { builtin.mod\_integer(n, 2) != 0 }

filter\_foldr([1, 2, 3], is\_odd) == filter([1, 2, 3], is\_odd)

}

/// Produce a list of transformed elements that satisfy a predicate.

///

/// ```aiken

/// let transform = fn(x) { if x % 2 == 0 { None } else { Some(3\*x) } }

/// list.filter\_map([1, 2, 3], transform) == [3, 9]

/// ```

pub fn filter\_map(self: List<a>, predicate: fn(a) -> Option<b>) -> List<b> {

when self is {

[] ->

[]

[x, ..xs] ->

when predicate(x) is {

None -> filter\_map(xs, predicate)

Some(y) ->

[y, ..filter\_map(xs, predicate)]

}

}

}

test filter\_map\_1() {

filter\_map([], fn(\_) { Some(42) }) == []

}

test filter\_map\_2() {

filter\_map(

[1, 2, 3, 4, 5, 6],

fn(x) {

if builtin.mod\_integer(x, 2) != 0 {

Some(3 \* x)

} else {

None

}

},

) == [3, 9, 15]

}

/// Find the first element satisfying the given predicate, if any.

///

/// ```aiken

/// list.find([1, 2, 3], fn(x) { x == 2 }) == Some(2)

/// list.find([4, 5, 6], fn(x) { x == 2 }) == None

/// ```

pub fn find(self: List<a>, predicate: fn(a) -> Bool) -> Option<a> {

when self is {

[] -> None

[x, ..xs] ->

if predicate(x) {

Some(x)

} else {

find(xs, predicate)

}

}

}

test find\_1() {

find([1, 2, 3], fn(x) { x == 1 }) == Some(1)

}

test find\_2() {

find([1, 2, 3], fn(x) { x > 42 }) == None

}

test find\_3() {

find([], fn(\_) { True }) == None

}

/// Map elements of a list into a new list and flatten the result.

///

/// ```aiken

/// list.flat\_map([1, 2, 3], fn(a) { [a, 2\*a] }) == [1, 2, 2, 4, 3, 6]

/// ```

pub fn flat\_map(self: List<a>, with: fn(a) -> List<b>) -> List<b> {

foldr(self, [], fn(x, xs) { concat(with(x), xs) })

}

test flat\_map\_1() {

flat\_map([], fn(a) { [a] }) == []

}

test flat\_map\_2() {

flat\_map([1, 2, 3], fn(a) { [a, a] }) == [1, 1, 2, 2, 3, 3]

}

/// Reduce a list from left to right.

///

/// ```aiken

/// list.foldl([1, 2, 3], 0, fn(n, total) { n + total }) == 6

/// list.foldl([1, 2, 3], [], fn(x, xs) { [x, ..xs] }) == [3, 2, 1]

/// ```

pub fn foldl(self: List<a>, zero: b, with: fn(a, b) -> b) -> b {

when self is {

[] -> zero

[x, ..xs] -> foldl(xs, with(x, zero), with)

}

}

test foldl\_1() {

foldl([], 0, fn(\_, \_) { 1 }) == 0

}

test foldl\_2() {

foldl([1, 2, 3, 4, 5], 0, fn(n, total) { n + total }) == 15

}

test foldl\_3() {

foldl([1, 2, 3, 4], [], fn(x, xs) { [x, ..xs] }) == [4, 3, 2, 1]

}

/// Reduce a list from right to left.

///

/// ```aiken

/// list.foldr([1, 2, 3], 0, fn(n, total) { n + total }) == 6

/// list.foldr([1, 2, 3], [], fn(x, xs) { [x, ..xs] }) == [1, 2, 3]

/// ```

pub fn foldr(self: List<a>, zero: b, with: fn(a, b) -> b) -> b {

when self is {

[] -> zero

[x, ..xs] -> with(x, foldr(xs, zero, with))

}

}

test foldr\_1() {

foldr([1, 2, 3, 4, 5], 0, fn(n, total) { n + total }) == 15

}

test foldr\_2() {

foldr(

[1, 2, 3],

"",

fn(n, \_str) {

if builtin.mod\_integer(n, 2) == 0 {

"foo"

} else {

"bar"

}

},

) == "bar"

}

test foldr\_3() {

foldr([1, 2, 3, 4], [], fn(x, xs) { [x, ..xs] }) == [1, 2, 3, 4]

}

/// Return all elements except the last one.

///

/// ```aiken

/// list.init([]) == None

/// list.init([1, 2, 3]) == Some([1, 2])

/// ```

pub fn init(self: List<a>) -> Option<List<a>> {

when self is {

[] -> None

\_ -> Some(do\_init(self))

}

}

fn do\_init(self: List<a>) -> List<a> {

when self is {

[] -> fail @"unreachable"

[\_] ->

[]

[x, ..xs] ->

[x, ..do\_init(xs)]

}

}

test init\_1() {

init([]) == None

}

test init\_2() {

init([1]) == Some([])

}

test init\_3() {

init([1, 2, 3, 4]) == Some([1, 2, 3])

}

/// Figures out whether a list contain the given element.

///

/// ```aiken

/// list.has([1, 2, 3], 2) == True

/// list.has([1, 2, 3], 14) == False

/// list.has([], 14) == False

/// ```

pub fn has(self: List<a>, elem: a) -> Bool {

when self is {

[] -> False

[x, ..xs] ->

if x == elem {

True

} else {

has(xs, elem)

}

}

}

test has\_1() {

has([1, 2, 3], 1) == True

}

test has\_2() {

has([1, 2, 3], 14) == False

}

test has\_3() {

has([], 14) == False

}

/// Gets the index of an element of a list, if any. Otherwise, returns None.

///

/// ```aiken

/// list.index\_of([1, 5, 2], 2) == Some(2)

/// list.index\_of([1, 7, 3], 4) == None

/// list.index\_of([1, 0, 9, 6], 6) == 3

/// list.index\_of([], 6) == None

/// ```

pub fn index\_of(self: List<a>, elem: a) -> Option<Int> {

do\_index\_of(self, elem, 0)

}

fn do\_index\_of(self: List<a>, elem: a, i: Int) -> Option<Int> {

when self is {

[] -> None

[x, ..xs] ->

if x == elem {

Some(i)

} else {

do\_index\_of(xs, elem, i + 1)

}

}

}

test index\_of\_1() {

index\_of([1, 5, 2], 2) == Some(2)

}

test index\_of\_2() {

index\_of([1, 7, 3], 4) == None

}

test index\_of\_3() {

index\_of([1, 0, 9, 6], 6) == Some(3)

}

test index\_of\_4() {

index\_of([], 6) == None

}

/// Get the first element of a list

///

/// ```aiken

/// list.head([1, 2, 3]) == Some(1)

/// list.head([]) == None

/// ```

pub fn head(self: List<a>) -> Option<a> {

when self is {

[] -> None

\_ -> Some(builtin.head\_list(self))

}

}

test head\_1() {

head([1, 2, 3]) == Some(1)

}

test head\_2() {

head([]) == None

}

/// Like [`foldr`](#foldr), but also provides the position (0-based) of the elements when iterating.

///

/// ```aiken

/// let group = fn(i, x, xs) { [(i, x), ..xs] }

/// list.indexed\_foldr(["a", "b", "c"], [], group) == [

/// (0, "a"),

/// (1, "b"),

/// (2, "c")

/// ]

/// ```

pub fn indexed\_foldr(

self: List<a>,

zero: result,

with: fn(Int, a, result) -> result,

) -> result {

do\_indexed\_foldr(0, self, zero, with)

}

fn do\_indexed\_foldr(

n: Int,

self: List<a>,

zero: result,

with: fn(Int, a, result) -> result,

) -> result {

when self is {

[] -> zero

[x, ..xs] -> with(n, x, do\_indexed\_foldr(n + 1, xs, zero, with))

}

}

test indexed\_foldr\_1() {

indexed\_foldr([], 0, fn(i, x, xs) { i + x + xs }) == 0

}

test indexed\_foldr\_2() {

let letters =

["a", "b", "c"]

indexed\_foldr(letters, [], fn(i, x, xs) { [(i, x), ..xs] }) == [

(0, "a"),

(1, "b"),

(2, "c"),

]

}

/// List [`map`](#map) but provides the position (0-based) of the elements while iterating.

///

/// ```aiken

/// list.indexed\_map([1, 2, 3], fn(i, x) { i + x }) == [1, 3, 5]

/// ```

pub fn indexed\_map(self: List<a>, with: fn(Int, a) -> result) -> List<result> {

do\_indexed\_map(0, self, with)

}

fn do\_indexed\_map(

n: Int,

self: List<a>,

with: fn(Int, a) -> result,

) -> List<result> {

when self is {

[] ->

[]

[x, ..xs] ->

[with(n, x), ..do\_indexed\_map(n + 1, xs, with)]

}

}

test indexed\_map\_1() {

indexed\_map([], fn(i, \_n) { i }) == []

}

test indexed\_map\_2() {

indexed\_map(

[4, 8, 13, 2],

fn(i, n) {

if n == 8 {

n

} else {

i

}

},

) == [0, 8, 2, 3]

}

/// Checks whether a list is empty.

///

/// ```aiken

/// list.is\_empty([]) == True

/// list.is\_empty([1, 2, 3]) == False

/// ```

pub fn is\_empty(self: List<a>) -> Bool {

when self is {

[] -> True

\_ -> False

}

}

test is\_empty\_1() {

is\_empty([]) == True

}

test is\_empty\_2() {

is\_empty([1, 2, 3]) == False

}

/// Get the last in the given list, if any.

///

/// ```aiken

/// list.last([]) == None

/// list.last([1, 2, 3]) == Some(3)

/// ```

pub fn last(self: List<a>) -> Option<a> {

when self is {

[] -> None

[x] -> Some(x)

[\_, ..xs] -> last(xs)

}

}

test last\_1() {

last([]) == None

}

test last\_2() {

last([1]) == Some(1)

}

test last\_3() {

last([1, 2, 3, 4]) == Some(4)

}

/// Get the number of elements in the given list.

///

/// ```aiken

/// list.length([]) == 0

/// list.length([1, 2, 3]) == 3

/// ```

pub fn length(self: List<a>) -> Int {

when self is {

[] -> 0

[\_, ..xs] -> 1 + length(xs)

}

}

test length\_1() {

length([]) == 0

}

test length\_2() {

length([1, 2, 3]) == 3

}

/// Apply a function to each element of a list.

///

/// ```aiken

/// list.map([1, 2, 3, 4], fn(n) { n + 1 }) == [2, 3, 4, 5]

/// ```

pub fn map(self: List<a>, with: fn(a) -> result) -> List<result> {

when self is {

[] ->

[]

[x, ..xs] ->

[with(x), ..map(xs, with)]

}

}

test map\_1() {

map([], fn(n) { n + 1 }) == []

}

test map\_2() {

map([1, 2, 3, 4], fn(n) { n + 1 }) == [2, 3, 4, 5]

}

/// Apply a function of two arguments, combining elements from two lists.

///

/// Note: if one list is longer, the extra elements are dropped.

///

/// ```aiken

/// list.map2([1, 2, 3], [1, 2], fn(a, b) { a + b }) == [2, 4]

/// ```

pub fn map2(

self: List<a>,

bs: List<b>,

with: fn(a, b) -> result,

) -> List<result> {

when self is {

[] ->

[]

[x, ..xs] ->

when bs is {

[] ->

[]

[y, ..ys] ->

[with(x, y), ..map2(xs, ys, with)]

}

}

}

test map2\_1() {

map2([], [1, 2, 3], fn(a, b) { a + b }) == []

}

test map2\_2() {

map2([1, 2, 3], [1, 2], fn(a, b) { a + b }) == [2, 4]

}

test map2\_3() {

map2([42], [1, 2, 3], fn(\_a, b) { Some(b) }) == [Some(1)]

}

/// Apply a function of three arguments, combining elements from three lists.

///

/// Note: if one list is longer, the extra elements are dropped.

///

/// ```aiken

/// list.map3([1, 2, 3], [1, 2], [1, 2, 3], fn(a, b, c) { a + b + c }) == [3, 6]

/// ```

pub fn map3(

self: List<a>,

bs: List<b>,

cs: List<c>,

with: fn(a, b, c) -> result,

) -> List<result> {

when self is {

[] ->

[]

[x, ..xs] ->

when bs is {

[] ->

[]

[y, ..ys] ->

when cs is {

[] ->

[]

[z, ..zs] ->

[with(x, y, z), ..map3(xs, ys, zs, with)]

}

}

}

}

test map3\_1() {

map3([], [], [1, 2, 3], fn(a, b, c) { a + b + c }) == []

}

test map3\_2() {

map3([1, 2, 3], [1, 2], [1, 2, 3], fn(a, b, c) { a + b + c }) == [3, 6]

}

/// Add an element in front of the list. Sometimes useful when combined with

/// other functions.

///

/// ```aiken

/// list.push([2, 3], 1) == [1, ..[2, 3]] == [1, 2, 3]

/// ```

pub fn push(self: List<a>, elem: a) -> List<a> {

[elem, ..self]

}

test push\_1() {

push([], 0) == [0]

}

test push\_2() {

push([2, 3], 1) == [1, 2, 3]

}

/// Construct a list of a integer from a given range.

///

/// ```aiken

/// list.range(0, 3) == [0, 1, 2, 3]

/// list.range(-1, 1) == [-1, 0, 1]

/// ```

pub fn range(from: Int, to: Int) -> List<Int> {

if from > to {

[]

} else {

[from, ..range(from + 1, to)]

}

}

test range\_1() {

range(0, 3) == [0, 1, 2, 3]

}

test range\_2() {

range(-1, 1) == [-1, 0, 1]

}

/// Construct a list filled with n copies of a value.

///

/// ```aiken

/// list.repeat("na", 3) == ["na", "na", "na"]

/// ```

pub fn repeat(elem: a, n\_times: Int) -> List<a> {

if n\_times <= 0 {

[]

} else {

[elem, ..repeat(elem, n\_times - 1)]

}

}

test repeat\_1() {

repeat(42, 0) == []

}

test repeat\_2() {

repeat(14, 3) == [14, 14, 14]

}

/// Return the list with its elements in the reserve order.

///

/// ```aiken

/// list.reverse([1, 2, 3]) == [3, 2, 1]

/// ```

pub fn reverse(self: List<a>) -> List<a> {

foldl(self, [], fn(x, xs) { [x, ..xs] })

}

test reverse\_1() {

reverse([]) == []

}

test reverse\_2() {

reverse([1, 2, 3]) == [3, 2, 1]

}

/// Returns a tuple with all elements that satisfy the predicate at first

/// element, and the rest as second element.

///

/// ```aiken

/// list.partition([1, 2, 3, 4], fn(x) { x % 2 == 0 }) == ([2, 4], [1, 3])

/// ```

pub fn partition(self: List<a>, predicate: fn(a) -> Bool) -> (List<a>, List<a>) {

when self is {

[] -> ([], [])

[x, ..xs] -> {

let (left, right) = partition(xs, predicate)

if predicate(x) {

([x, ..left], right)

} else {

(left, [x, ..right])

}

}

}

}

test partition\_1() {

partition([], fn(x) { x > 2 }) == ([], [])

}

test partition\_2() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

partition(xs, fn(x) { x > 5 }) == ([10, 9, 8, 7, 6], [5, 4, 3, 2, 1])

}

test partition\_3() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

partition(xs, fn(x) { x == 42 }) == ([], xs)

}

test partition\_4() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

partition(xs, fn(x) { x < 42 }) == (xs, [])

}

test partition\_5() {

partition([1, 2, 3, 4], fn(x) { x % 2 == 0 }) == ([2, 4], [1, 3])

}

/// Extract a sublist from the given list using 0-based indexes. Negative

/// indexes wrap over, so `-1` refers to the last element of the list.

