第 6 节： Working with the Python Standard Library

0 / 15 | 54 分钟

- 99. Module Introduction
  2 分钟
- 100. Exploring the Python Standard Library
  2 分钟
- 101. Importing Packages - Theory
  4 分钟
- 102. Importing "hashlib" to Create a Unique Hash
  7 分钟 资源
- 103. Blockchain Theory: Understanding Hashes
  1 分钟
- 104. Using Other Import Syntaxes
  2 分钟 资源
- 105. The "Proof of Work"
  7 分钟 资源
- 106. Blockchain Theory: Understanding the "Proof of Work"
  2 分钟
- 107. Adding the Proof of Work to our Blockchain
  6 分钟 资源
- 108. Including the Proof of Work in our Mining Function
  8 分钟
- 109. Fixing a Hash Order Fault
  6 分钟 资源
- 110. Splitting Up our Code
  5 分钟 资源
- 作业 5： Time to Practice - The Standard Library
- 111. Wrap Up
  2 分钟 资源
- 112. Useful Resources & Links
  1 分钟 资源

Intro

The Standard Library

Batteries Included

Module Overview

What about the Blockchain?

- Adding a Proof of Work for Mined Blocks
- Generating a “real Hash”

Exploring Standard Library

国内版 国际版

python standard library

网页 图片 视频 学术 词典 地图

检测到您输入了英文，试试切换到国际版？搜英文结果更丰富更准确

13,300,000 条结果 时间不限

Python教程 - 免费教程

来自 360 跟大家一起学习Python,从0基础走向精通,让你0压力学习,相互交流，相互学习Python在线免费学习交流

sem.tanzhouedu.com · 2019-6-1 · 广告

The Python Standard Library — Python 3.7.3 documentation 翻译此页

2019-5-27 · The Python Standard Library¶. While The Python Language Reference describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions. Python's standard library is very extensive, offering a …

https://docs.python.org/3/library

Keyword

This module allows a Python program to determine if a string is a keyword. …

Code

The Python Standard Library ... The code module provides facilities to…

PWD

The Python Standard Library » Unix Specific Services » | pwd — The…

Html

The Python Standard Library ... This function uses the rules defined by the…

Text Processing Services

The Python Standard Library » | Text Processing Services¶ The modules …

Unicodedata

This module provides access to the Unicode Character Database (UCD)…

从 docs.python.org 中搜索结果 搜索

相关搜索

- python standard library pdf
- python standard library 2.7
- the python standard library pdf
- python standard lib
- python standard error
- python 3 library
- python library reference
- the python standard library

国内版 国际版

python standard library

All Images Videos 翻译成中文 关闭取词

310,000 Results Any time

The Python Standard Library — Python 3.7.3 documentation

https://**docs.python.org**/3/**library**

The **Python Standard Library**¶. While The **Python** Language Reference describes the exact syntax and semantics of the **Python** language, this **library** reference manual describes the **standard library** that is distributed with **Python**. It also describes some of the optional components that are commonly included in **Python** distributions. **Python**'s **standard library** is very extensive, offering a wide range ...

Built-in Functions

ascii (object) ¶. As repr(), return a string containing a printable representation o…

Built-in Types

The Python Standard Library » | Built-in Types¶ The following sections describ…

Introduction

The Python Standard Library » | Introduction¶ The "Python library"…

Random

The Python Standard Library ... Python uses the Mersenne Twister as the core…

String

A primary use case for template strings is for internationalization (i18n) since i…

Webbrowser

The Python Standard Library » Internet Protocols and Support » | webbrowser …

Datetime

The standard library has timezone class for handling arbitrary fixed offsets from…

Search results from docs.python.org Search

The Python Standard Library — Python 2.7.16 documentation

https://**docs.python.org**/2/**library**

The **Python Standard Library**¶. While The **Python** Language Reference describes the exact syntax and semantics of the **Python** language, this **library** reference manual describes the **standard library** that is distributed with **Python**. It also describes some of the optional components that are commonly included in **Python** distributions. **Python**'s **standard library** is very extensive, offering a wide range ...

