towers Documentation

Release 0.1.1

Francis Horsman

Jan 13, 2020

Main modules:

The **'Towers of Hanoi'** algorithm.

CHAPTER 1

Towers

class towers.core.towers.**Towers**(*height=1*, *rods=None*, *moves=0*, *verbose=False*)

A representation of the towers including all logic.

class JsonEncoder(*skipkeys=False*, *ensure_ascii=True*, *check_circular=True*, *allow_nan=True*, *sort_keys=False*, *indent=None*, *separators=None*, *encoding='utf-8'*, *default=None*)

default(*obj*)

Implement this method in a subclass such that it returns a serializable object for `o`, or calls the base implementation (to raise a `TypeError`).

For example, to support arbitrary iterators, you could implement default like this:

```
def default(self, o):
    try:
        iterable = iter(o)
    except TypeError:
        pass
    else:
        return list(iterable)
    # Let the base class default method raise the TypeError
    return JSONEncoder.default(self, o)
```

__bool__()

A Towers is considered True if it's state is completed.

Return type bool

__call__()

Run the towers. Convenience method.

Raises See *Towers.move_tower()*.

__contains__(*x*)

Does this *Towers* contain the given `Rod`.

Parameters **x** (`Rod`) – The `Rod` to find.

Return type bool

__copy__()
Return a shallow copy of this instance.

Return type *Towers*

__deepcopy__(**d*)
Return a deep copy of this instance.

Parameters **d** (*dict*) – Memoisation dict.

Return type *Towers*

__enter__()
Context-Manager entry, validate our entry state for towers-start conditions.

Raises See *Towers.validate_start()*.

__eq__(*other*)
Compare Towers instances for equivalence.

Parameters **other** (Towers) – The other *Towers* to compare.

Return type bool

__exit__(**args*, ***kwargs*)
Context-Manager exit, validate our exit state for towers-end conditions.

Raises See *Towers.validate_end()*.

__getitem__(*index*)
Get the Rod at the given index.

Parameters **index** (*int*) – The index to get the Rod at.

Return type *Rod*

__init__(*height=1*, *rods=None*, *moves=0*, *verbose=False*)

Parameters

- **height** (*int*) – The height of the towers (ie: max number of disks each one rod can hold).
- **rods** (Rods) – An existing Rods instance to use with this *Towers* (the heights must match).
- **moves** (*int*) – The number of moves already taken.
- **verbose** – True=enable verbose logging mode.

__iter__()
Run the towers, yielding Move instances.

__len__()
Determine how many Rod's this *Towers* contains.

Return type int

__nonzero__()
A Towers is considered non-zero if it's state is completed.

Return type bool

context(***kwds*)
Create a temp context for performing moves. The state of this instance will be reset at context exit.

Parameters

- **reset_on_success** (*bool*) – Reset this instance's state on exit from the context if no error occurred. Default = True.
- **reset_on_error** (*bool*) – Reset this instance's state on exit from the context if an error occurred. Default = False.

end_rod
Retrieve the end `Rod` for this towers.

Return type *Rod*

classmethod from_json(*d*)
Return a class instance from a json serializable representation.

Parameters **d** (*str | dict*) – The json or decoded-json from which to create a new instance.

Return type *Towers*

Raises See *Towers*.__new__.

height
Obtain the height of the *Towers* (ie: max number of disks each one rod can hold).

Return type int

move_disk(*start*, *end*)
Move the *Disk* from one Rod to another.

Note Generator, yields *Move* instances.

Parameters

- **start** (`Rod`) – The `Rod` to remove the `Disk` from.
- **end** (`Rod`) – The `Rods` to move the `Disk` to.

move_tower(*height*, *start*, *end*, *tmp*)
Move the stack of *Disks* on a *Rod*.

Parameters

- **height** (*int*) – The height of the `Disk` to move.
- **start** (`Rod`) – The `Rod` to move the `Disk` from.
- **end** (`Rod`) – The `Rod` to move the `Disk` to.
- **tmp** (`Rod`) – The intermediary `Rod` to use when moving the `Disk`.

moves
Determine how many moves have occurred so far.

Return type int

static moves_for_height(*height*)
Determine the max number of moves required to solve the puzzle for the given height

Parameters **height** (*int*) – The height of the `Rods` (number of `Disk` on a `Rod`).

Return type int

start_rod
Retrieve the start `Rod` for this towers.

Return type *Rod*

tmp_rod
Retrieve the temporary `Rod` for this towers.

Return type *Rod*

to_json()
Return a json serializable representation of this instance.

Return type object

validate()
Perform self validation.

Raises

- ***InvalidTowerHeight*** – The height of the tower is invalid
- ***DuplicateDisk*** – This `Rod` already contains this `Disk`.
- ***CorruptRod*** – A `Disk` is on top of a `Disk` of smaller size.

validate_end()
Validate the end conditions for this towers.