///

/// ```aiken

/// list.slice([1, 2, 3, 4, 5, 6], from: 2, to: 4) == [3, 4, 5]

/// list.slice([1, 2, 3, 4, 5, 6], from: -2, to: -1) == [5, 6]

/// list.slice([1, 2, 3, 4, 5, 6], from: 1, to: -1) == [2, 3, 4, 5, 6]

/// ```

pub fn slice(self: List<a>, from: Int, to: Int) {

let (i, l) =

if from >= 0 {

(from, None)

} else {

let l = length(self)

(l + from, Some(l))

}

let j =

if to >= 0 {

to - i + 1

} else {

when l is {

Some(l) -> l + to - i + 1

None -> length(self) + to - i + 1

}

}

self

|> drop(i)

|> take(j)

}

test slice\_1() {

slice([1, 2, 3], 0, 2) == [1, 2, 3]

}

test slice\_2() {

slice([1, 2, 3, 4, 5, 6], from: 2, to: 4) == [3, 4, 5]

}

test slice\_3() {

slice([1, 2, 3, 4, 5, 6], from: -2, to: -1) == [5, 6]

}

test slice\_4() {

slice([1, 2, 3, 4, 5, 6], from: 1, to: -1) == [2, 3, 4, 5, 6]

}

test slice\_5() {

slice([1, 2, 3, 4, 5, 6], from: -4, to: -3) == [3, 4]

}

test slice\_6() {

slice([1, 2, 3, 4, 5, 6], from: -2, to: 1) == []

}

/// Sort a list in ascending order using the given comparison function.

///

/// ```aiken

/// use aiken/int

///

/// sort([3, 1, 4, 0, 2], int.compare) == [0, 1, 2, 3, 4]

/// sort([1, 2, 3], int.compare) == [1, 2, 3]

/// ```

pub fn sort(self: List<a>, compare: fn(a, a) -> Ordering) -> List<a> {

when self is {

[] ->

[]

[x, ..xs] -> insert(sort(xs, compare), x, compare)

}

}

fn insert(self: List<a>, e: a, compare: fn(a, a) -> Ordering) -> List<a> {

when self is {

[] ->

[e]

[x, ..xs] ->

if compare(e, x) == Less {

[e, ..self]

} else {

[x, ..insert(xs, e, compare)]

}

}

}

test sort\_1() {

let xs =

[6, 7, 5, 4, 1, 3, 9, 8, 0, 2]

sort(xs, int.compare) == [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

}

test sort\_2() {

let xs =

[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

sort(xs, int.compare) == [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

}

test sort\_3() {

let xs =

[9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

sort(xs, int.compare) == [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

}

test sort\_4() {

sort([], int.compare) == []

}

/// Cut a list in two, such that the first list contains the given number of /

/// elements and the second list contains the rest.

///

/// Fundamentally equivalent to (but more efficient):

///

/// ```aiken

/// // span(xs, n) == (take(xs, n), drop(xs, n))

/// span([1, 2, 3, 4, 5], 3) == ([1, 2, 3], [4, 5])

/// ```

pub fn span(self: List<a>, n: Int) -> (List<a>, List<a>) {

when self is {

[] -> ([], [])

[x, ..xs] ->

if n <= 0 {

([], self)

} else {

let (left, right) = span(xs, n - 1)

([x, ..left], right)

}

}

}

test span\_1() {

span([], 2) == ([], [])

}

test span\_2() {

span([1, 2, 3], 2) == ([1, 2], [3])

}

test span\_3() {

span([1, 2, 3], -1) == ([], [1, 2, 3])

}

test span\_4() {

span([1, 2, 3], 42) == ([1, 2, 3], [])

}

/// Get elements of a list after the first one, if any.

///

/// ```aiken

/// list.tail([]) == None

/// list.tail([1, 2, 3]) == Some([2, 3])

/// ```

pub fn tail(self: List<a>) -> Option<List<a>> {

when self is {

[] -> None

[\_, ..xs] -> Some(xs)

}

}

test tail\_1() {

tail([1, 2, 3]) == Some([2, 3])

}

test tail\_2() {

tail([]) == None

}

/// Get the first `n` elements of a list.

///

/// ```aiken

/// list.take([1, 2, 3], 2) == [1, 2]

/// list.take([1, 2, 3], 14) == [1, 2, 3]

/// ```

pub fn take(self: List<a>, n: Int) -> List<a> {

if n <= 0 {

[]

} else {

when self is {

[] ->

[]

[x, ..xs] ->

[x, ..take(xs, n - 1)]

}

}

}

test take\_1() {

take([], 42) == []

}

test take\_2() {

take([1, 2, 3], 2) == [1, 2]

}

/// Returns the longest prefix of the given list where all elements satisfy the predicate.

///

/// ```aiken

/// list.take\_while([1, 2, 3], fn(x) { x > 2 }) == []

/// list.take\_while([1, 2, 3], fn(x) { x < 2 }) == [1]

/// ```

pub fn take\_while(self: List<a>, predicate: fn(a) -> Bool) -> List<a> {

when self is {

[] ->

[]

[x, ..xs] ->

if predicate(x) {

[x, ..take\_while(xs, predicate)]

} else {

[]

}

}

}

test take\_while\_1() {

take\_while([], fn(x) { x > 2 }) == []

}

test take\_while\_2() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

take\_while(xs, fn(x) { x > 5 }) == [10, 9, 8, 7, 6]

}

test take\_while\_3() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

take\_while(xs, fn(x) { x == 42 }) == []

}

test take\_while\_4() {

let xs =

[10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

take\_while(xs, fn(x) { x < 42 }) == xs

}

/// Removes duplicate elements from a list.

///

/// ```aiken

/// list.unique([1, 2, 3, 1]) == [1, 2, 3]

/// ```

pub fn unique(self: List<a>) -> List<a> {

when self is {

[] ->

[]

[x, ..xs] ->

[x, ..unique(filter(xs, fn(y) { y != x }))]

}

}

test unique\_1() {

unique([]) == []

}

test unique\_2() {

let xs =

[1, 2, 3, 1, 1, 3, 4, 1, 2, 3, 2, 4, 5, 6, 7, 8, 9, 10, 9]

unique(xs) == [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

}

/// Decompose a list of tuples into a tuple of lists.

///

/// ```

/// list.unzip([(1, "a"), (2, "b")]) == ([1, 2], ["a", "b"])

/// ```

pub fn unzip(self: List<(a, b)>) -> (List<a>, List<b>) {

when self is {

[] -> ([], [])

[(a, b), ..xs] -> {

let (a\_tail, b\_tail) = unzip(xs)

([a, ..a\_tail], [b, ..b\_tail])

}

}

}

test unzip\_1() {

unzip([]) == ([], [])

}

test unzip\_2() {

unzip([(1, "a"), (2, "b")]) == ([1, 2], ["a", "b"])

}

/// Combine two lists together.

///

/// Note: if one list is longer, the extra elements are dropped.

///

/// ```aiken

/// list.zip([1, 2], ["a", "b", "c"]) == [(1, "a"), (2, "b")]

/// ```

pub fn zip(self: List<a>, bs: List<b>) -> List<(a, b)> {

when self is {

[] ->

[]

[x, ..xs] ->

when bs is {

[] ->

[]

[y, ..ys] ->

[(x, y), ..zip(xs, ys)]

}

}

}

test zip\_1() {

zip([], [1, 2, 3]) == []

}

test zip\_2() {

zip([1, 2, 3], []) == []

}

test zip\_3() {

zip([1, 2], ["a", "b", "c"]) == [(1, "a"), (2, "b")]

}

/// Reduce a list from left to right using the accumulator as left operand.

/// Said differently, this is [`foldl`](#foldl) with callback arguments swapped.

///

/// ```aiken

/// list.reduce([#[1], #[2], #[3]], #[0], bytearray.concat) == #[0, 1, 2, 3]

/// list.reduce([True, False, True], False, fn(b, a) { or { b, a } }) == True

/// ```

pub fn reduce(self: List<a>, zero: b, with: fn(b, a) -> b) -> b {

foldl(self, zero, flip(with))

}

test reduce\_1() {

reduce([], 0, fn(n, total) { n + total }) == 0

}

test reduce\_2() {

reduce([1, 2, 3], 0, fn(n, total) { n + total }) == 6

}

test reduce\_3() {

reduce([True, False, True], False, fn(left, right) { left || right }) == True

}

test reduce\_4() {

reduce(

[#[1], #[2], #[3]],

#[9],

fn(left, right) { bytearray.concat(left, right) },

) == #[9, 1, 2, 3]

}

### 2.10. Math

//// This module contains some basic Math utilities. Standard arithmetic

//// operations on integers are available through native operators:

////

//// Operator | Description

//// --- | :---

//// `+` | Arithmetic sum

//// `-` | Arithmetic difference

//// `/` | Whole division

//// `\*` | Arithmetic multiplication

//// `%` | Remainder by whole division

////

//// Here are a few examples:

////

//// ```aiken

//// 1 + 1 // 2

//// 10 - 2 // 8

//// 40 / 14 // 2

//// 3 \* 4 // 12

//// 10 % 3 // 1

use aiken/builtin

/// Calculate the absolute value of an integer.

///

/// ```aiken

/// math.abs(-42) == 42

/// math.abs(14) == 14

/// ```

pub fn abs(self: Int) -> Int {

if self < 0 {

0 - self

} else {

self

}

}

test abs\_1() {

abs(14) == 14

}

test abs\_2() {

abs(-42) == 42

}

/// Restrict the value of an integer between two min and max bounds

///

/// ```aiken

/// math.clamp(14, min: 0, max: 10) == 10

/// ```

pub fn clamp(self: Int, min: Int, max: Int) -> Int {

if self < min {

min

} else {

if self > max {

max

} else {

self

}

}

}

test clamp\_1() {

clamp(14, min: 0, max: 10) == 10

}

test clamp\_2() {

clamp(7, min: 0, max: 10) == 7

}

test clamp\_3() {

clamp(7, min: 10, max: 100) == 10

}

/// Return the maximum of two integers.

pub fn max(a: Int, b: Int) -> Int {

if a > b {

a

} else {

b

}

}

test max\_1() {

max(0, 0) == 0

}

test max\_2() {

max(14, 42) == 42

}

test max\_3() {

max(42, 14) == 42

}

/// Return the minimum of two integers.

pub fn min(a: Int, b: Int) -> Int {

if a > b {

b

} else {

a

}

}

test min\_1() {

min(0, 0) == 0

}

test min\_2() {

min(14, 42) == 14

}

test min\_3() {

min(42, 14) == 14

}

/// Calculates a number to the power of `e` using the exponentiation by

/// squaring method.

///

/// ```aiken

/// math.pow(3, 5) == 243

/// math.pow(7, 2) == 49

/// math.pow(3, -4) == 0

/// math.pow(0, 0) == 1

/// math.pow(513, 3) == 135005697

/// ```

pub fn pow(self: Int, e: Int) -> Int {

if e < 0 {

0

} else if e == 0 {

1

} else if e % 2 == 0 {

pow(self \* self, e / 2)

} else {

self \* pow(self \* self, ( e - 1 ) / 2)

}

}

test pow\_3\_5() {

pow(3, 5) == 243

}

test pow\_7\_2() {

pow(7, 2) == 49

}

test pow\_3\_\_4() {

// negative powers round to zero

pow(3, -4) == 0

}

test pow\_0\_0() {

// sorry math

pow(0, 0) == 1

}

test pow\_513\_3() {

pow(513, 3) == 135005697

}

test pow\_2\_4() {

pow(2, 4) == 16

}

test pow\_2\_42() {

pow(2, 42) == 4398046511104

}

/// Calculates the power of 2 for a given exponent `e`. Much cheaper than

/// using `pow(2, \_)` for small exponents (0 < e < 256).

///

/// ```aiken

/// math.pow2(-2) == 0

/// math.pow2(0) == 1

/// math.pow2(1) == 2

/// math.pow2(4) == 16

/// math.pow2(42) == 4398046511104

/// ```

pub fn pow2(e: Int) -> Int {

// do\_pow2(e, 1)

if e < 8 {

if e < 0 {

0

} else {

builtin.index\_bytearray(#[1, 2, 4, 8, 16, 32, 64, 128], e)

}

} else if e < 32 {

256 \* pow2(e - 8)

} else {

4294967296 \* pow2(e - 32)

}

}

test pow2\_neg() {

pow2(-2) == 0

}

test pow2\_0() {

pow2(0) == 1

}

test pow2\_1() {

pow2(1) == 2

}

test pow2\_4() {

pow2(4) == 16

}

test pow2\_42() {

pow2(42) == 4398046511104

}

test pow2\_256() {

pow2(256) == 115792089237316195423570985008687907853269984665640564039457584007913129639936

}

/// The logarithm in base `b` of an element using integer divisions.

///

/// ```aiken

/// math.log(10, base: 2) == 3

/// math.log(42, base: 2) == 5

/// math.log(42, base: 3) == 3

/// math.log(5, base: 0) == 0

/// math.log(4, base: 4) == 1

/// math.log(4, base: 42) == 0

/// ```

pub fn log(self: Int, base: Int) -> Int {

if base <= 0 {

0

} else if self == base {

1

} else if self < base {

0

} else {

1 + log(self / base, base)

}

}

test log\_10\_2() {

log(10, base: 2) == 3

}

test log\_42\_2() {

log(42, base: 2) == 5

}

test log\_42\_3() {

log(42, base: 3) == 3

}

test log\_5\_0() {

log(5, base: 0) == 0

}

test log\_4\_4() {

log(4, base: 4) == 1

}

test log\_4\_43() {

log(4, base: 43) == 0

}

/// The greatest common divisor of two integers.

///

/// ```aiken

/// math.gcd(42, 14) == 14

/// math.gcd(14, 42) == 14

/// math.gcd(0, 0) == 0

/// ```

pub fn gcd(x: Int, y: Int) -> Int {

abs(do\_gcd(x, y))

}

fn do\_gcd(x: Int, y: Int) -> Int {

when y is {

0 -> x

\_ -> do\_gcd(y, x % y)

}

}

test gcd\_test1() {

gcd(10, 300) == 10

}

test gcd\_test2() {

gcd(-10, 300) == 10

}

test gcd\_test3() {

gcd(42, 14) == 14

}

/// Calculates the square root of an integer using the [Babylonian

/// method](https://en.wikipedia.org/wiki/Methods\_of\_computing\_square\_roots#Babylonian\_method). This returns either the exact result or the smallest integer

/// nearest to the square root.

///

/// Returns `None` for negative values.

///

/// ```aiken

/// math.sqrt(0) == Some(0)

/// math.sqrt(25) == Some(5)

/// math.sqrt(44203) == Some(210)

/// math.sqrt(-42) == None

/// ```

pub fn sqrt(self: Int) -> Option<Int> {

if self < 0 {

None

} else if self <= 1 {

Some(self)

} else {

Some(sqrt\_babylonian(self, self, ( self + 1 ) / 2))

}

}

// The basic idea is that if x is an overestimate to the square root of a

// non-negative real number S then S/x will be an underestimate, or vice versa,

// and so the average of these two numbers may reasonably be expected to provide a

// better approximation (though the formal proof of that assertion depends on the

// inequality of arithmetic and geometric means that shows this average is always

// an overestimate of the square root.

fn sqrt\_babylonian(self: Int, x: Int, y: Int) -> Int {

if y >= x {

x

} else {

sqrt\_babylonian(self, y, ( y + self / y ) / 2)

}

}

test sqrt1() {

sqrt(0) == Some(0)

}

test sqrt2() {

sqrt(1) == Some(1)

}

test sqrt3() {

sqrt(25) == Some(5)

}

test sqrt4() {

sqrt(44203) == Some(210)

}

test sqrt5() {

sqrt(975461057789971041) == Some(987654321)

}

test sqrt6() {

sqrt(-42) == None

}

/// Checks if an integer has a given integer square root x.