Python Standard Library

Book by Fredrik Lundh

Goodreads 3/5

Amazon 3.5/5

"Python Standard Library" is an essential guide for serious Python programmers. Python is a modular language that imports most useful operations from the standard library. You can't really program in Python without using it. In this book, author Fredrik Lund… +

Author: Fredrik Lundh

First published: May 20, 2001

Number of pages: 304

Genres: Reference · Computer Science · Programming · Technical

Get the book

Amazon Buy

Barnes and Noble Buy

Editorial reviews

amazon

Ideal for any working Python developer, Fredrik Lundh's Python Standard Library provides an excellent tour of some of the most important modules in today's Python 2.0 standard. Mixing sample code and plenty of expert advice, this title will be indispensable for programmers. The book presents sample script code--written by a frequent contribut… Read more

Read more Editorial Reviews at Amazon.com

Previous topic
10. Full Grammar specification

Next topic
Introduction

This Page
Report a Bug
Show Source

The Python Standard Library

While The Python Language Reference describes the exact syntax and semantics of the Python language, this library reference manual describes the standard library that is distributed with Python. It also describes some of the optional components that are commonly included in Python distributions.

Python's standard library is very extensive, offering a wide range of facilities as indicated by the long table of contents listed below. The library contains built-in modules (written in C) that provide access to system functionality such as file I/O that would otherwise be inaccessible to Python programmers, as well as modules written in Python that provide standardized solutions for many problems that occur in everyday programming. Some of these modules are explicitly designed to encourage and enhance the portability of Python programs by abstracting away platform-specifics into platform-neutral APIs.

The Python installers for the Windows platform usually include the entire standard library and often also include many additional components. For Unix-like operating systems Python is normally provided as a collection of packages, so it may be necessary to use the packaging tools provided with the operating system to obtain some or all of the optional components.

In addition to the standard library, there is a growing collection of several thousand components (from individual programs and modules to packages and entire application development frameworks), available from the Python Package Index.

- Numeric and Mathematical Modules
  - `numbers` — Numeric abstract base classes
  - `math` — Mathematical functions
  - `cmath` — Mathematical functions for complex numbers
  - `decimal` — Decimal fixed point and floating point arithmetic
  - `fractions` — Rational numbers
  - `random` — Generate pseudo-random numbers
  - `statistics` — Mathematical statistics functions

Table of Contents

Previous topic «

`random` — Generate pseudo-random numbers

Source code: Lib/random.py

This module implements pseudo-random number generators for various distributions.