Raises

- ***InvalidTowerHeight*** – The height of the tower is invalid
- ***DuplicateDisk*** – This `Rod` already contains this `Disk`.
- ***CorruptRod*** – A `Disk` is on top of a `Disk` of smaller size.
- ***InvalidEndingConditions*** – End conditions are invalid.

validate_start()
Validate the start conditions for this towers

Raises

- ***InvalidTowerHeight*** – The height of the *Towers* is invalid
- ***DuplicateDisk*** – This `Rod` already contains this `Disk`.
- ***CorruptRod*** – A `Disk` is on top of a `Disk` of smaller size.
- ***InvalidStartingConditions*** – Initial conditions are invalid.

verbose
Obtain this instance's verbose flag.

Return type bool

CHAPTER 2

Rods

Rods is a collection of **Rod**'s, one representing the *start*, *end* and *intermediary* rods for the tower.

class towers.core.rods.**Rods**

A collection of 3 Rod's that form the Tower.

Parameters

- **start** (Rod) – The rod containing the disks at their start position.
- **end** (Rod) – The rod containing the disks at their end position.
- **tmp** (Rod) – The intermediary rod.
- **height** (*int*) – The height of the tower.

Raises

- ***InvalidTowerHeight*** – The height of the tower is invalid.
- ***InvalidRod*** – A rod is not of expected type *Rod*.
- ***InvalidRodHeight*** – A rod height is inconsistent with the specified height.
- ***DuplicateDisk*** – A rod contains a duplicate disk
- ***CorruptRod*** – A disk is on top of a disk of smaller size on a Rod.

__bool__()

A Rods is considered True if it contains any disks on any rods.

Return type bool

__copy__()

Return a shallow copy of this instance.

Return type *Rods*

__deepcopy__(**a*)

Return a deep copy of this instance.

Return type *Rods*

__iter__()
Iterate over all the rods.

Return type *Rod*

__len__()
Obtain the number of Rods.

Return type int

__nonzero__()
A Rods is considered non-zero if it contains any disks on any rods.

Return type bool

classmethod from_json(*d*)
Return a class instance from a json serializable representation.

Parameters **d** (*str | dict*) – The json or decoded-json from which to create a new instance.

Return type *Rods*

Raises See *Rods*.__new__.

height
Retrieve the height of the rods (ie: max number of disks each one can hold).

Return type int

to_json()
Return a json serializable representation of this instance.

Return type object

validate()
Perform self validation.

Raises

- ***DuplicateDisk*** – This rod already contains this disk
- ***CorruptRod*** – A disk is on top of a disk of smaller size.

CHAPTER 3

Rod

Note: A tower that contains **Disks**.

class towers.core.rod.**Rod**

A single tower containing disks.

__bool__()

A Rod is considered True if it contains any disks.

Return type bool

__copy__()

Return a shallow copy of this instance.

Return type *Rod*

__deepcopy__(**d*)

Return a deep copy of this instance.

Parameters **d** (*dict*) – Memoisation dict.

Return type *Rod*

__eq__(*other*)

Compare Rod instances for equivalence.

Parameters **other** (Rod) –

Return type bool

__iter__()

Iterate over all the disks in this rod.

Return type *Disk*

__len__() <==> *len(x)*

static **__new__**(*cls*, *name*, *disks=None*, *height=0*)

Parameters

- **name** (*str*) – The name of the rod.
- **disks** (*List[Disk]*) – (optional) mutatable list of *Disks*.
- **height** (*int*) – The height of the rod.

Return type *Rod*

Raises See *Rod.validate*.

__nonzero__()
A Rod is considered non-zero if it contains any disks.

Return type bool

append(*disk*, *validate=True*)
Append the disk to this rod and optionally validate.

Parameters

- **disk** (*Disk*) – The disk to add to the top of our rod.
- **validate** (*bool*) – True=perform self validation.

classmethod from_json(*d*)
Return a class instance from a json serializable representation.

Parameters **d** (*Union[str,dict]*) – The json or decoded-json from which to create a new instance.

Return type *Rod*

Raises See *Rod.__new__*.

pop()
Pop the top most disk from this rod and return it

Return type *Disk*

to_json()
Return a json serializable representation of this instance.

Return type object

validate()
Perform self validation.

Raises

- ***DuplicateDisk*** – This rod already contains this disk
- ***CorruptRod*** – A disk is on top of a disk of smaller size.
- ***InvalidTowerHeight*** – The height of the tower is invalid.
- ***InvalidDiskPosition*** – The position of the disk is invalid.