/// The check has constant time complexity (O(1)).

///

/// ```aiken

/// math.is\_sqrt(0, 0)

/// math.is\_sqrt(25, 5)

/// ! math.is\_sqrt(25, -5)

/// math.is\_sqrt(44203, 210)

/// ```

pub fn is\_sqrt(self: Int, x: Int) -> Bool {

x \* x <= self && ( x + 1 ) \* ( x + 1 ) > self

}

test is\_sqrt1() {

is\_sqrt(44203, 210)

}

test is\_sqrt2() {

is\_sqrt(975461057789971041, 987654321)

}

### 2.11 option

/// Picks the first element which is not None. If there's no such element, return None.

///

/// ```aiken

/// option.choice([]) == None

/// option.choice([Some(14), Some(42)]) == Some(14)

/// option.choice([None, Some(42)]) == Some(42)

/// option.choice([None, None]) == None

/// ```

pub fn choice(self: List<Option<a>>) -> Option<a> {

when self is {

[] -> None

[head, ..others] ->

when head is {

None -> choice(others)

\_ -> head

}

}

}

test choice\_1() {

Some(1) == choice([Some(1), Some(2)])

}

test choice\_2() {

None == choice([])

}

test choice\_3() {

Some(1) == choice([None, Some(1)])

}

/// Provide a default value, turning an optional value into a normal value.

///

/// ```aiken

/// option.or\_else(None, "aiken") == "aiken"

/// option.or\_else(Some(42), 14) == 42

/// ```

pub fn or\_else(self: Option<a>, default: a) -> a {

when self is {

None -> default

Some(a) -> a

}

}

test or\_else\_1() {

or\_else(None, "aiken") == "aiken"

}

test or\_else\_2() {

or\_else(Some(42), 14) == 42

}

/// Like [`or\_else`](#or\_else) but allows returning an `Option`.

/// This is effectively mapping the error branch.

///

/// ```aiken

/// option.or\_try(None, fn(\_) { Some("aiken") }) == Some("aiken")

/// option.or\_try(Some(42), fn(\_) { Some(14) }) == Some(42)

/// ```

pub fn or\_try(self: Option<a>, compute\_default: fn() -> Option<a>) -> Option<a> {

when self is {

None -> compute\_default()

\_ -> self

}

}

test or\_try\_1() {

or\_try(None, fn() { Some("aiken") }) == Some("aiken")

}

test or\_try\_2() {

or\_try(Some(42), fn() { fail }) == Some(42)

}

/// Apply a function to the inner value of an [`Option`](#option)

///

/// ```aiken

/// option.map(None, fn(n) { n \* 2 }) == None

/// option.map(Some(14), fn(n) { n \* 2 }) == Some(28)

/// ```

pub fn map(self: Option<a>, with: fn(a) -> result) -> Option<result> {

when self is {

None -> None

Some(a) -> Some(with(a))

}

}

test map\_1() {

map(None, fn(\_) { Void }) == None

}

test map\_2() {

map(Some(14), fn(n) { n + 1 }) == Some(15)

}

/// Combine two [`Option`](#option) together.

///

/// ```aiken

/// type Foo {

/// Foo(Int, Int)

/// }

///

/// option.map2(Some(14), Some(42), Foo) == Some(Foo(14, 42))

/// option.map2(None, Some(42), Foo) == None

/// option.map2(Some(14), None, Foo) == None

/// ```

pub fn map2(

opt\_a: Option<a>,

opt\_b: Option<b>,

with: fn(a, b) -> result,

) -> Option<result> {

when opt\_a is {

None -> None

Some(a) ->

when opt\_b is {

None -> None

Some(b) -> Some(with(a, b))

}

}

}

test map2\_1() {

map2(None, Some(42), fn(\_, \_) { 14 }) == None

}

test map2\_2() {

map2(Some(42), None, fn(\_, \_) { 14 }) == None

}

test map2\_3() {

map2(Some(14), Some(42), fn(a, b) { (a, b) }) == Some((14, 42))

}

/// Combine three [`Option`](#option) together.

///

/// ```aiken

/// type Foo {

/// Foo(Int, Int, Int)

/// }

///

/// option.map3(Some(14), Some(42), Some(1337), Foo) == Some(Foo(14, 42, 1337))

/// option.map3(None, Some(42), Some(1337), Foo) == None

/// option.map3(Some(14), None, None, Foo) == None

/// ```

pub fn map3(

opt\_a: Option<a>,

opt\_b: Option<b>,

opt\_c: Option<c>,

with: fn(a, b, c) -> result,

) -> Option<result> {

when opt\_a is {

None -> None

Some(a) ->

when opt\_b is {

None -> None

Some(b) ->

when opt\_c is {

None -> None

Some(c) -> Some(with(a, b, c))

}

}

}

}

test map3\_1() {

map3(None, Some(42), None, fn(\_, \_, \_) { 14 }) == None

}

test map3\_2() {

map3(Some(42), None, None, fn(\_, \_, \_) { 14 }) == None

}

test map3\_3() {

map3(Some(14), Some(42), Some(1337), fn(a, b, c) { c - a + b }) == Some(1365)

}

/// Chain together many computations that may fail.

///

/// ```aiken

/// self

/// |> dict.get(policy\_id)

/// |> option.and\_then(dict.get(\_, asset\_name))

/// |> option.or\_else(0)

/// ```

pub fn and\_then(

self: Option<a>,

then: fn(a) -> Option<result>,

) -> Option<result> {

when self is {

None -> None

Some(a) -> then(a)

}

}

fn try\_decrement(n: Int) -> Option<Int> {

if n > 0 {

Some(n - 1)

} else {

None

}

}

test and\_then\_1() {

let result =

None

|> and\_then(try\_decrement)

result == None

}

test and\_then\_2() {

let result =

Some(14)

|> and\_then(try\_decrement)

result == Some(13)

}

test and\_then\_3() {

let result =

Some(0)

|> and\_then(try\_decrement)

result == None

}

/// Converts from `Option<Option<a>>` to `Option<a>`.

///

/// ```aiken

/// option.flatten(Some(Some(42))) == Some(42)

/// option.flatten(Some(None)) == None

/// option.flatten(None) == None

/// ```

///

/// Flattening only removes one level of nesting at a time:

///

/// ```aiken

/// flatten(Some(Some(Some(42)))) == Some(Some(42))

/// Some(Some(Some(42))) |> flatten |> flatten == Some(42)

/// ```

pub fn flatten(opt: Option<Option<a>>) -> Option<a> {

when opt is {

Some(inner) -> inner

None -> None

}

}

test flatten\_1() {

let x: Option<Option<Int>> = Some(Some(6))

Some(6) == flatten(x)

}

test flatten\_2() {

let x: Option<Option<Int>> = Some(None)

None == flatten(x)

}

test flatten\_3() {

let x: Option<Option<Int>> = None

None == flatten(x)

}

test flatten\_4() {

let x: Option<Option<Option<Int>>> = Some(Some(Some(6)))

let result =

x

|> flatten

|> flatten

Some(6) == result

}

/// Asserts whether an option is `Some`, irrespective of the value it contains.

pub fn is\_some(self: Option<a>) -> Bool {

when self is {

Some(\_) -> True

\_ -> False

}

}

test is\_some\_1() {

is\_some(Some(0)) == True

}

test is\_some\_2() {

is\_some(None) == False

}

/// Asserts whether an option is `None`.

pub fn is\_none(self: Option<a>) -> Bool {

when self is {

Some(\_) -> False

\_ -> True

}

}

test is\_none\_1() {

is\_none(Some(0)) == False

}

test is\_none\_2() {

is\_none(None) == True

}

### 2.12 pairs

//// A module for working with associative lists (a.k.a `Pairs`).

////

//// While any function that works on `List` also work on `Pairs`, this module provides some extra helpers

//// that are specifically tailored to working with associative lists. Fundamentally, a `Pairs<k, v>` is

//// a type-alias to `List<Pair<k, v>>`.

////

//// ### Important

////

//// Unlike dictionnaries (a.k.a. `Dict`), associative lists make no assumption

//// about the ordering of elements within the list. As a result, lookup

//// functions do traverse the entire list when invoked. They are also not \_sets\_,

//// and thus allow for duplicate keys. This is reflected in the functions used

//// to interact with them.

/// Remove a single key-value pair from the `Pairs`. If the key is not found, no changes are made.

/// Duplicate keys are not removed. Only the \*\*first\*\* key found is removed.

///

/// ```aiken

/// pairs.remove\_first([], "a") == []

/// pairs.remove\_first([Pair("a", 1)], "a") == []

/// pairs.remove\_first([Pair("a", 1), Pair("b", 2)], "a") == [Pair("b", 2)]

/// pairs.remove\_first([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == [Pair("b", 2), Pair("a", 3)]

/// ```

pub fn remove\_first(self: Pairs<key, value>, key k: key) -> Pairs<key, value> {

when self is {

[] ->

[]

[Pair(k2, v2), ..rest] ->

if k == k2 {

rest

} else {

[Pair(k2, v2), ..remove\_first(rest, k)]

}

}

}

test remove\_first\_1() {

remove\_first([], "a") == []

}

test remove\_first\_2() {

remove\_first([Pair("a", 14)], "a") == []

}

test remove\_first\_3() {

let fixture =

[Pair("a", 14)]

remove\_first(fixture, "b") == fixture

}

test remove\_first\_4() {

let fixture =

[Pair("a", 1), Pair("b", 2), Pair("a", 3)]

remove\_first(fixture, "a") == [Pair("b", 2), Pair("a", 3)]

}

/// Remove a single key-value pair from the Pairs. If the key is not found, no changes are made.

/// Duplicate keys are not removed. Only the \*\*last\*\* key found is removed.

///

/// ```aiken

/// pairs.remove\_last([], "a") == []

/// pairs.remove\_last([Pair("a", 1)], "a") == []

/// pairs.remove\_last([Pair("a", 1), Pair("b", 2)], "a") == [Pair("b", 2)]

/// pairs.remove\_last([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == [Pair("a", 1), Pair("b", 2)]

/// ```

pub fn remove\_last(self: Pairs<key, value>, key k: key) -> Pairs<key, value> {

when self is {

[] ->

[]

[Pair(k2, v2), ..rest] ->

if k == k2 {

let tail = remove\_last(rest, k)

if tail == rest {

rest

} else {

[Pair(k2, v2), ..tail]

}

} else {

[Pair(k2, v2), ..remove\_last(rest, k)]

}

}

}

test remove\_last\_1() {

remove\_last([], "a") == []

}

test remove\_last\_2() {

remove\_last([Pair("a", 14)], "a") == []

}

test remove\_last\_3() {

let fixture =

[Pair("a", 14)]

remove\_last(fixture, "b") == fixture

}

test remove\_last\_4() {

let fixture =

[Pair("a", 1), Pair("b", 2), Pair("a", 3)]

remove\_last(fixture, "a") == [Pair("a", 1), Pair("b", 2)]

}

/// Remove all key-value pairs matching the key from the Pairs. If the key is not found, no changes are made.

///

/// ```aiken

/// pairs.remove\_all([], "a") == []

/// pairs.remove\_all([Pair("a", 1)], "a") == []

/// pairs.remove\_all([Pair("a", 1), Pair("b", 2)], "a") == [Pair("b", 2)]

/// pairs.remove\_all([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == [Pair("b", 2)]

/// ```

pub fn remove\_all(self: Pairs<key, value>, key k: key) -> Pairs<key, value> {

when self is {

[] ->

[]

[Pair(k2, v2), ..rest] ->

if k == k2 {

remove\_all(rest, k)

} else {

[Pair(k2, v2), ..remove\_all(rest, k)]

}

}

}

test remove\_all\_1() {

remove\_all([], "a") == []

}

test remove\_all\_2() {

remove\_all([Pair("a", 14)], "a") == []

}

test remove\_all\_3() {

let fixture =

[Pair("a", 14)]

remove\_all(fixture, "b") == fixture

}

test remove\_all\_4() {

let fixture =

[Pair("a", 1), Pair("b", 2), Pair("a", 3)]

remove\_all(fixture, "a") == [Pair("b", 2)]

}

/// Finds the first key in the alist associated with a given value, if any.

///

/// ```aiken

/// pairs.find\_first([], 1) == None

/// pairs.find\_first([Pair("a", 1)], 1) == Some("a")

/// pairs.find\_first([Pair("a", 1), Pair("b", 2)], 1) == Some("a")

/// pairs.find\_first([Pair("a", 1), Pair("b", 2), Pair("c", 1)], 1) == Some("a")

/// ```

pub fn find\_first(self: Pairs<key, value>, v: value) -> Option<key> {

when self is {

[] -> None

[Pair(k2, v2), ..rest] ->

if v == v2 {

Some(k2)

} else {

find\_first(rest, v)

}

}

}

test find\_first\_1() {

find\_first([], "a") == None

}

test find\_first\_2() {

find\_first([Pair("a", 14)], 14) == Some("a")

}

test find\_first\_3() {

find\_first([Pair("a", 14)], 42) == None

}

test find\_first\_4() {

find\_first([Pair("a", 14), Pair("b", 42), Pair("c", 14)], 14) == Some("a")

}

/// Finds the last key in the alist associated with a given value, if any.

///

/// ```aiken

/// pairs.find\_last([], 1) == None

/// pairs.find\_last([Pair("a", 1)], 1) == Some("a")

/// pairs.find\_last([Pair("a", 1), Pair("b", 2)], 1) == Some("a")

/// pairs.find\_last([Pair("a", 1), Pair("b", 2), Pair("c", 1)], 1) == Some("c")

/// ```

pub fn find\_last(self: Pairs<key, value>, v: value) -> Option<key> {

when self is {

[] -> None

[Pair(k2, v2), ..rest] ->

if v == v2 {

when find\_last(rest, v) is {

None -> Some(k2)

some -> some

}

} else {

find\_last(rest, v)

}

}

}

test find\_last\_1() {

find\_last([], "a") == None

}

test find\_last\_2() {

find\_last([Pair("a", 14)], 14) == Some("a")

}

test find\_last\_3() {

find\_last([Pair("a", 14)], 42) == None

}

test find\_last\_4() {

find\_last([Pair("a", 14), Pair("b", 42), Pair("c", 14)], 14) == Some("c")

}

/// Finds all keys in the alist associated with a given value.

///

/// ```aiken

/// pairs.find\_all([], 1) == []

/// pairs.find\_all([Pair("a", 1)], 1) == ["a"]

/// pairs.find\_all([Pair("a", 1), Pair("b", 2)], 1) == ["a"]

/// pairs.find\_all([Pair("a", 1), Pair("b", 2), Pair("c", 1)], 1) == ["a", "c"]

/// ```

pub fn find\_all(self: Pairs<key, value>, v: value) -> List<key> {

when self is {

[] ->

[]

[Pair(k2, v2), ..rest] ->

if v == v2 {

[k2, ..find\_all(rest, v)]

} else {

find\_all(rest, v)

}

}

}

test find\_all\_1() {

find\_all([], "a") == []

}

test find\_all\_2() {

find\_all([Pair("a", 14)], 14) == ["a"]

}

test find\_all\_3() {

find\_all([Pair("a", 14)], 42) == []

}

test find\_all\_4() {

find\_all([Pair("a", 14), Pair("b", 42), Pair("c", 14)], 14) == ["a", "c"]

}

/// Fold over the key-value pairs in a Pairs. The fold direction follows the

/// order of elements in the Pairs and is done from right-to-left.

