mersenne-twister-predictor

Dec 29, 2020

Contents:

CHAPTER 1

mersenne-twister-predictor

1.1 mt19937predictor module

mt19937predictor.**LOWER_MASK = 2147483647**

mt19937predictor.**M = 397**

mt19937predictor.**MATRIX_A = 2567483615**

class mt19937predictor.**MT19937Predictor**

Bases: random.Random

Usage:

```
>>> import random
>>> from mt19937predictor import MT19937Predictor
>>> predictor = MT19937Predictor()
>>> for _ in range(624):
...     x = random.getrandbits(32)
...     predictor.setrandbits(x, 32)
>>> random.getrandbits(32) == predictor.getrandbits(32)
True
>>> random.random() == predictor.random()
True
>>> a = list(range(100))
>>> b = list(range(100))
>>> random.shuffle(a)
>>> predictor.shuffle(b)
>>> a == b
True
```

gauss(**args*)

Raises NotImplementedError

genrand_int32()

getrandbits(*bits*)
The interface for `random.Random.getrandbits()` in Python's Standard Library

getstate(**args*)

Raises `NotImplementedError`

random()
The interface for `random.Random.random()` in Python's Standard Library

seed(**args*)

Raises `NotImplementedError`

setrand_int32(*y*)
Feceive the target PRNG's outputs and reconstruct the inner state. when 624 consecutive DOWRDs is given, the inner state is uniquely determined.

setrandbits(*y*, *bits*)
The interface for `random.Random.getrandbits()` in Python's Standard Library

setstate(**args*)

Raises `NotImplementedError`

`mt19937predictor.`**N = 624**
624 values (of 32bit) is just enough to reconstruct the internal state

`mt19937predictor.`**UPPER_MASK = 2147483648**

`mt19937predictor.`**generate**(*mt*, *kk*)

`mt19937predictor.`**genrand_int32**(*mt*, *mti*)

`mt19937predictor.`**tempering**(*y*)

`mt19937predictor.`**untempering**(*y*)

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

m

Index