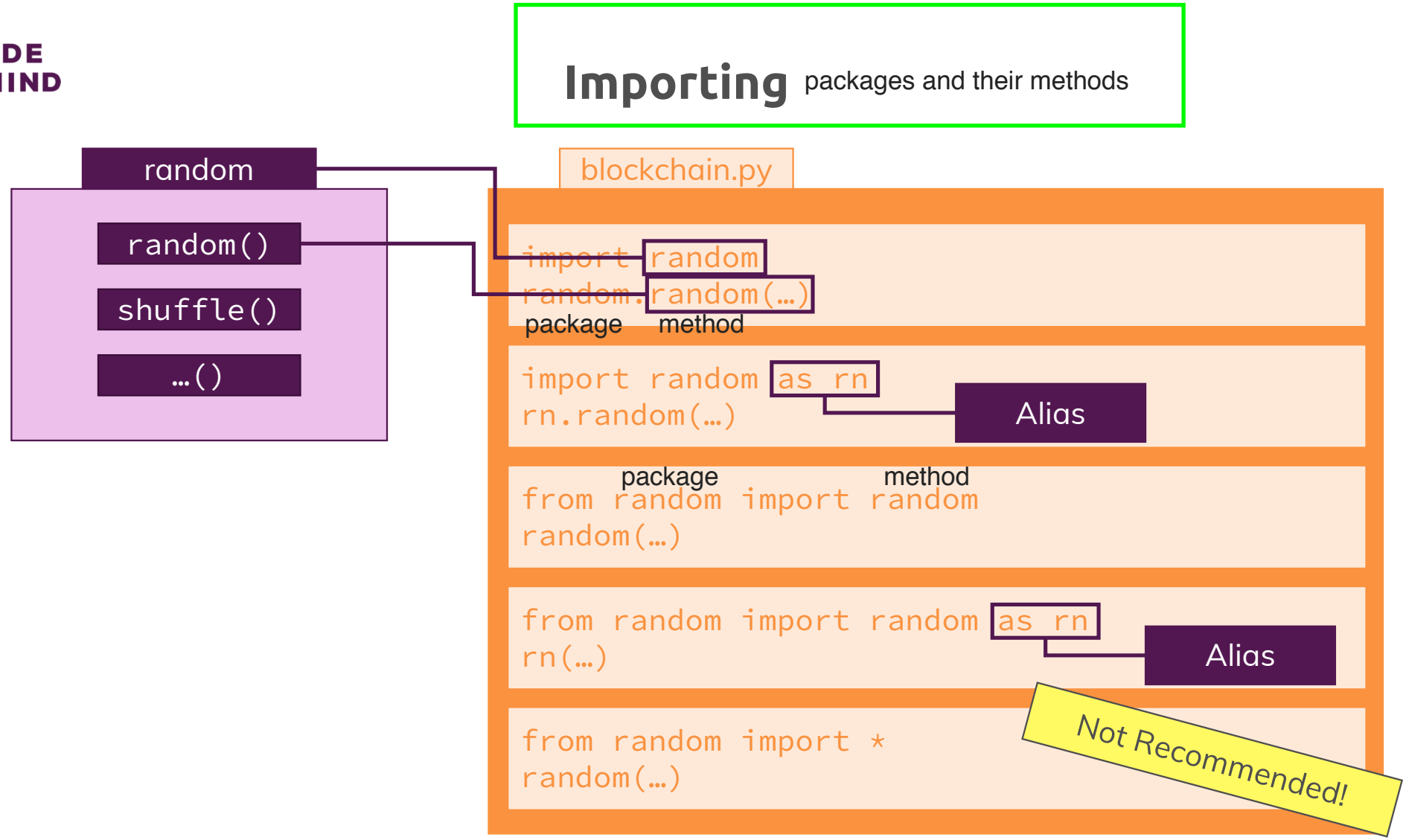
For integers, there is uniform selection from a range. For sequences, there is uniform selection of a random element, a function to generate a random permutation of a list in-place, and a function for random sampling without replacement.

On the real line, there are functions to compute uniform, normal (Gaussian), lognormal, negative exponential, gamma, and beta distributions. For generating distributions of angles, the von Mises distribution is available.

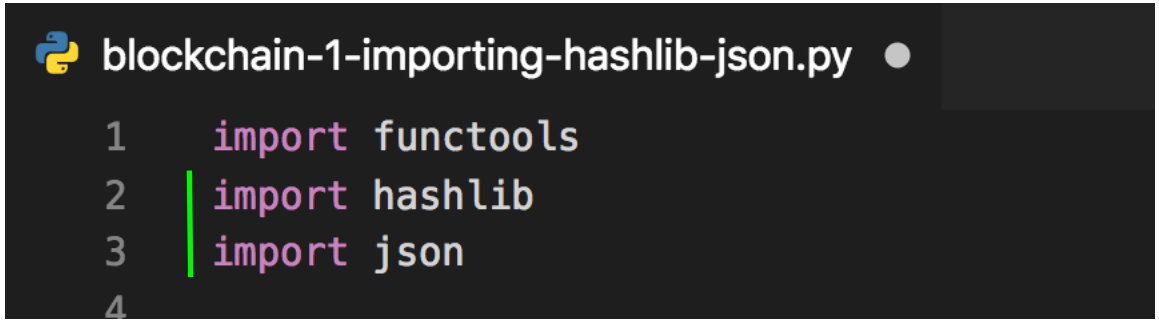
Almost all module functions depend on the basic function `random()`, which generates a random float uniformly in the semi-open range [0.0, 1.0). Python uses the Mersenne Twister as the core generator. It produces 53-bit precision floats and has a period of 2**19937-1. The underlying implementation in C is both fast and threadsafe. The Mersenne Twister is one of the most extensively tested random number generators in existence. However, being completely deterministic, it is not suitable for all purposes, and is completely unsuitable for cryptographic purposes.