///

/// ```aiken

/// let fixture = [

/// Pair(1, 100),

/// Pair(2, 200),

/// Pair(3, 300),

/// ]

///

/// pairs.foldr(fixture, 0, fn(k, v, result) { k \* v + result }) == 1400

/// ```

pub fn foldr(

self: Pairs<key, value>,

zero: result,

with: fn(key, value, result) -> result,

) -> result {

when self is {

[] -> zero

[Pair(k, v), ..rest] -> with(k, v, foldr(rest, zero, with))

}

}

test foldr\_1() {

foldr([], 14, fn(\_, \_, \_) { 42 }) == 14

}

test foldr\_2() {

foldr(

[Pair("a", 42), Pair("b", 14)],

zero: 0,

with: fn(\_, v, total) { v + total },

) == 56

}

test foldr\_3() {

let fixture =

[Pair(1, 100), Pair(2, 200), Pair(3, 300)]

foldr(fixture, 0, fn(k, v, result) { k \* v + result }) == 1400

}

/// Fold over the key-value pairs in a pairs. The fold direction follows keys

/// in ascending order and is done from left-to-right.

///

/// ```aiken

/// let fixture = [

/// Pair(1, 100),

/// Pair(2, 200),

/// Pair(3, 300),

/// ]

///

/// pairs.foldl(fixture, 0, fn(k, v, result) { k \* v + result }) == 1400

/// ```

pub fn foldl(

self: Pairs<key, value>,

zero: result,

with: fn(key, value, result) -> result,

) -> result {

when self is {

[] -> zero

[Pair(k, v), ..rest] -> foldl(rest, with(k, v, zero), with)

}

}

test foldl\_1() {

foldl([], 14, fn(\_, \_, \_) { 42 }) == 14

}

test foldl\_2() {

foldl(

[Pair("a", 42), Pair("b", 14)],

zero: 0,

with: fn(\_, v, total) { v + total },

) == 56

}

/// Get the value in the alist by its key.

/// If multiple values with the same key exist, only the first one is returned.

///

/// ```aiken

/// pairs.get\_first([], "a") == None

/// pairs.get\_first([Pair("a", 1)], "a") == Some(1)

/// pairs.get\_first([Pair("a", 1), Pair("b", 2)], "a") == Some(1)

/// pairs.get\_first([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == Some(1)

/// ```

pub fn get\_first(self: Pairs<key, value>, key k: key) -> Option<value> {

when self is {

[] -> None

[Pair(k2, v), ..rest] ->

if k == k2 {

Some(v)

} else {

get\_first(rest, k)

}

}

}

test get\_first\_1() {

get\_first([], "a") == None

}

test get\_first\_2() {

get\_first([Pair("a", 1)], "a") == Some(1)

}

test get\_first\_3() {

get\_first([Pair("a", 1), Pair("b", 2)], "a") == Some(1)

}

test get\_first\_4() {

get\_first([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == Some(1)

}

test get\_first\_5() {

get\_first([Pair("a", 1), Pair("b", 2), Pair("c", 3)], "d") == None

}

/// Get the value in the alist by its key.

/// If multiple values with the same key exist, only the last one is returned.

///

/// ```aiken

/// pairs.get\_last([], "a") == None

/// pairs.get\_last([Pair("a", 1)], "a") == Some(1)

/// pairs.get\_last([Pair("a", 1), Pair("b", 2)], "a") == Some(1)

/// pairs.get\_last([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == Some(3)

/// ```

pub fn get\_last(self: Pairs<key, value>, key k: key) -> Option<value> {

when self is {

[] -> None

[Pair(k2, v), ..rest] ->

if k == k2 {

when get\_last(rest, k) is {

None -> Some(v)

some -> some

}

} else {

get\_last(rest, k)

}

}

}

test get\_last\_1() {

get\_last([], "a") == None

}

test get\_last\_2() {

get\_last([Pair("a", 1)], "a") == Some(1)

}

test get\_last\_3() {

get\_last([Pair("a", 1), Pair("b", 2)], "a") == Some(1)

}

test get\_last\_4() {

get\_last([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == Some(3)

}

test get\_last\_5() {

get\_last([Pair("a", 1), Pair("b", 2), Pair("c", 3)], "d") == None

}

/// Get all values in the alist associated with a given key.

///

/// ```aiken

/// pairs.get\_all([], "a") == []

/// pairs.get\_all([Pair("a", 1)], "a") == [1]

/// pairs.get\_all([Pair("a", 1), Pair("b", 2)], "a") == [1]

/// pairs.get\_all([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == [1, 3]

/// ```

pub fn get\_all(self: Pairs<key, value>, key k: key) -> List<value> {

when self is {

[] ->

[]

[Pair(k2, v), ..rest] ->

if k == k2 {

[v, ..get\_all(rest, k)]

} else {

get\_all(rest, k)

}

}

}

test get\_all\_1() {

get\_all([], "a") == []

}

test get\_all\_2() {

get\_all([Pair("a", 1)], "a") == [1]

}

test get\_all\_3() {

get\_all([Pair("a", 1), Pair("b", 2)], "a") == [1]

}

test get\_all\_4() {

get\_all([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == [1, 3]

}

test get\_all\_5() {

get\_all([Pair("a", 1), Pair("b", 2), Pair("c", 3)], "d") == []

}

/// Check if a key exists in the pairs.

///

/// ```aiken

/// pairs.has\_key([], "a") == False

/// pairs.has\_key([Pair("a", 1)], "a") == True

/// pairs.has\_key([Pair("a", 1), Pair("b", 2)], "a") == True

/// pairs.has\_key([Pair("a", 1), Pair("b", 2), Pair("a", 3)], "a") == True

/// ```

pub fn has\_key(self: Pairs<key, value>, k: key) -> Bool {

when self is {

[] -> False

// || is lazy so this is fine

[Pair(k2, \_), ..rest] -> k == k2 || has\_key(rest, k)

}

}

test has\_key\_1() {

!has\_key([], "a")

}

test has\_key\_2() {

has\_key([Pair("a", 14)], "a")

}

test has\_key\_3() {

!has\_key([Pair("a", 14)], "b")

}

test has\_key\_4() {

has\_key([Pair("a", 14), Pair("b", 42)], "b")

}

test has\_key\_5() {

has\_key([Pair("a", 14), Pair("b", 42), Pair("a", 42)], "a")

}

/// Extract all the keys present in a given `Pairs`.

///

/// ```aiken

/// pairs.keys([]) == []

/// pairs.keys([Pair("a", 1)]) == ["a"]

/// pairs.keys([Pair("a", 1), Pair("b", 2)]) == ["a", "b"]

/// pairs.keys([Pair("a", 1), Pair("b", 2), Pair("a", 3)]) == ["a", "b", "a"]

/// ```

pub fn keys(self: Pairs<key, value>) -> List<key> {

when self is {

[] ->

[]

[Pair(k, \_), ..rest] ->

[k, ..keys(rest)]

}

}

test keys\_1() {

keys([]) == []

}

test keys\_2() {

keys([Pair("a", 0)]) == ["a"]

}

test keys\_3() {

keys([Pair("a", 0), Pair("b", 0)]) == ["a", "b"]

}

/// Apply a function to all key-value pairs in a alist, replacing the values.

///

/// ```aiken

/// let fixture = [Pair("a", 100), Pair("b", 200)]

///

/// pairs.map(fixture, fn(\_k, v) { v \* 2 }) == [Pair("a", 200), Pair("b", 400)]

/// ```

pub fn map(

self: Pairs<key, value>,

with: fn(key, value) -> result,

) -> Pairs<key, result> {

when self is {

[] ->

[]

[Pair(k, v), ..rest] ->

[Pair(k, with(k, v)), ..map(rest, with)]

}

}

test map\_1() {

let fixture =

[Pair("a", 1), Pair("b", 2)]

map(fixture, with: fn(k, \_) { k }) == [Pair("a", "a"), Pair("b", "b")]

}

test map\_2() {

let fixture =

[Pair("a", 1), Pair("b", 2)]

map(fixture, with: fn(\_, v) { v + 1 }) == [Pair("a", 2), Pair("b", 3)]

}

/// Extract all the values present in a given `Pairs`.

///

/// ```aiken

/// pairs.values([]) == []

/// pairs.values([Pair("a", 1)]) == [1]

/// pairs.values([Pair("a", 1), Pair("b", 2)]) == [1, 2]

/// pairs.values([Pair("a", 1), Pair("b", 2), Pair("a", 3)]) == [1, 2, 3]

/// ```

pub fn values(self: Pairs<key, value>) -> List<value> {

when self is {

[] ->

[]

[Pair(\_, v), ..rest] ->

[v, ..values(rest)]

}

}

test values\_1() {

values([]) == []

}

test values\_2() {

values([Pair("a", 1)]) == [1]

}

test values\_3() {

values([Pair("a", 1), Pair("b", 2)]) == [1, 2]

}

test values\_4() {

values([Pair("a", 1), Pair("b", 2), Pair("a", 3)]) == [1, 2, 3]

}

### 2.13 string

use aiken/builtin.{

append\_bytearray, append\_string, decode\_utf8, encode\_utf8, length\_of\_bytearray,

}

use aiken/cbor

/// Combine two `String` together.

///

/// ```aiken

/// string.concat(left: @"Hello", right: @", World!") == @"Hello, World!"

/// ```

pub fn concat(left: String, right: String) -> String {

append\_string(left, right)

}

test concat\_1() {

concat(@"", @"") == @""

}

test concat\_2() {

concat(@"", @"foo") == concat(@"foo", @"")

}

test concat\_3() {

concat(left: @"Hello", right: @", World!") == @"Hello, World!"

}

/// Convert a `ByteArray` into a `String`

///

/// <br/>⚠️<pre>WARNING</pre> | This functions fails if the underlying `ByteArray` isn't UTF-8-encoded. <br/>In particular, you cannot convert arbitrary hash digests using this function. <br/>For converting arbitrary `ByteArray`s, use [bytearray.to\_hex](/stdlib/aiken/bytearray.html#to\_hex).

/// --- | ---

///

/// ```aiken

/// string.from\_bytearray("foo") == @"foo"

///

/// string.from\_bytearray(#"666f6f") == @"foo"

///

/// string.from\_bytearray(some\_hash) -> fail

/// ```

pub fn from\_bytearray(bytes: ByteArray) -> String {

decode\_utf8(bytes)

}

test from\_bytearray\_1() {

from\_bytearray(#[]) == @""

}

test from\_bytearray\_2() {

from\_bytearray(#[65, 66, 67]) == @"ABC"

}

test from\_bytearray\_3() {

from\_bytearray("ABC") == @"ABC"

}

/// Convert an `Int` to its `String` representation.

///

/// ```aiken

/// string.from\_int(42) == @"42"

/// ```

pub fn from\_int(n: Int) -> String {

cbor.diagnostic(n)

}

test from\_int\_1() {

from\_int(0) == @"0"

}

test from\_int\_2() {

from\_int(5) == @"5"

}

test from\_int\_3() {

from\_int(42) == @"42"

}

test from\_int\_4() {

from\_int(200) == @"200"

}

/// Join a list of strings, separated by a given \_delimiter\_.

///

/// ```aiken

/// string.join([], @"+") == @""

/// string.join([@"a", @"b", @"c"], @",") == @"a,b,c"

/// ```

pub fn join(list: List<String>, delimiter: String) -> String {

do\_join(list, encode\_utf8(delimiter), #"")

|> decode\_utf8

}

fn do\_join(xs, delimiter, bytes) {

when xs is {

[] -> bytes

[x, ..rest] ->

do\_join(

rest,

delimiter,

if length\_of\_bytearray(bytes) == 0 {

encode\_utf8(x)

} else {

append\_bytearray(bytes, append\_bytearray(delimiter, encode\_utf8(x)))

},

)

}

}

test join\_1() {

join([], @",") == @""

}

test join\_2() {

join([@"a", @"b", @"c"], @",") == @"a,b,c"

}

/// Convert a `String` into a `ByteArray`

///

/// ```aiken

/// string.to\_bytearray(@"foo") == "foo"

/// ```

pub fn to\_bytearray(self: String) -> ByteArray {

encode\_utf8(self)

}

test to\_bytearray\_1() {

to\_bytearray(@"") == ""

}

test to\_bytearray\_2() {

to\_bytearray(@"ABC") == #[65, 66, 67]

}

test to\_bytearray\_3() {

to\_bytearray(@"ABC") == "ABC"

}

### 2.14 time

/// A number of milliseconds since 00:00:00 UTC on 1 January 1970.

pub type PosixTime =

Int

### 2.15 transaction

use aiken/builtin

use aiken/dict.{Dict}

use aiken/hash.{Blake2b\_224, Blake2b\_256, Hash, blake2b\_256}

use aiken/interval.{Interval}

use aiken/list

use aiken/option

use aiken/time.{PosixTime}

use aiken/transaction/certificate.{Certificate}

use aiken/transaction/credential.{

Address, Script, ScriptCredential, StakeCredential, VerificationKey,

VerificationKeyCredential,

}

use aiken/transaction/value.{MintedValue, PolicyId, Value}

/// A context given to a script by the Cardano ledger when being executed.

///

/// The context contains information about the entire transaction that contains

/// the script. The transaction may also contain other scripts; to distinguish

/// between multiple scripts, the `ScriptContext` also contains a `purpose`

/// which indicates which script (or, for what purpose) of the transaction is

/// being executed.

pub type ScriptContext {

transaction: Transaction,

purpose: ScriptPurpose,

}

/// Characterizes the kind of script being executed.

pub type ScriptPurpose {

/// For scripts executed as minting/burning policies, to insert

/// or remove assets from circulation. It's parameterized by the identifier

/// of the associated policy.

Mint(PolicyId)

/// For scripts that are used as payment credentials for addresses in

/// transaction outputs. They govern the rule by which the output they

/// reference can be spent.

Spend(OutputReference)

/// For scripts that validate reward withdrawals from a reward account.

///

/// The argument identifies the target reward account.

WithdrawFrom(StakeCredential)

/// Needed when delegating to a pool using stake credentials defined as a

/// Plutus script. This purpose is also triggered when de-registering such

/// stake credentials.

///

/// It embeds the certificate that's being validated.

Publish(Certificate)

}

/// A Cardano `Transaction`, as seen by Plutus scripts.

///

/// Note that this is a representation of a transaction, and not the 1:1

/// translation of the transaction as seen by the ledger. In particular,

/// Plutus scripts can't see inputs locked by bootstrap addresses, outputs

/// to bootstrap addresses or just transaction metadata.

pub type Transaction {

inputs: List<Input>,

reference\_inputs: List<Input>,

outputs: List<Output>,

fee: Value,

mint: MintedValue,

certificates: List<Certificate>,

withdrawals: Pairs<StakeCredential, Int>,

validity\_range: ValidityRange,

extra\_signatories: List<Hash<Blake2b\_224, VerificationKey>>,

redeemers: Pairs<ScriptPurpose, Redeemer>,

datums: Dict<Hash<Blake2b\_256, Data>, Data>,

id: TransactionId,

}

/// A placeholder / empty `Transaction` to serve as a base in a transaction

/// builder. This is particularly useful for constructing test transactions.

