mod Documentation

Release 0.3.0

Y. SOMDA

Sep 04, 2020

Contents:

1 Description 3

2 Usage 5

3 Package documentation: `mod.Mod` 7

4 Install 9

5 Links 11

6 Indices 13

Index 15

CHAPTER 1

Description

Modular arithmetic is arithmetic for integers, where numbers wrap around when reaching a given value called *modulus*. For example `6  1 (mod 5)`. Modular arithmetic has several practical applications including: music, banking, book publishing, cryptography... and of course math.

The purpose of this package is to simplify the use of modular arithmetic in **Python3**.

$$\pmod{n}$$

CHAPTER 2

Usage

This package provides `Mod` integers that compute arithmetic operations like `+ - * // **` with a modulus:

```
from mod import Mod

# Funny math here

x = Mod(5, 7)       # x  5 (mod 7)

(x + 2) == 0        # True: 5 + 2  7  0 (mod 7)
(x + 7) == x        # True: 5 + 7  12  5 (mod 7)
(x**3) == (x + 1)   # True: 5^3  125  6 (mod 7)
(1 // x) == 3       # True: 5 × 3  15  1 (mod 7)  5^1  3 (mod 7)
```

A naive implementation of RSA encryption algorithm using `mod` package:

```
from mod import Mod


# My RSA keys
public_key = Mod(3, 61423)
private_key = Mod(40619, 61423)

# My very secret message
top_secret_message = 666

# RSA encryption
encrypted = top_secret_message**public_key

# RSA decryption
decrypted = encrypted**private_key

# My secret message have been correctly encrypted and decrypted :-)
assert decrypted == top_secret_message
```

Note:

- `Mod` is based on integer modulo operation `%`, not `math.fmod`
- the result of an operation between a `Mod` and an `int` is a `Mod`
- the result of an operation between a `Mod` and a `float` is a `float`

CHAPTER 3

Package documentation: `mod.Mod`

class `mod.`**`Mod`**(*value*, *modulus*)

Integer number that automatically adds a modulus to arithmetic operations.

The floor division `//` implements the inverse of a multiplication with a modulus. Therefore, it should be used with care to avoid errors.

```
>>> number = Mod(2, 3)
>>> number
(2 % 3)
>>> quintuple = 5 * number
>>> quintuple // 5
(2 % 3)
>>>
```

Parameters

- **`value`** (*int*) – Mod number value
- **`modulus`** (*int*) – modulus associated with the value

Raises **`ValueError`** – one of the parameters is not a number or *modulus* == 0

`copy`(*modulus=None*)

Copy the Mod number

Parameters **`modulus`** (*int*) – modulus of the new Mod number

Return type *Mod*

`inverse`

Modular inverse of the number.

y is the inverse of **x** with the modulus **n** when:

$$yx1(mod.n)$$

Return type *Mod*

modulus

Modulus value

Return type int

CHAPTER 4

Install

Run the following command to install `mod` package

```
pip3 install mod
```

CHAPTER 5

Links

- Package documentation located at http://mod.readthedocs.io/en/latest/
- Python package available at https://pypi.python.org/pypi/mod
- Source code repository: https://github.com/yoeo/mod

CHAPTER 6

Indices

- genindex
- modindex
- search

mod — Copyright (c) 2020 Y. SOMDA, MIT License

Index

C

`copy()` (*mod.Mod method*), 7

I

`inverse` (*mod.Mod attribute*), 7

M

`Mod` (*class in mod*), 7
`modulus` (*mod.Mod attribute*), 8