ACADEMIND

Importing packages and their methods



Importing hashlib and json to create a unique hash



- hash的用途
  + same input, same output
  + cannot be translated back: cannot be reverse engineered for security reason
  + shorter than concatenating data

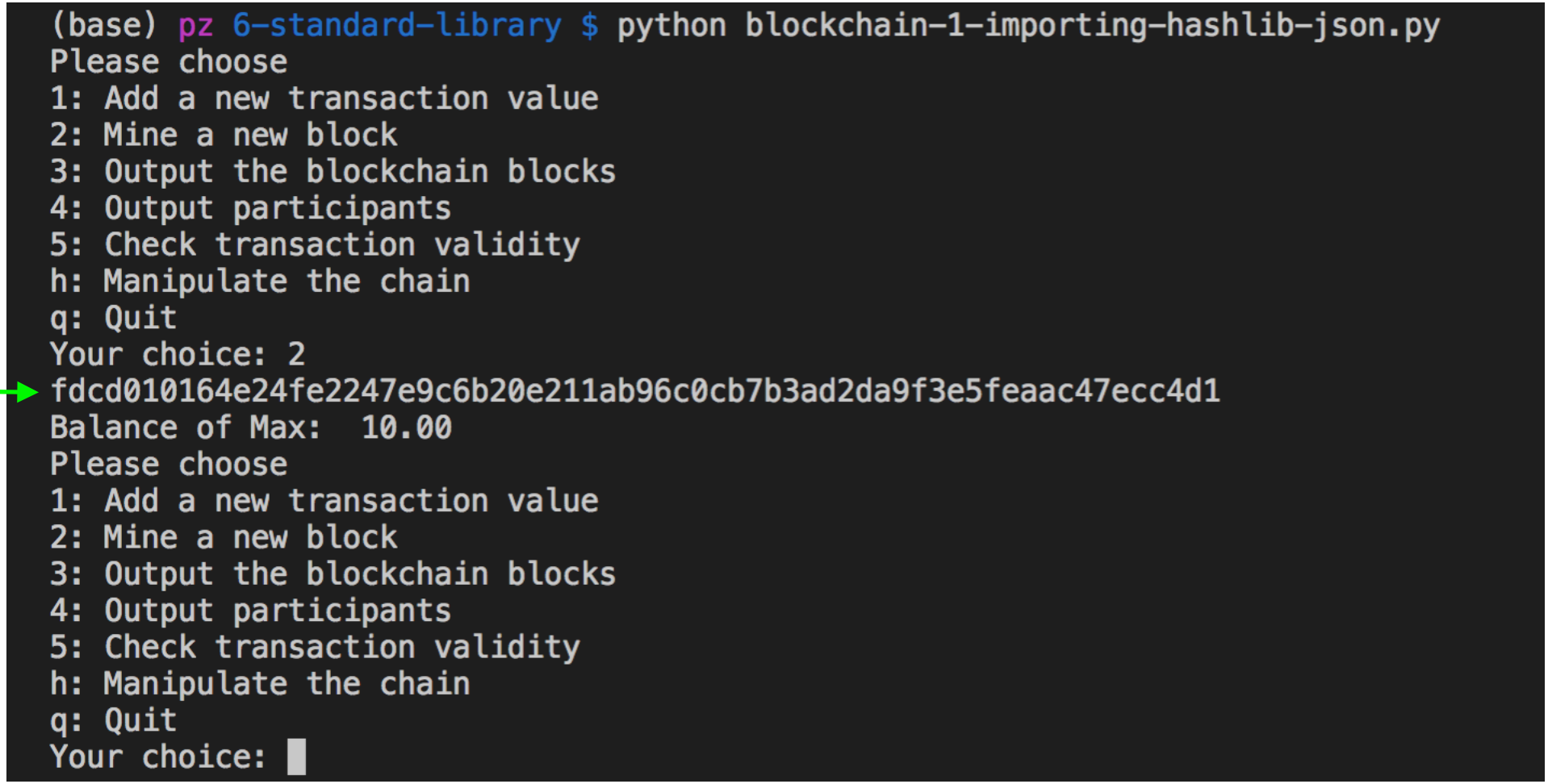
blockchain-1-importing-hashlib-json.py

```
23
24  def hash_block(block):
25      """Hashes a block and returns a string representation of it.
26
27      Arguments:
28          :block: The block that should be hashed.
29      """
30      return hashlib.sha256(json.dumps(block).encode()).hexdigest()
31
```