///

/// Every field is empty or null, and we have in particular:

///

/// ```aiken

/// use aiken/transaction

///

/// transaction.placeholder().id == TransactionId {

/// hash: #"0000000000000000000000000000000000000000000000000000000000000000",

/// }

///

/// transaction.placeholder().validity\_range == interval.everything()

/// ```

pub fn placeholder() -> Transaction {

Transaction {

inputs: [],

reference\_inputs: [],

outputs: [],

fee: value.zero(),

mint: value.zero() |> value.to\_minted\_value(),

certificates: [],

withdrawals: [],

validity\_range: interval.everything(),

extra\_signatories: [],

redeemers: [],

datums: dict.new(),

id: TransactionId {

hash: #"0000000000000000000000000000000000000000000000000000000000000000",

},

}

}

/// An interval of POSIX time, measured in number milliseconds since 1970-01-01T00:00:00Z.

pub type ValidityRange =

Interval<PosixTime>

/// A unique transaction identifier, as the hash of a transaction body. Note that the transaction id

/// isn't a direct hash of the `Transaction` as visible on-chain. Rather, they correspond to hash

/// digests of transaction body as they are serialized on the network.

pub type TransactionId {

hash: Hash<Blake2b\_256, Transaction>,

}

/// An `Input` made of an output reference and, the resolved value associated with that output.

pub type Input {

output\_reference: OutputReference,

output: Output,

}

/// An `OutputReference` is a unique reference to an output on-chain. The `output\_index`

/// corresponds to the position in the output list of the transaction (identified by its id)

/// that produced that output

pub type OutputReference {

transaction\_id: TransactionId,

output\_index: Int,

}

/// A transaction `Output`, with an address, a value and optional datums and script references.

pub type Output {

address: Address,

value: Value,

datum: Datum,

reference\_script: Option<Hash<Blake2b\_224, Script>>,

}

/// An output `Datum`.

pub type Datum {

NoDatum

/// A datum referenced by its hash digest.

DatumHash(Hash<Blake2b\_256, Data>)

/// A datum completely inlined in the output.

InlineDatum(Data)

}

/// A type-alias for Redeemers, passed to scripts for validation. The `Data` is

/// opaque because it is user-defined and it is the script's responsibility to

/// parse it into its expected form.

pub type Redeemer =

Data

/// Find an input by its [`OutputReference`](#OutputReference). This is typically used in

/// combination with the `Spend` [`ScriptPurpose`](#ScriptPurpose) to find a script's own

/// input.

///

/// ```aiken

/// validator {

/// fn(datum, redeemer, ctx: ScriptContext) {

/// expect Spend(my\_output\_reference) =

/// ctx.purpose

///

/// expect Some(input) =

/// ctx.transaction.inputs

/// |> transaction.find\_input(my\_output\_reference)

/// }

/// }

/// ```

pub fn find\_input(

inputs: List<Input>,

output\_reference: OutputReference,

) -> Option<Input> {

inputs

|> list.find(fn(input) { input.output\_reference == output\_reference })

}

/// Find a [`Datum`](#Datum) by its hash, if present. The function looks first for

/// datums in the witness set, and then for inline datums if it doesn't find any in

/// witnesses.

pub fn find\_datum(

outputs: List<Output>,

datums: Dict<Hash<Blake2b\_256, Data>, Data>,

datum\_hash: Hash<Blake2b\_256, Data>,

) -> Option<Data> {

datums

|> dict.get(datum\_hash)

|> option.or\_try(

fn() {

outputs

|> list.filter\_map(

fn(output) {

when output.datum is {

InlineDatum(data) ->

if

blake2b\_256(builtin.serialise\_data(data)) == datum\_hash{

Some(data)

} else {

None

}

\_ -> None

}

},

)

|> list.head

},

)

}

/// Find all outputs that are paying into the given script hash, if any. This is useful for

/// contracts running over multiple transactions.

pub fn find\_script\_outputs(

outputs: List<Output>,

script\_hash: Hash<Blake2b\_224, Script>,

) -> List<Output> {

outputs

|> list.filter(

fn(output) {

when output.address.payment\_credential is {

ScriptCredential(addr\_script\_hash) ->

script\_hash == addr\_script\_hash

VerificationKeyCredential(\_) -> False

}

},

)

}

### 3. Test Case Name: Aiken-Trie

graph TD

ROOT((Root)) -->|c| C(C)

C -->|a| A(A)

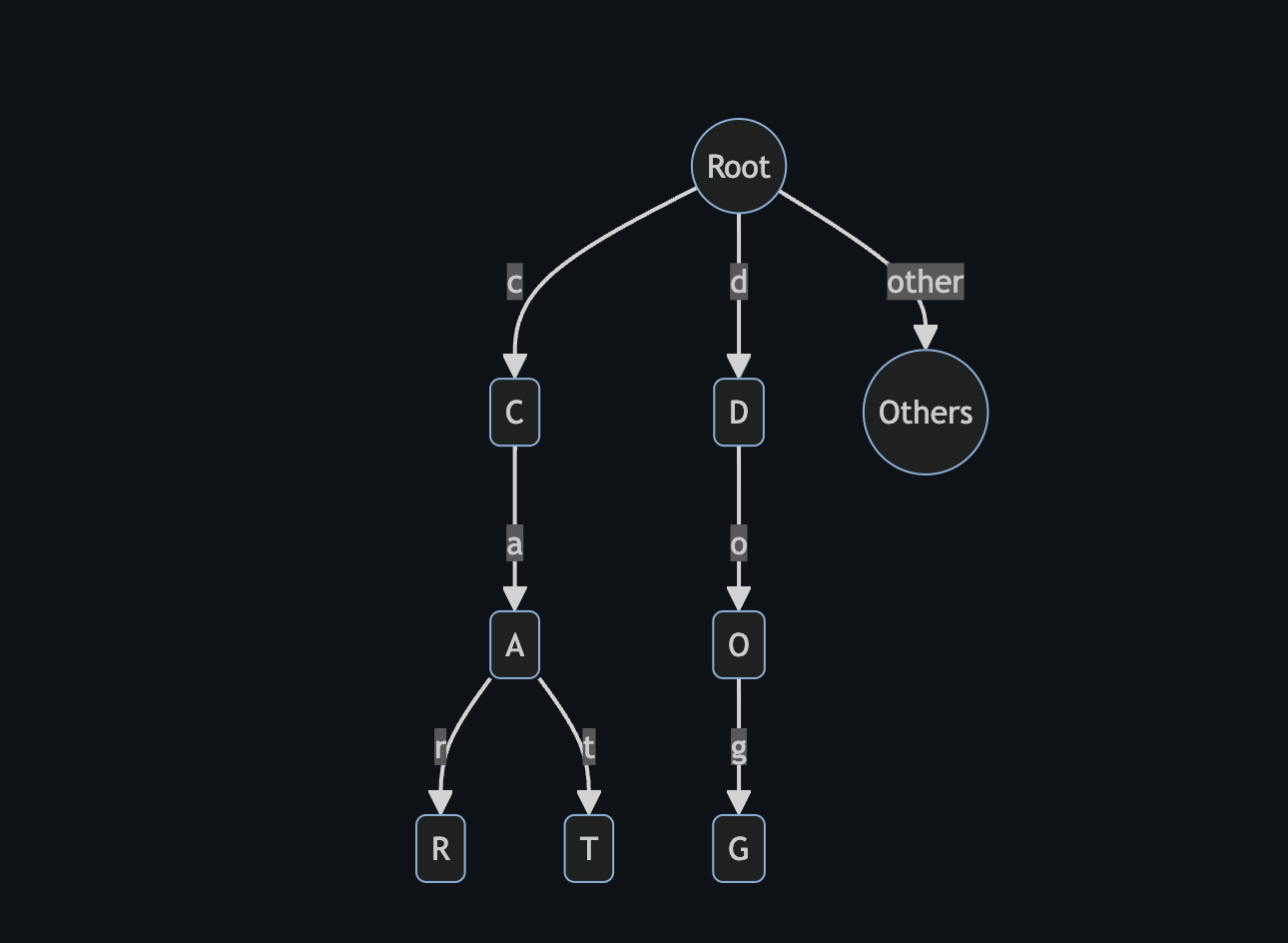
A -->|r| R(R)

A -->|t| T(T)

ROOT -->|d| D(D)

D -->|o| O(O)

O -->|g| G(G)

ROOT -->|other| OTH((Others))

#### **Test Result:**

import { Emulator, Lucid } from "lucid-cardano";

import { defaultProtocolParams } from "./constants";

import { generateAccount } from "./utils";

import { trieScript } from "../src/const";

import {

appendTrie,

betweenTrie,

createTrie,

getTrieOrigin,

getUtxoByKey,

} from "../src";

console.log(`Trie Script Size: ${trieScript.script.length / 2}`);

describe("Synthetics", () => {

let emulator: Emulator;

let lucid: Lucid;

let trieAddress: string;

let trieRewardAddress: string;

beforeEach(async () => {

emulator = new Emulator([]);

lucid = await Lucid.new(emulator);

trieAddress = lucid.utils.validatorToAddress(trieScript);

trieRewardAddress = lucid.utils.validatorToRewardAddress(trieScript);

const TRIE\_USER = await generateAccount({

lovelace: BigInt(100000000000000000),

});

emulator = new Emulator([TRIE\_USER], defaultProtocolParams);

emulator.chain[trieRewardAddress] = {

registeredStake: true,

delegation: { rewards: BigInt(0), poolId: "" },

};

lucid = await Lucid.new(emulator);

lucid.selectWalletFromPrivateKey(TRIE\_USER.privateKey);

});

it("Should pass e2e on creation, two insertions", async () => {

let trie = await createTrie(lucid, trieAddress, trieRewardAddress);

emulator.awaitBlock(1);

let trieOrigin = await getTrieOrigin(lucid, trie.trieUnit, trieAddress);

let trieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await appendTrie(

lucid,

trie.trieUnit,

trieOrigin!,

trieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

);

emulator.awaitBlock(1);

let newTrieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await betweenTrie(

lucid,

trie.trieUnit,

trieOrigin!,

newTrieUtxo!,

"hello",

trieAddress,

trieRewardAddress,

);

});

it("Should fail on duplicate insertion (1)", async () => {

let trie = await createTrie(lucid, trieAddress, trieRewardAddress);

emulator.awaitBlock(1);

let trieOrigin = await getTrieOrigin(lucid, trie.trieUnit, trieAddress);

let trieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await appendTrie(

lucid,

trie.trieUnit,

trieOrigin!,

trieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

);

emulator.awaitBlock(1);

let newTrieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

betweenTrie(

lucid,

trie.trieUnit,

trieOrigin!,

newTrieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

)

.then(() => {

throw "This should have failed";

})

.catch(() => {

// failure successful

});

});

it("Should fail on duplicate insertion (2)", async () => {

let trie = await createTrie(lucid, trieAddress, trieRewardAddress);

emulator.awaitBlock(1);

let trieOrigin = await getTrieOrigin(lucid, trie.trieUnit, trieAddress);

let trieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await appendTrie(

lucid,

trie.trieUnit,

trieOrigin!,

trieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

);

emulator.awaitBlock(1);

let newTrieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

betweenTrie(

lucid,

trie.trieUnit,

trieOrigin!,

newTrieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

)

.then(() => {

throw "This should have failed";

})

.catch(() => {

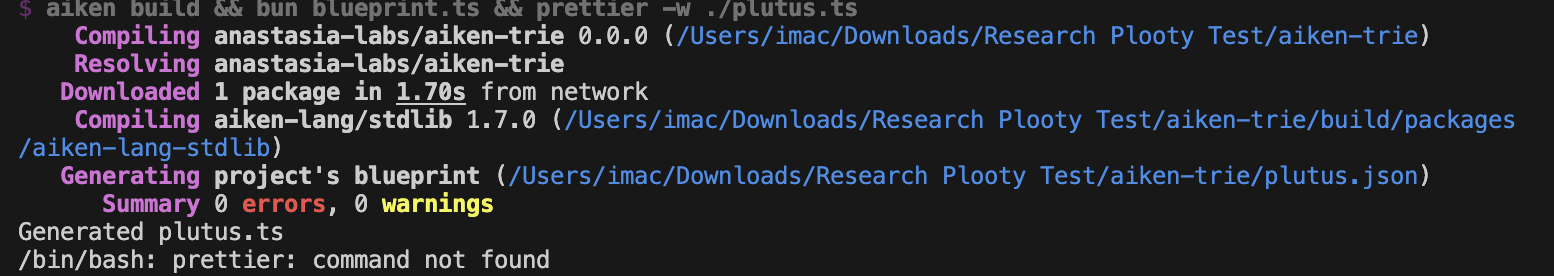
// failure successful

});

});

});

#### Output:



import { Emulator, Lucid } from "lucid-cardano";

import { defaultProtocolParams } from "./constants";

import { generateAccount } from "./utils";

import { trieScript } from "../src/const";

import {

appendTrie,

betweenTrie,

createTrie,

getTrieOrigin,

getUtxoByKey,

} from "../src";

console.log(`Trie Script Size: ${trieScript.script.length / 2}`);

describe("Synthetics", () => {

let emulator: Emulator;

let lucid: Lucid;

let trieAddress: string;

let trieRewardAddress: string;

beforeEach(async () => {

emulator = new Emulator([]);

lucid = await Lucid.new(emulator);

trieAddress = lucid.utils.validatorToAddress(trieScript);

trieRewardAddress = lucid.utils.validatorToRewardAddress(trieScript);

const TRIE\_USER = await generateAccount({

lovelace: BigInt(100000000000000000),

});

emulator = new Emulator([TRIE\_USER], defaultProtocolParams);

emulator.chain[trieRewardAddress] = {

registeredStake: true,

delegation: { rewards: BigInt(0), poolId: "" },

};

lucid = await Lucid.new(emulator);

lucid.selectWalletFromPrivateKey(TRIE\_USER.privateKey);

});

it("Should pass e2e on creation, two insertions", async () => {

let trie = await createTrie(lucid, trieAddress, trieRewardAddress);

emulator.awaitBlock(1);

let trieOrigin = await getTrieOrigin(lucid, trie.trieUnit, trieAddress);

let trieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await appendTrie(

lucid,

trie.trieUnit,

trieOrigin!,

trieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

);

emulator.awaitBlock(1);

let newTrieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await betweenTrie(

lucid,

trie.trieUnit,

trieOrigin!,

newTrieUtxo!,

"hello",

trieAddress,

trieRewardAddress,

);

});

it("Should fail on duplicate insertion (1)", async () => {

let trie = await createTrie(lucid, trieAddress, trieRewardAddress);

emulator.awaitBlock(1);

let trieOrigin = await getTrieOrigin(lucid, trie.trieUnit, trieAddress);

let trieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await appendTrie(

lucid,

trie.trieUnit,

trieOrigin!,

trieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

);

emulator.awaitBlock(1);

let newTrieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

betweenTrie(

lucid,

trie.trieUnit,

trieOrigin!,

newTrieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

)

.then(() => {

throw "This should have failed";

})

.catch(() => {

// failure successful

});

});

it("Should fail on duplicate insertion (2)", async () => {

let trie = await createTrie(lucid, trieAddress, trieRewardAddress);

emulator.awaitBlock(1);

let trieOrigin = await getTrieOrigin(lucid, trie.trieUnit, trieAddress);

let trieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

await appendTrie(

lucid,

trie.trieUnit,

trieOrigin!,

trieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

);

emulator.awaitBlock(1);

let newTrieUtxo = await getUtxoByKey(lucid, trie.trieUnit, "", trieAddress);

betweenTrie(

lucid,

trie.trieUnit,

trieOrigin!,

newTrieUtxo!,

"hello\_world",

trieAddress,

trieRewardAddress,

)

.then(() => {

throw "This should have failed";

})

.catch(() => {

// failure successful

});

});

});

### 

### 4. Test Case Name: Aiken-Mode

;;; aiken-mode.el --- Major mode for Aiken -\*- lexical-binding: t -\*-

;; Copyright © 2023 Sebastian Nagel <sebastian.nagel@ncoding.at>

;; Author: Sebastian Nagel <sebastian.nagel@ncoding.at>

;; URL: https://github.com/aiken-lang/aiken-mode

;; Keywords: languages aiken

;; Version: 1.0.2

;; Package-Requires: ((emacs "26.1"))

;; SPDX-License-Identifier: MPL-2.0

;; This file is NOT part of GNU Emacs.

;; This Source Code Form is subject to the terms of the Mozilla Public

;; License, v. 2.0. If a copy of the MPL was not distributed with this

;; file, You can obtain one at http://mozilla.org/MPL/2.0/.

;;; Commentary:

;; Provides syntax highlighting for the Aiken smart contract language.

;;; Code:

;; Aiken syntax

(defvar aiken-keywords

'("if"

"else"

"when"

"is"

"fn"

"use"

"let"

"pub"

"type"

"opaque"

"const"

"todo"

"error"

"expect"

"test"

"trace"

"fail"

"validator"

"and"

"or"))

(defvar aiken-operators

'(

"="

"->"

".."

"|>"

">="

"<="

">"

"<"

"!="

"=="

"&&"

"||"

"!"

"+"

"-"

"/"

"\*"

"%"

"?"))

(defvar aiken-font-lock-keywords

(append

`(

;; Keywords

(,(regexp-opt aiken-keywords 'symbols) . font-lock-keyword-face)

;; CamelCase is a type

("[[:upper:]][[:word:]]\*" . font-lock-type-face)

;; Operators

(,(regexp-opt aiken-operators nil) . font-lock-builtin-face))

;; Identifiers after keywords

(mapcar (lambda (x)

(list (concat (car x) "[^(]\\(\\w\*\\)")

1 ;; apply face ot first match group

(cdr x)))

'(("const" . font-lock-type-face)

("type" . font-lock-type-face)

("use" . font-lock-constant-face)

("fn" . font-lock-function-name-face)))))

;; Mode definitions

;;;###autoload

(define-derived-mode aiken-mode prog-mode "Aiken"

"Major mode for Aiken code."

:group 'aiken-mode

(setq-local indent-tabs-mode nil)

;; Syntax highlighting via font-lock

(setq-local font-lock-defaults '(aiken-font-lock-keywords))

;; Syntax: make \_ part of words

(modify-syntax-entry ?\_ "w" aiken-mode-syntax-table)

;; Comment syntax

(modify-syntax-entry ?/ ". 124b" aiken-mode-syntax-table)

(modify-syntax-entry ?\n "> b" aiken-mode-syntax-table)

(modify-syntax-entry ?\^m "> b" aiken-mode-syntax-table)

;; Comment settings

(setq-local comment-start "// ")

(setq-local comment-end "")

(setq-local comment-start-skip "//+ \*")

(setq-local comment-use-syntax t)

(setq-local comment-auto-fill-only-comments t))

;;;###autoload

(add-to-list 'auto-mode-alist '("\\.ak\\'" . aiken-mode))

(provide 'aiken-mode)

;;; aiken-mode.el ends here

### 5. Test Case Name:Aiken-Markle\_Tree

5. Aiken-Markle-Tree

<!-- markdown-toc start - Don't edit this section. Run M-x markdown-toc-refresh-toc -->

# Table of Contents

- [Aiken Merkle Tree](#aiken-merkle-tree)

- [Introduction](#introduction)

- [Documentation](#documentation)

- [Merkle Tree](#merkle-tree)

- [Aiken Merkle Tree](#aiken-merkle-tree-implementation)

- [Getting Started](#getting-started)

- [Prerequisites](#prerequisites)

- [Building and Developing](#building-and-developing)

- [Sample validator](#sample-validator)

- [Case study](#case-study)

- [Acknowledgments](#acknowledgments)

<!-- markdown-toc end -->

# Aiken Merkle Tree

## Introduction

The Aiken Merkle Tree project provides a Plutarch-based implementation of Merkle Trees for the Cardano blockchain. This project allows developers to leverage the security and efficiency of Merkle Trees in their Cardano smart contracts, ensuring data integrity and efficient data verification.

This project is funded by the Cardano Treasury in [Catalyst Fund 10](https://projectcatalyst.io/funds/10/f10-osde-open-source-dev-ecosystem/anastasia-labs-the-trifecta-of-data-structures-merkle-trees-tries-and-linked-lists-for-cutting-edge-contracts) and is aimed at enhancing the capabilities of Cardano smart contracts in handling complex data structures.

## Documentation

### Merkle Tree

A Merkle tree, named after its inventor Ralph Merkle, is a fundamental data structure in computer science and cryptography. It's particularly well-suited for managing and verifying large data structures, especially in distributed systems like blockchain technologies. Here's a detailed explanation:

#### Basic concept

A Merkle tree is a type of binary tree, consisting of nodes. Here's how it's structured:

- \*\*Leaf Nodes\*\*: These are the bottom-most nodes in the tree and contain hashed data. The data could be transactions (as in blockchain), files, or any data chunks.

- \*\*Non-Leaf (Intermediate) Nodes\*\*: These nodes store a cryptographic hash of the combined data of their child nodes.

- \*\*Root Node\*\*: The single node at the top of the tree contains a hash formed by its child nodes, ultimately representing the hashes of all lower levels.

#### Hash function

The core of a Merkle tree is the hash function (like SHA-256 in Bitcoin). This function takes digital data of any size and produces a fixed-size string of bytes, typically a unique digital fingerprint of the input data.

#### Construction

- \*\*Hashing the Data\*\*: First, each piece of data at the leaf level is hashed.

- \*\*Pairing and Hashing Upwards\*\*: These hashes are then paired and concatenated, and the resultant string is hashed again. This process continues until you reach the single hash at the top - the root hash.

- \*\*Tree Structure\*\*: This process creates a tree-like structure where each parent node is a hash of its children, providing a secure and efficient means of verifying the contents of the tree.

#### Features

- \*\*Efficiency in Verification\*\*: To verify any single data chunk's integrity, you don't need to download the entire tree. You only need the hashes of the nodes along the path from your data chunk to the root.

- \*\*Tamper-Proof\*\*: Any change in a leaf node (data) will result in a completely different root hash through a cascading effect of changes in the intermediate hashes. This makes it easy to detect alterations.

- \*\*Concurrency Friendly\*\*: Multiple branches of the tree can be processed simultaneously, making Merkle trees highly efficient for parallel processing.

#### Example

Consider a Merkle tree with four leaf nodes (A, B, C, D).

```

Merkle Root

|

+-----------+-----------+

| |

Hash(A+B) Hash(C+D)

| |

+---+---+ +---+---+

| | | |

Hash(A) Hash(B) Hash(C) Hash(D)

```

1. Each of A, B, C, and D is hashed: Hash(A), Hash(B), Hash(C), Hash(D).

2. The hashes of A and B are combined and hashed: Hash(Hash(A) + Hash(B)). Similarly for C and D.

3. The hash results from step 2 are combined and hashed to give the Merkle root.

Thus, the Merkle root is a digest of all the data in the leaf nodes.

In conclusion, Merkle trees offer a secure and efficient way to summarize and verify large data sets.

### Aiken Merkle Tree implementation

The Aiken Merkle Tree implementation provides several functions to create and manipulate Merkle Trees. Below is a brief overview of each function:

-`from\_list`: Constructs a Merkle Tree from a list of serialized data.

-`to\_list`: Deconstructs a Merkle Tree back into a list of elements.

-`root`: Retrieves the root hash of a Merkle Tree.

-`is\_empty`: Checks if a Merkle Tree is empty.

-`size`: Returns the number of leaf nodes in a Merkle Tree.

-`get\_proof`: Generates a proof of membership for an element in the Merkle Tree.

-`is\_member`: Verifies if an element is part of a Merkle Tree using a proof.

-`combine`: Combines two hashes into a new one.

## Getting Started

### Prerequisites

Before you begin, ensure you have [aiken](https://aiken-lang.org/installation-instructions) installed on your system.

### Building and developing

Once Aiken is installed, you should be able to seamlessly use the repository to

develop, build and run packages.

Download the Git repository:

```sh

git clone https://github.com/Anastasia-Labs/aiken-merkle-tree.git

```

Navigate to the repository directory:

```sh

cd aiken-merkle-tree

```

Execute the test suite:

```sh

aiken check

```

Build:

```sh

aiken build

```

![aiken-merkle-tree.gif](/assets/images/aiken-merkle-tree.gif)

# Sample validator

For a complete example, including tests and further explanations, reger to the provided sample validator: [sample.ak](validators/sample.ak).

# Case study

For an in-depth real-world case study on the application of Merkle Trees within the Cardano blockchain environment, particularly in the context of sidechain to main chain token transfers, refer to the following resource:

[Cardano Sidechain Toolkit - Main Chain Plutus Scripts](https://docs.cardano.org/cardano-sidechains/sidechain-toolkit/mainchain-plutus-scripts/)

This case study provides valuable insights into how Merkle Trees are integrated into blockchain transactions, offering practical examples and detailed workflows.

# Acknowledgments

This library takes the Aiken code from the aiken-lang.

The repository can be found at <https://github.com/aiken-lang/trees>

#### Validator:

**// /\*\***

**// \* This file contains the implementation of a spend\_validator function that validates a spend transaction**

**// \* using a Merkle tree. It also includes a test case for the spend\_validator function.**

**// \*/**

**use aiken/transaction.{OutputReference, ScriptContext, Spend, TransactionId}**

**use aiken\_merkle\_tree/mt.{Proof, Root, from\_list, get\_proof, is\_member, root}**

**type MyDatum {**

**merkle\_root: Root,**

**}**

**type MyRedeemer {**

**my\_proof: Proof<ByteArray>,**

**user\_data: ByteArray,**

**}**

**// /\*\***

**// \* The spend\_validator function validates a spend transaction by checking if the user data is a member of**

**// \* the Merkle tree with the given Merkle root and proof.**

**// \***

**// \* @param datum - The MyDatum object containing the Merkle root.**

**// \* @param redeemer - The MyRedeemer object containing the proof and user data.**

**// \* @param \_ctx - The ScriptContext.**

**// \*/**

**validator {**

**fn spend\_validator(datum: MyDatum, redeemer: MyRedeemer, \_ctx: ScriptContext) {**

**let MyDatum { merkle\_root } = datum**

**let MyRedeemer { my\_proof, user\_data } = redeemer**

**is\_member(merkle\_root, user\_data, my\_proof, identity)**

**}**

**}**

**// /\*\***

**// \* Test case for the spend\_validator function.**

**// \*/**

**test spend\_validator\_1() {**

**let data =**

**["dog", "cat", "mouse"]**

**let merkle\_tree = from\_list(data, identity)**

**let datum = MyDatum { merkle\_root: root(merkle\_tree) }**

**expect Some(proof) = get\_proof(merkle\_tree, "cat", identity)**

**let redeemer = MyRedeemer { my\_proof: proof, user\_data: "cat" }**

**let placeholder\_utxo =**

**OutputReference { transaction\_id: TransactionId(""), output\_index: 0 }**

**let context =**

**ScriptContext {**

**purpose: Spend(placeholder\_utxo),**

**transaction: transaction.placeholder(),**

**}**

**spend\_validator(datum, redeemer, context)**

**}**



### 6. Test Case Name: Emacs Mode Aiken-Lang

# Emacs mode for [aiken](https://github.com/aiken-lang/aiken)

An emacs major mode providing syntax highlighting, indentation and formatting

commands for the Aiken smart contract language.

## Features

- [x] Syntax highlighting

- [ ] `aiken fmt` command and on-save

- [ ] Indentation

- [ ] Aiken LSP client

## Installation

The package is [not yet on MELPA](https://github.com/melpa/melpa/pull/8736), so pointing your emacs config to this

repository is the way to go for now.

### [doom-emacs](https://github.com/doomemacs/doomemacs/) (recommended :smiling\_imp:)

Add this to your `packages.el`:

```elisp

(package! aiken-mode

:recipe (:host github :repo "aiken-lang/aiken-mode"))

```

Add this to your `config.el`:

``` elisp

(use-package! aiken-mode)

```

and run `doom sync`.

For faster feedback time during development:

```elisp

(package! aiken-mode

:recipe (:local-repo "~/path/to/aiken-mode"))

```

### use-package

```elisp

(use-package aiken-mode

:load-path "~/path/to/aiken-mode")

```

### vanilla

```elisp

(add-to-list 'load-path "~/path/to/aiken-mode")

(load-library "aiken-mode")

```

### 7. Tet Case Name: LenFi Smart Contract

Link: https://github.com/lenfiLabs/lenfi-smart-contracts/blob/main/README.md

# Lenfi - Permissionless Borrowing

## Introduction

We are building the product and its validators with a singular focus on creating a \*\*Permissionless system\*\*. This system is designed to be balanced, governed by smart contracts known as validators. It is an open system, free from dependencies on specific systems or individuals.

## Terminology

- \*\*NFT\*\*: A unique, non-fungible token whose uniqueness is guaranteed by a consumed UTXO reference.

- \*\*Pool\*\*: A UTXO with a Pool NFT, accompanied by a deposit and details (datum) locked in a `pool.ak` validator.

- \*\*Oracle\*\*: A UTXO with an Oracle NFT and details (datum) locked in a `collateral.ak` validator.

- \*\*Liquidity Provider (LP) Token\*\*: Fungible tokens that represent a share of the pool deposit.

- \*\*Collateral\*\*: Assets used as security to cover the value of loans taken from the pool.

- \*\*Interest Rate\*\*: Annual Percentage Rate (APR) charged on the loan amount.

- \*\*Loan\*\*: An agreement to borrow assets from a pool, governed by specific interest rates and collateral requirements.

- \*\*Pool Manager NFT\*\*: An NFT minted at the time of pool creation that allows for control over the stake address attached to `collateral.ak` and `pool.ak`.

## Off-chain validations

These are validation done off-chain, before pool appears on the client side. Failing to match any of below criteria would result in pool being dismissed as 'valid' and not displayed on the UI.

\*\*Pool validation\*\*

- Pool NFT is sent to `pool.ak` and Stake Credentials are correct (`pool\_stake.ak` script credentials)

- Pool NFT policyId matches `pool.ak`

- Minted LP tokens policyId matches `liquidity\_token.ak` policy. Also, it is recorded in pool datum lpToken polilcyId and asset name

- Pool config NFT is minted (policyId matches `pool\_config.ak`) and is sent to pool Config validator

- `pool\_stake.ak` script is locked to `leftovers.ak` for future contract references. With datum coressponding to Pool NFT.

\*\*Pool datum validation\*\*

- `collateralAddress`: validator ScriptCredential matches `collateral.ak`; Stake ScriptCredential matches pool stake credentials;

- `LoanCs`, `CollateralCs` are from approved list of borrowable and collateral assets

- `OracleCollateralNft/OracleLoanNft` is from approved list of oracle NFTs

- `lpToken` token name matches pool token name

- `poolNftName` matches pool NFT minted on the transaction

- `PoolConfigAssetname` matches pool NFT name

- `balance` match pool deposit amount and is used to calculate total\_lp\_tokens. Cannot be 0.