- json: encode object or complex data structures as a string
- json.dumps(block): json.dumps(dict): convert dictionary to string
- string.encode(): format string into utf-8, which can be utilized by sha256() method

- hashlib.sha256(utf-8 string) --> unique hash
- hashlib.sha256(utf-8 string).hexdigest() --> 十六进制字符串

```python
def mine_block():
    """Create a new block and add open transactions to it."""
    # Fetch the currently last block of the blockchain
    last_block = blockchain[-1]
    # Hash the last block (=> to be able to compare it to the stored hash value)
    hashed_block = hash_block(last_block)
    print(hashed_block)
    # Miners should be rewarded, so let's create a reward transaction
    reward_transaction = {
        'sender': 'MINING',
        'recipient': owner,
        'amount': MINING_REWARD
    }
    # Copy transaction instead of manipulating the original open_transactions list
    # This ensures that if for some reason the mining should fail, we don't have the
    copied_transactions = open_transactions[:]
    copied_transactions.append(reward_transaction)
    block = {
        'previous_hash': hashed_block,
        'index': len(blockchain),
        'transactions': copied_transactions
    }
    blockchain.append(block)
    return True
```


(base) pz 6-standard-library $ python blockchain-1-importing-hashlib-json.py
Please choose
1: Add a new transaction value
2: Mine a new block
3: Output the blockchain blocks
4: Output participants
5: Check transaction validity
h: Manipulate the chain
q: Quit
Your choice: 2
fdcd010164e24fe2247e9c6b20e211ab96c0cb7b3ad2da9f3e5feaac47ecc4d1
Balance of Max: 10.00
Please choose
1: Add a new transaction value
2: Mine a new block
3: Output the blockchain blocks
4: Output participants
5: Check transaction validity
h: Manipulate the chain
q: Quit
Your choice:

python3中digest()和hexdigest()区别

hashlib是涉及安全散列和消息摘要，提供多个不同的加密算法接口，如SHA1、SHA224、SHA256、SHA384、SHA512、MD5等。

其中

`hash.digest()`
返回摘要，作为二进制数据字符串值

`hash.hexdigest()`
返回摘要，作为十六进制数据字符串值

举个例子

```
import hashlib

md5 = hashlib.md5()
md5.update("a".encode('utf-8'))
print(u"digest返回的摘要：%s"% md5.digest())
print(u"hexdigest返回的摘要：%s"% md5.hexdigest())
```

结果

```
digest返回的摘要：b'\x0c\xc1u\xb9\xc0\xf1\xb6\xa81\xc3\x99\xe2iw&a'
hexdigest返回的摘要：0cc175b9c0f1b6a831c399e269772661
```

二进制
十六进制

Blockchain Theory: Understanding Hashes

The word "Hash" is omnipresent in the Blockchain world. It's important that you **a) understand what exactly a Hash is** and **b) understand why we use Hashes**.

a) What is a Hash?

To keep things simple, a Hash can simply be thought of as a long string (64 characters when using the SHA256 algorithm) that can **NOT be reverse engineered**. A hash is created based on some input values and the **same input values will always yield the exact same Hash**. So it's NOT a random value.