- `lent\_out` = 0

- `total\_lp\_tokens` match amount of LP tokens minted in the transaction

\*\*Oracle validation\*\*

- Oracle NFT is sent to oracle contract. Both match `oracle\_validator.ak` \*\*Oracle datum validation\*\*

- `poolNftCs` - hand checked valid DEX pool NFT associated to the token and ADA

- `oracleNftCs` - minted NFT CS

- `tokenACs` - matches DEX relation

- `tokenBCs` - is ADA ("","")

- `tokenAAmount` - current or approximate tokenA amount in a DEX

- `tokenBAmount` - current or approximate ADA amount in a DEX

- `expirationTime` - epoch time no later than 20 minutes from now

- `maturityTime` - epoch time no later than 20 minutes from now

\*\*Pool Config validation\*\*

- Pool Config NFT is sent to `pool\_config.ak` \*\*Config datum validation\*\* \_\*\*All the values are vetted by Lenfi team.\*\*\_

- `liquidation\_threshold` - is positive and ranging between 1,000,000 - 3,000,000. Cannot be changed

- `initial\_collateral\_ratio` - Is higher than `liquidation\_threshold`. Cannot be changed

- `pool\_fee` - is not negative.

- `loan\_fee\_details.tier\_1\_fee` - is not negative

- `loan\_fee\_details.tier\_1\_threshold` - is not negative

- `loan\_fee\_details.tier\_2\_fee` - higher than `tier\_1\_fee`

- `loan\_fee\_details.tier\_2\_threshold` - higher than `tier\_1\_threshold`

- `loan\_fee\_details.tier\_3\_fee` - higher than `tier\_2\_fee`

- `loan\_fee\_details.tier\_3\_threshold` - higher than `tier\_2\_threshold`

- `liquidation\_fee` - Is not negative

- `platform\_fee\_collector\_address`- is valid address

- `min\_transition` - is not negative

- `min\_loan` - is not negative

\*\*Reference Token validation\*\*

Reference token is minted at a time of protocol creation. Stores script hashes in a lock-forever script, replaces some script params and avoids circular imports thruought the protocol usage.

Offchain validates that Reference Token is and NFT locked in 'always fail' script with correct Datum values. `delegator\_nft\_policy` - Correct pool manager NFT policyId (`placeholder\_nft.ak` with param of `3`) `liquidations\_pkh` - Correct liquidations script hash (`leftovers.ak`) `pool\_script\_hash` - Correct pool script hash (`pool.ak`) `pool\_config\_script\_hash`- Correct pool config script hash (`pool\_config.ak`) `merge\_script\_hash`- Correct delayed merge script hash (`delayed\_merge.ak`)

## Governance control

Unique governance NFT is minted that has an access to control every pool config (stored in poolConfigValidator). NFT will be held by developer team before it is moved to DAO validator (see `pool\_config.ak` for more details). Governance NFT holder will have a right to change any `Pool Config Datum` except `liquidation\_threshold` and `initial\_collateral\_ratio`.

## Validators Definitions

- `collateral.ak`: Ensures fair loan issuance and repayment from the pool.

- `pool.ak`: Manages the proper utilization of user funds.

- `liquidity\_token.ak`: Oversees fair deposits and withdrawals from the pool.

- `pool\_config.ak`: Manages pool-specific parameters which can be adjusted based on off-chain consensus.

- `pool\_stake.ak`: Ensures that delegated ADA is withdrawn to the pool. Allows to control stake address.

- `order\_contract.ak`: Executes user orders fairly when direct execution with the pool is not possible.

- `oracle\_validator.ak`: Streams token prices to the protocol.

- `leftovers.ak`: Manages the return of any remaining collateral after loan repayment or liquidation.

- `delayed\_repayment\_merge.ak`: Allows to repay/liquidate the loan by sending funds to intermediate contract. This is handy when pool is busy.

- `placeholder\_nft.ak`: Ensures that minted NFTs are unique.

## Protocol Requirements

- Each pool only accepts one collateral asset defined in the pool datum

- LP token holders are entitled to a proportional share of the pool balance.

- An LP token's value in relation to the pool balance can never decrease; it can only appreciate due to accrued interest.

- Borrower NFT guarantees the right to reclaim 100% of collateral if the loan is repaid.

- Borrower NFT also guarantees the right to any remaining collateral, after loan and fees, if liquidation occurs.

- Loans must be overcollateralized.

- Loans can only be liquidated if undercollateralized at the time of liquidation.

- Liquidators are allowed to claim a percentage of the collateral as a fee during liquidation.

- Liquidateion fee is set in `pool\_config.ak` as a % of total collateral

- All loans must eventually be repaid; failure to do so will result in the loan becoming undercollateralized due to increasing interest.

- Delegation rewards can only be withdrawn to the pool.

- Pool Manager NFT allows delegation to any Stake Pool (SPO).

- Pool configuration (`pool\_config.ak`) values can be adjusted.

- Oracles are used to determine asset values during borrow, liquidate, and withdraw actions.

- Oracle data must be updated based on values from a Decentralized Exchange (DEX) and is considered expired if older than 30 minutes.

- Oracle data cannot be updated more frequently than every 10 minutes.

- A pool fee, set in `pool\_config`, is payable to the pool.

- Utilization rate (`lent\_out / (balance + lent\_out)`) determines the platform fee payable to the `platform\_fee\_collector\_address`.

- The pool datum must always accurately reflect the total supply of LP tokens n=n.

- Each pool must have unique: Pool NFT, pool address, stake address, and collateral address.

- The same stake address must be enforced for both the pool and the collateral.

- A pool can be destroyed if all balance is withdrawn and the Pool Manager NFT is burned.

- When user creates a batch order, they strictly define value they want to receive.

- Anyone can execute a batch order created by user.

## Use Cases

- Create a pool

- Delegate to an SPO

- Destroy a pool

- Deposit to a pool (may use `order\_contract.ak`)

- Withdraw from a pool (may use `order\_contract.ak`)

- Borrow (may use `order\_contract.ak`)

- Repay (may use `order\_contract.ak`)

- Liquidate (may use `order\_contract.ak`)

- Create an oracle

- Update oracle prices

## User Journeys

### Lender

1. Creates a pool.

2. Deposits funds into the pool and mints LP tokens.

3. Withdraws 50% from the pool and burns the corresponding LP tokens.

4. Withdraws the remaining 50% and destroys the pool.

### Borrower

1. Borrows from the pool, locks collateral, and mints a Borrower NFT.

2. Repays the loan, unlocks the collateral, and burns the Borrower NFT.

### Liquidator

1. Liquidates an undercollateralized loan by repaying the loan on behalf of the borrower.

2. Locks the remaining collateral for the borrower to claim later.

## Actions

### Creating a Pool

- A unique Pool NFT must be minted and locked in `pool.ak`.

- A unique Pool Manager NFT and Pool Config NFT must be minted.

- An initial deposit must be made to the pool.

- LP tokens must be minted to represent the initial deposit.

- LP token asset names must match the asset name of the Pool NFT.

- The `balance` and `total\_lp\_tokens` fields in the pool datum must be correctly initialized.

### Depositing to the Pool

- LP tokens must be minted to represent the deposit.

- The asset name of the LP tokens must match the asset name of the Pool NFT into which the deposit is made.

- The `balance` field in the pool datum must be incremented by the deposit amount.

- The `total\_lp\_tokens` field in the pool datum must be incremented by the amount of minted LP tokens.

### Withdrawing from the Pool

- The corresponding amount of LP tokens must be burned to represent the withdrawal.

- The asset name of the LP tokens must match the asset name of the Pool NFT from which the withdrawal is made.

- The asset name of the LP tokens must match the asset name of the Pool NFT into which the deposit is made.

- The `balance` field in the pool datum must be decremented by the withdrawal amount.

- The `total\_lp\_tokens` field in the pool datum must be decremented by the amount of burned LP tokens.

### Borrowing

- Mint a Borrower NFT upon initiating a loan.

- The loan amount should be smaller than the value of the collateral plus a buffer, defined as the liquidation threshold.

- Determine the interest rate as an Annual Percentage Rate (APR) for the entire loan duration.

- Securely lock collateral, which can either be claimed by the borrower upon repayment or liquidated if the loan becomes undercollateralized.

- Subject borrowers to a Loan-to-Value (LTV) ratio, as stipulated in the pool's `pool\_config.ak`.

- Specify the loan duration at the time of borrowing.

- Invoke `oracle\_validator.ak` to determine the current value of both the collateral and the borrowed asset.

- Use `leftovers.ak` to ensure that any remaining collateral is returned to the borrower once the loan is repaid or liquidated.

- Decrease the `balance` and increment `lent\_out` fields in the pool's datum.

### Repayment

- Only the holder of the Borrower NFT is authorized to repay the loan.

- Calculate the interest based on the duration the loan was active.

- Both the loan amount and interest should be repaid to the pool.

- Update the pool datum: increment the `balance` and decrement the `lent\_out` fields.

- Grant the borrower rights to reclaim 100% of the initially locked collateral.

### Liquidation

- Allow liquidation only when a loan is undercollateralized.

- Repay the loan and interest to the pool.

- Entitle the liquidator to a percentage of the collateral, where the percentage is specified in `pool\_config` (denominated by 1,000,000, e.g., 2% = 20,000).

- Make the remaining collateral claimable by the holder of the Borrower NFT.

### Delegation and Withdrawal

- Enable the Pool Manager NFT holder to delegate `pool.ak` and `collateral.ak` stake addresses to any Stake Pool Operator (SPO).

- Allow delegation rewards to be withdrawn only to the pool.

- Convert non-ADA assets to ADA at a fair market price (utilize the oracle) upon withdrawal.

### Oracle UTXO Update

- Update the Oracle UTXO strictly based on values from a Decentralized Exchange (DEX) UTXO.

- Restrict updates to a minimum interval of 10 minutes.

- Updated value is an average of datum from unlocked oracle UTXO and DEX

# Lenfi - Permissionless Borrowing

## Introduction

We are building the product and its validators with a singular focus on creating a \*\*Permissionless system\*\*. This system is designed to be balanced, governed by smart contracts known as validators. It is an open system, free from dependencies on specific systems or individuals.

## Terminology

- \*\*NFT\*\*: A unique, non-fungible token whose uniqueness is guaranteed by a consumed UTXO reference.

- \*\*Pool\*\*: A UTXO with a Pool NFT, accompanied by a deposit and details (datum) locked in a `pool.ak` validator.

- \*\*Oracle\*\*: A UTXO with an Oracle NFT and details (datum) locked in a `collateral.ak` validator.

- \*\*Liquidity Provider (LP) Token\*\*: Fungible tokens that represent a share of the pool deposit.

- \*\*Collateral\*\*: Assets used as security to cover the value of loans taken from the pool.

- \*\*Interest Rate\*\*: Annual Percentage Rate (APR) charged on the loan amount.

- \*\*Loan\*\*: An agreement to borrow assets from a pool, governed by specific interest rates and collateral requirements.

- \*\*Pool Manager NFT\*\*: An NFT minted at the time of pool creation that allows for control over the stake address attached to `collateral.ak` and `pool.ak`.

## Off-chain validations

These are validation done off-chain, before pool appears on the client side. Failing to match any of below criteria would result in pool being dismissed as 'valid' and not displayed on the UI.

\*\*Pool validation\*\*

- Pool NFT is sent to `pool.ak` and Stake Credentials are correct (`pool\_stake.ak` script credentials)

- Pool NFT policyId matches `pool.ak`

- Minted LP tokens policyId matches `liquidity\_token.ak` policy. Also, it is recorded in pool datum lpToken polilcyId and asset name

- Pool config NFT is minted (policyId matches `pool\_config.ak`) and is sent to pool Config validator

- `pool\_stake.ak` script is locked to `leftovers.ak` for future contract references. With datum coressponding to Pool NFT.

\*\*Pool datum validation\*\*

- `collateralAddress`: validator ScriptCredential matches `collateral.ak`; Stake ScriptCredential matches pool stake credentials;

- `LoanCs`, `CollateralCs` are from approved list of borrowable and collateral assets

- `OracleCollateralNft/OracleLoanNft` is from approved list of oracle NFTs

- `lpToken` token name matches pool token name

- `poolNftName` matches pool NFT minted on the transaction

- `PoolConfigAssetname` matches pool NFT name

- `balance` match pool deposit amount and is used to calculate total\_lp\_tokens. Cannot be 0.

- `lent\_out` = 0

- `total\_lp\_tokens` match amount of LP tokens minted in the transaction

\*\*Oracle validation\*\*

- Oracle NFT is sent to oracle contract. Both match `oracle\_validator.ak` \*\*Oracle datum validation\*\*

- `poolNftCs` - hand checked valid DEX pool NFT associated to the token and ADA

- `oracleNftCs` - minted NFT CS

- `tokenACs` - matches DEX relation

- `tokenBCs` - is ADA ("","")

- `tokenAAmount` - current or approximate tokenA amount in a DEX

- `tokenBAmount` - current or approximate ADA amount in a DEX

- `expirationTime` - epoch time no later than 20 minutes from now

- `maturityTime` - epoch time no later than 20 minutes from now

\*\*Pool Config validation\*\*

- Pool Config NFT is sent to `pool\_config.ak` \*\*Config datum validation\*\* \_\*\*All the values are vetted by Lenfi team.\*\*\_

- `liquidation\_threshold` - is positive and ranging between 1,000,000 - 3,000,000. Cannot be changed

- `initial\_collateral\_ratio` - Is higher than `liquidation\_threshold`. Cannot be changed

- `pool\_fee` - is not negative.

- `loan\_fee\_details.tier\_1\_fee` - is not negative

- `loan\_fee\_details.tier\_1\_threshold` - is not negative

- `loan\_fee\_details.tier\_2\_fee` - higher than `tier\_1\_fee`

- `loan\_fee\_details.tier\_2\_threshold` - higher than `tier\_1\_threshold`

- `loan\_fee\_details.tier\_3\_fee` - higher than `tier\_2\_fee`

- `loan\_fee\_details.tier\_3\_threshold` - higher than `tier\_2\_threshold`

- `liquidation\_fee` - Is not negative

- `platform\_fee\_collector\_address`- is valid address

- `min\_transition` - is not negative

- `min\_loan` - is not negative

\*\*Reference Token validation\*\*

Reference token is minted at a time of protocol creation. Stores script hashes in a lock-forever script, replaces some script params and avoids circular imports thruought the protocol usage.

Offchain validates that Reference Token is and NFT locked in 'always fail' script with correct Datum values. `delegator\_nft\_policy` - Correct pool manager NFT policyId (`placeholder\_nft.ak` with param of `3`) `liquidations\_pkh` - Correct liquidations script hash (`leftovers.ak`) `pool\_script\_hash` - Correct pool script hash (`pool.ak`) `pool\_config\_script\_hash`- Correct pool config script hash (`pool\_config.ak`) `merge\_script\_hash`- Correct delayed merge script hash (`delayed\_merge.ak`)

## Governance control

Unique governance NFT is minted that has an access to control every pool config (stored in poolConfigValidator). NFT will be held by developer team before it is moved to DAO validator (see `pool\_config.ak` for more details). Governance NFT holder will have a right to change any `Pool Config Datum` except `liquidation\_threshold` and `initial\_collateral\_ratio`.

## Validators Definitions

- `collateral.ak`: Ensures fair loan issuance and repayment from the pool.

- `pool.ak`: Manages the proper utilization of user funds.

- `liquidity\_token.ak`: Oversees fair deposits and withdrawals from the pool.

- `pool\_config.ak`: Manages pool-specific parameters which can be adjusted based on off-chain consensus.

- `pool\_stake.ak`: Ensures that delegated ADA is withdrawn to the pool. Allows to control stake address.

- `order\_contract.ak`: Executes user orders fairly when direct execution with the pool is not possible.

- `oracle\_validator.ak`: Streams token prices to the protocol.

- `leftovers.ak`: Manages the return of any remaining collateral after loan repayment or liquidation.

- `delayed\_repayment\_merge.ak`: Allows to repay/liquidate the loan by sending funds to intermediate contract. This is handy when pool is busy.

- `placeholder\_nft.ak`: Ensures that minted NFTs are unique.

## Protocol Requirements

- Each pool only accepts one collateral asset defined in the pool datum

- LP token holders are entitled to a proportional share of the pool balance.

- An LP token's value in relation to the pool balance can never decrease; it can only appreciate due to accrued interest.

- Borrower NFT guarantees the right to reclaim 100% of collateral if the loan is repaid.

- Borrower NFT also guarantees the right to any remaining collateral, after loan and fees, if liquidation occurs.

- Loans must be overcollateralized.

- Loans can only be liquidated if undercollateralized at the time of liquidation.

- Liquidators are allowed to claim a percentage of the collateral as a fee during liquidation.

- Liquidateion fee is set in `pool\_config.ak` as a % of total collateral

- All loans must eventually be repaid; failure to do so will result in the loan becoming undercollateralized due to increasing interest.

- Delegation rewards can only be withdrawn to the pool.

- Pool Manager NFT allows delegation to any Stake Pool (SPO).

- Pool configuration (`pool\_config.ak`) values can be adjusted.

- Oracles are used to determine asset values during borrow, liquidate, and withdraw actions.

- Oracle data must be updated based on values from a Decentralized Exchange (DEX) and is considered expired if older than 30 minutes.

- Oracle data cannot be updated more frequently than every 10 minutes.

- A pool fee, set in `pool\_config`, is payable to the pool.

- Utilization rate (`lent\_out / (balance + lent\_out)`) determines the platform fee payable to the `platform\_fee\_collector\_address`.

- The pool datum must always accurately reflect the total supply of LP tokens n=n.

- Each pool must have unique: Pool NFT, pool address, stake address, and collateral address.

- The same stake address must be enforced for both the pool and the collateral.

- A pool can be destroyed if all balance is withdrawn and the Pool Manager NFT is burned.

- When user creates a batch order, they strictly define value they want to receive.

- Anyone can execute a batch order created by user.

## Use Cases

- Create a pool

- Delegate to an SPO

- Destroy a pool

- Deposit to a pool (may use `order\_contract.ak`)

- Withdraw from a pool (may use `order\_contract.ak`)

- Borrow (may use `order\_contract.ak`)

- Repay (may use `order\_contract.ak`)

- Liquidate (may use `order\_contract.ak`)

- Create an oracle

- Update oracle prices

## User Journeys

### Lender

1. Creates a pool.

2. Deposits funds into the pool and mints LP tokens.

3. Withdraws 50% from the pool and burns the corresponding LP tokens.

4. Withdraws the remaining 50% and destroys the pool.

### Borrower

1. Borrows from the pool, locks collateral, and mints a Borrower NFT.

2. Repays the loan, unlocks the collateral, and burns the Borrower NFT.

### Liquidator

1. Liquidates an undercollateralized loan by repaying the loan on behalf of the borrower.

2. Locks the remaining collateral for the borrower to claim later.

## Actions

### Creating a Pool

- A unique Pool NFT must be minted and locked in `pool.ak`.

- A unique Pool Manager NFT and Pool Config NFT must be minted.

- An initial deposit must be made to the pool.

- LP tokens must be minted to represent the initial deposit.

- LP token asset names must match the asset name of the Pool NFT.

- The `balance` and `total\_lp\_tokens` fields in the pool datum must be correctly initialized.

### Depositing to the Pool

- LP tokens must be minted to represent the deposit.

- The asset name of the LP tokens must match the asset name of the Pool NFT into which the deposit is made.

- The `balance` field in the pool datum must be incremented by the deposit amount.

- The `total\_lp\_tokens` field in the pool datum must be incremented by the amount of minted LP tokens.

### Withdrawing from the Pool

- The corresponding amount of LP tokens must be burned to represent the withdrawal.

- The asset name of the LP tokens must match the asset name of the Pool NFT from which the withdrawal is made.

- The asset name of the LP tokens must match the asset name of the Pool NFT into which the deposit is made.

- The `balance` field in the pool datum must be decremented by the withdrawal amount.

- The `total\_lp\_tokens` field in the pool datum must be decremented by the amount of burned LP tokens.

### Borrowing

- Mint a Borrower NFT upon initiating a loan.

- The loan amount should be smaller than the value of the collateral plus a buffer, defined as the liquidation threshold.

- Determine the interest rate as an Annual Percentage Rate (APR) for the entire loan duration.

- Securely lock collateral, which can either be claimed by the borrower upon repayment or liquidated if the loan becomes undercollateralized.

- Subject borrowers to a Loan-to-Value (LTV) ratio, as stipulated in the pool's `pool\_config.ak`.

- Specify the loan duration at the time of borrowing.

- Invoke `oracle\_validator.ak` to determine the current value of both the collateral and the borrowed asset.

- Use `leftovers.ak` to ensure that any remaining collateral is returned to the borrower once the loan is repaid or liquidated.

- Decrease the `balance` and increment `lent\_out` fields in the pool's datum.

### Repayment

- Only the holder of the Borrower NFT is authorized to repay the loan.

- Calculate the interest based on the duration the loan was active.

- Both the loan amount and interest should be repaid to the pool.

- Update the pool datum: increment the `balance` and decrement the `lent\_out` fields.

- Grant the borrower rights to reclaim 100% of the initially locked collateral.

### Liquidation

- Allow liquidation only when a loan is undercollateralized.

- Repay the loan and interest to the pool.

- Entitle the liquidator to a percentage of the collateral, where the percentage is specified in `pool\_config` (denominated by 1,000,000, e.g., 2% = 20,000).

- Make the remaining collateral claimable by the holder of the Borrower NFT.

### Delegation and Withdrawal

- Enable the Pool Manager NFT holder to delegate `pool.ak` and `collateral.ak` stake addresses to any Stake Pool Operator (SPO).

- Allow delegation rewards to be withdrawn only to the pool.

- Convert non-ADA assets to ADA at a fair market price (utilize the oracle) upon withdrawal.

### Oracle UTXO Update

- Update the Oracle UTXO strictly based on values from a Decentralized Exchange (DEX) UTXO.

- Restrict updates to a minimum interval of 10 minutes.

- Updated value is an average of datum from unlocked oracle UTXO and DEX

### 8. Test Case NameAiken-aicone

link:<https://github.com/SundaeSwap-finance/aicone?tab=readme-ov-file>

# aicone

Aiken libraries written and maintained by SundaeSwap Labs

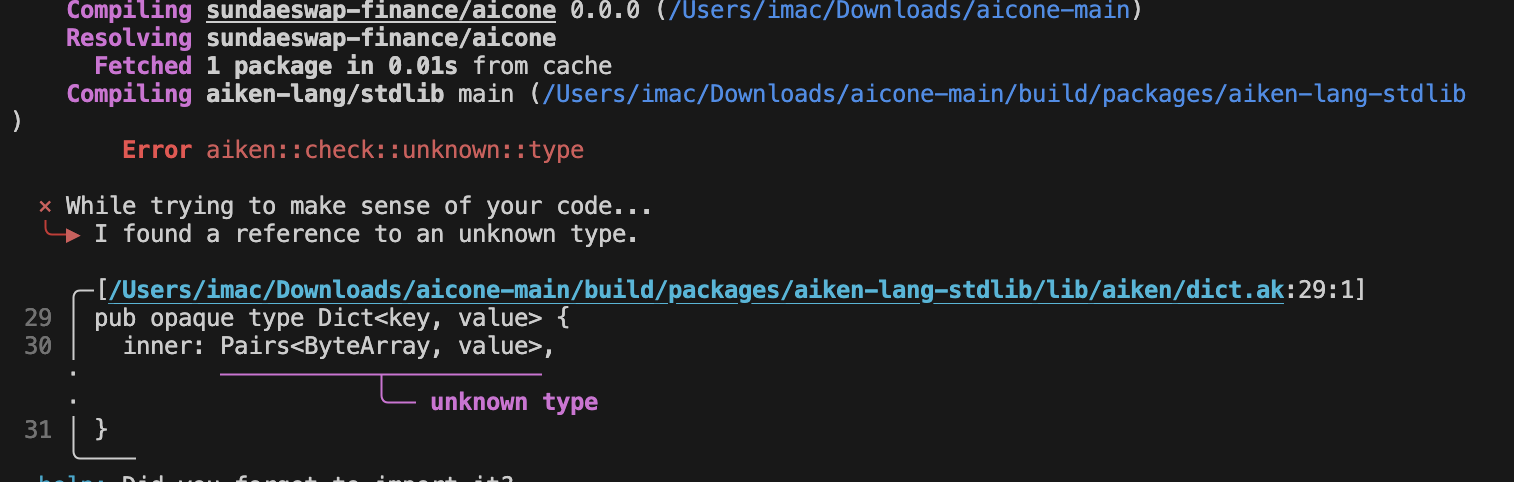
> Note: everything is provided as one library right now, until Aiken supports monorepos.

> see: https://github.com/aiken-lang/aiken/discussions/590

## sundae/multisig

Provides a multisig interface very similar to [Cardano Native Scripts / Simple Scripts](https://github.com/input-output-hk/cardano-node/blob/master/doc/reference/simple-scripts.md#type-sig).

Note: it can't actually be identical to cardano simple scripts, as they use Slots, and plutus scripts don't have access to the slot, only the POSIX time.



### 9. Test Case Name:Aiken Vim/NeoVim

# Aiken Vim

A plugin for working with [Aiken](https://github.com/txpipe/aiken) on Vim / NeoVim.

## Features

- [x] Syntax Highlighting

- [x] Automatic indentation

## Installation

### vim-plug

Simply use:

```vim

Plug 'aiken-lang/editor-integration-nvim'

```

### [lazy.nvim](https://github.com/folke/lazy.nvim)

First add this to lazy.nvim setup:

```lua

{

"aiken-lang/editor-integration-nvim",

dependencies = {

'neovim/nvim-lspconfig',

}

},

```

Then to enable the Aiken LSP, add the following to `init.lua` file:

```lua

require'lspconfig'.aiken.setup({})

```

To enable the auto formatting on save, add the following to `init.lua` file:

```lua

vim.api.nvim\_create\_autocmd("BufWritePre", {

pattern = "\*.ak",

callback = function()

vim.lsp.buf.format({async = false})

end

})

```

### Manual

Copy the content of `ftdetect`, `indent` and `syntax` to your `$HOME/.config/nvim/`.

Make sure that `:syntax` is `on`.

## Preview

![](.github/preview.png)

## License

[MPL-2.0](./LICENSE)

### 10. Test Case Name: EnCoins-Aiken-Core

Link: https://github.com/encryptedcoins/encoins-core-aiken

use aiken/builtin.{

add\_integer, serialise\_data, sha2\_256, snd\_pair, tail\_list,

verify\_ed25519\_signature,

}

use aiken/dict.{has\_key, to\_list}

use aiken/list.{any, filter, foldr, length, map}

use aiken/math.{abs}

use aiken/transaction.{InlineDatum, Mint, ScriptContext}

use aiken/transaction/credential.{Address}

use aiken/transaction/value.{

PolicyId, Value, ada\_asset\_name, ada\_policy\_id, add, flatten, from\_lovelace,

from\_minted\_value, lovelace\_of, merge, negate, policies, tokens, zero,

}

use encoins\_core\_aiken/value\_extra.{isNonnegativeValue}

const minAdaTxOutInLedger = 4000000

const minMaxAdaTxOutInLedger = 1000000000

// Beacon token and verifierPKH

type EncoinsPolicyParams =

(Value, ByteArray)

// Ledger address, change address, total fees

type TxParams =

(Address, Address, Int)

type EncoinsInputOnChain =

(Int, List<(ByteArray, Int)>)

type ProofHash =

ByteArray

type ProofSignature =

ByteArray

type EncoinsRedeemerOnChain =

(TxParams, EncoinsInputOnChain, ProofHash, ProofSignature)

fn hashRedeemer(red: EncoinsRedeemerOnChain) -> ByteArray {

let (params, input, proofHash, \_) = red

sha2\_256(serialise\_data((params, input, proofHash)))

}

fn checkLedgerOuputValue1(vals: List<Value>) {

when vals is {

[] -> True

[x, ..] ->

length(flatten(x)) <= 2 && lovelace\_of(x) >= minMaxAdaTxOutInLedger && checkLedgerOuputValue2(

tail\_list(vals),

)

}

}

fn checkLedgerOuputValue2(vals: List<Value>) {

when vals is {

[] -> True

[x, ..] ->

length(flatten(x)) == 2 && lovelace\_of(x) == minAdaTxOutInLedger && checkLedgerOuputValue2(

tail\_list(vals),

)

}

}

validator(par: EncoinsPolicyParams) {

fn encoinsPolicyCheck(

red: EncoinsRedeemerOnChain,

context: ScriptContext,

) -> Bool {

// Destructuring arguments

let info = context.transaction

let vMint = from\_minted\_value(info.mint)

expect Mint(encoinsSymb) = context.purpose

let (beacon, verifierPKH) = par

let ((ledgerAddr, changeAddr, feesWithSign), (v, inputs), \_, sig) = red

let fees = abs(feesWithSign)

let val = from\_lovelace(v \* 1000000)

let valFees = from\_lovelace(fees \* 1000000)

let deposits = foldr(map(inputs, snd\_pair), 0, add\_integer)

let valDeposits = from\_lovelace(deposits \* minAdaTxOutInLedger)

// Defining conditions

// Condition 0

let actual = to\_list(tokens(vMint, encoinsSymb))

let cond0 = actual == inputs && policies(vMint) == [encoinsSymb]

// Condition 1

let cond1 = verify\_ed25519\_signature(verifierPKH, hashRedeemer(red), sig)

// Condition 2

let refIns = map(info.reference\_inputs, fn(x) { x.output })

let cond2 =

any(

refIns,

fn(o) {

o.address == ledgerAddr && isNonnegativeValue(

merge(o.value, negate(beacon)),

)

},

)

// Conditions 3 and 4

let ins = map(info.inputs, fn(x) { x.output })

let ledgerSpentOuts =

filter(

ins,

fn(x) {

x.address == ledgerAddr && isNonnegativeValue(

add(x.value, ada\_policy\_id, ada\_asset\_name, -minAdaTxOutInLedger),

)

},

## Test Results Summary

* + **Total Tests Conducted:** 10
  + **Passed:**10
  + **Failed:** 0
  + **Issues:** 1
  + **Pass Rate:** 100%

## Issues and Resolutions

**Description:** Based on the rules of the STEAK protocol a simple side chain should be added to allow easy consumption by third parties

* **Impact:**
* Minting policy only allows minting for specific token with latest random hash
* Accompanied by datum that describes the latest slot for which this has is valid
* Locked at script that only allows burning of token after some time (i.e. 1 day)
* **Resolution:** This can then be operated as a service for any chain consumer and allows for easy consumption of the random hash until 1 day after generation.