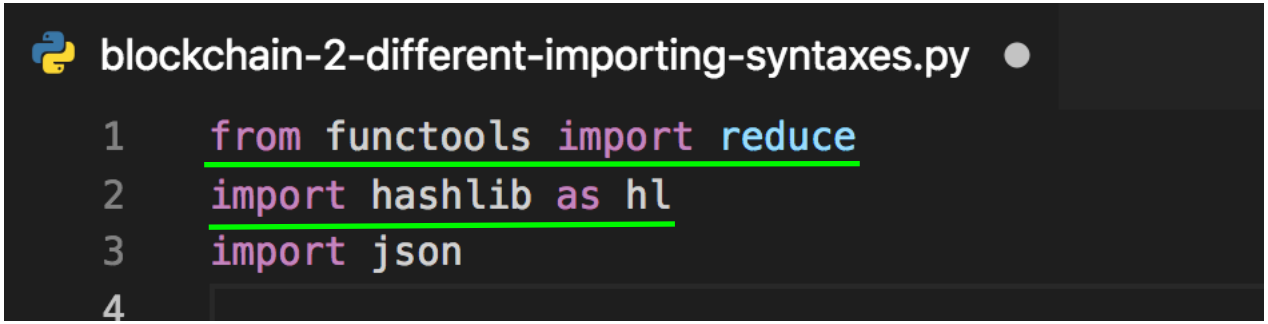
But you can't find out what the input was unless you have a Hash table - i.e. a map of inputs and resulting Hashes. That's of course something which is not easily created (brute force would be an option but you have billions of possible combinations - depending on the chosen Hashing algorithm).

b) Why do we use Hashes of Blocks instead of our "string concatenation" technique?

A Hash in our Blockchain has 64 characters. That's probably **shorter than a Block with dozens of Transactions** - the length of the Hash does not grow or shrink depending on the input value. So it's more efficient to store a Hash than to manually transform all data to string values.

We're not using a Hash because we want to hide the values. We could do this with a Hash but that's not the idea of a Blockchain. All the data should be publicly available, we don't want to hide the Block data - everyone should be able to check and validate the Blockchain.

Other importing sytaxes



```
23
24    def hash_block(block):
25        """Hashes a block and returns a string representation of it.
26
27        Arguments:
28            :block: The block that should be hashed.
29        """
30        return hl.sha256(json.dumps(block).encode()).hexdigest()
31
```

```
46                      for tx in open_transactions if tx['sender'] == participant]
47      tx_sender.append(open_tx_sender)
48      amount_sent = reduce(lambda tx_sum, tx_amt: tx_sum + sum(tx_amt)
49                           if len(tx_amt) > 0 else tx_sum + 0, tx_sender, 0)
50      # This fetches received coin amounts of transactions that were already included
51      # We ignore open transactions here because you shouldn't be able to spend coins
52      tx_recipient = [[tx['amount'] for tx in block['transactions']
53                       if tx['recipient'] == participant] for block in blockchain]
54      amount_received = reduce(lambda tx_sum, tx_amt: tx_sum + sum(tx_amt)
55                               if len(tx_amt) > 0 else tx_sum + 0, tx_recipient, 0)
56      # Return the total balance
57      return amount_received - amount_sent
58
```

```
pz Code $ cd 6-standard-library/
pz 6-standard-library $ python blockchain-2-different-importing-syntaxes.py
Please choose
1: Add a new transaction value
2: Mine a new block
3: Output the blockchain blocks
4: Output participants
5: Check transaction validity
h: Manipulate the chain
q: Quit
Your choice: 2
fdcd010164e24fe2247e9c6b20e211ab96c0cb7b3ad2da9f3e5feaac47ecc4d1
Balance of Max:  10.00
Please choose
1: Add a new transaction value
2: Mine a new block
3: Output the blockchain blocks
4: Output participants
5: Check transaction validity
h: Manipulate the chain
q: Quit
Your choice: 1
Enter the recipient of the transaction: Michael
Your transaction amount please: 5.5
Added transaction!
[{'sender': 'Max', 'recipient': 'Michael', 'amount': 5.5}]
Balance of Max:   4.50
Please choose
1: Add a new transaction value
2: Mine a new block
3: Output the blockchain blocks
4: Output participants
5: Check transaction validity
h: Manipulate the chain
q: Quit
Your choice: 2
fbc2da32f8291065778a94fae47324a59dd3c52391486ad41f532ca30fec1e00
Balance of Max:  14.50
Please choose
1: Add a new transaction value
2: Mine a new block
3: Output the blockchain blocks
4: Output participants
5: Check transaction validity
h: Manipulate the chain
q: Quit
Your choice: q
Balance of Max:  14.50
User left!
Done!
pz 6-standard-library $
```