CHAPTER 4

Disk

A disk is a sized element on a *Rod* where: **1 <= size <= rod_height**

class towers.core.disk.**Disk**

An immutable representation of a sized disk that sits on a *Rod*.

static **__new__**(*cls*, *original_position*, *height=1*)

Parameters

- **original_position** (*int*) – The position on the *Rod* that this disks originally sat. Zero = The bottom of the *Rod*.
- **height** (*int*) – The maximum position of this *Disk* on a Rod.

Return type *Disk*

Raises

- ***InvalidTowerHeight*** – The height of the tower is invalid.
- ***InvalidDiskPosition*** – The position of the disk is invalid.

classmethod **from_json**(*d*)

Return a class instance from a json serializable representation.

Parameters **d** (*str | dict*) – The json or decoded-json from which to create a new instance.

Return type *Disk*

Raises See *Disk.__new__*.

to_json()

Return a json serializable representation of this instance.

Return type object

validate()

Perform self validation

Raises

- ***InvalidTowerHeight*** – The height of the tower is invalid.
- ***InvalidDiskPosition*** – The position of the disk is invalid.

width
Obtain the width of the disk

Return type int

CHAPTER 5

Errors and Utils

Any error explicitly raised by *towers* is defined here.

exception towers.core.errors.**InvalidRod**(*rod*)

__init__(*rod*)

Parameters **rod** (*object*) – The Rod which is invalid.

exception towers.core.errors.**InvalidRods**(*rods*)

__init__(*rods*)

Parameters **rods** (*object*) – The Rods which are invalid

exception towers.core.errors.**InvalidRodHeight**(*rod*, *max_height*)

__init__(*rod*, *max_height*)

Parameters

- **rod** (Rod) – The Rod which has an invalid height.
- **max_height** (*int*) – The max allowed height of the Rod.

exception towers.core.errors.**DuplicateDisk**(*rod*, *disk_width*)

A duplicate disk was found on a tower.

__init__(*rod*, *disk_width*)

Parameters

- **rod** (Rod) – The duplicate Rod.
- **disk_width** (*int*) – The width of the Disk.

exception towers.core.errors.**CorruptRod**(*rod*, *disk*)

A Rod with an invalid stack of disks was found.

__init__(*rod*, *disk*)

Parameters

- **rod** (Rod) – The Rod which is corrupt.
- **disk** (*int*) – A Disk which sits directly atop a smaller Disk.

exception towers.core.errors.**InvalidStartingConditions**(*rods*, *moves*)
The Rods for the towers are not in the correct starting state.

__init__(*rods*, *moves*)

Parameters

- **rods** (Rod) – The Rod's.
- **moves** (*int*) – Total number of moves already made (should be zero).

exception towers.core.errors.**InvalidEndingConditions**(*rods*)
The Rod's for the towers are not in the correct ending state.

__init__(*rods*)

Parameters **rods** (Rod) – The Rod's.

exception towers.core.errors.**InvalidTowerHeight**(*height*)
The height of the Tower is invalid.

__init__(*height*)

Parameters **height** (*int*) – The invalid height.

exception towers.core.errors.**InvalidDiskPosition**(*position*, *height*)
The position of the Disk is invalid.

__init__(*position*, *height*)

Parameters

- **position** (*int*) – The invalid position on the Rod.
- **height** (*int*) – The height.

exception towers.core.errors.**InvalidMoves**(*moves*)
An invalid number of moves.

__init__(*moves*)

Parameters **moves** (*int*) – The invalid *moves*.

Note: Main *towers.core.utils.Serializable* is used by all main classes: Towers, Rods, Rod, Disk

---

class towers.core.utils.**Serializable**
A mixin which shows that a class is serializable.

from_json(*d*)
Return a class instance from a json serializable representation.

Parameters **d** (*str|dict*) – The json or decoded-json from which to create a new instance from.

to_json()
Return a json serializable representation of this instance.

Return type object

CHAPTER 6

Validation

Note: These methods are used internally, but there's no reason they can't be used externally.

towers.core.validation.**validate_height**(*height*)
Validate the height of a Tower`s or :class:`Rod.

Parameters **height** (*int*) – The height to validate.

Raises ***InvalidTowerHeight*** – The height of the Tower is invalid.

towers.core.validation.**validate_rods**(*rods*)
Validate the rods.

Parameters **rods** (*List[Rod]|None*) – The Rod's to validate.

Raises

- ***InvalidRods*** – expecting type Rods.
- ***DuplicateDisk*** – This Rod already contains this Disk
- ***CorruptRod*** – A Disk is on top of a Disk of smaller size.

towers.core.validation.**validate_moves**(*moves*)
Validate the number of moves.

Parameters **moves** (*int*) – The moves count to validate.

Raises ***InvalidMoves*** – The number of moves is not an number or is less than zero.

CHAPTER 7

Moves

Note: When the Towers is iterated over, a series of **Move**'s are yielded.

class towers.core.moves.**Move**

Parameters

- **disk** (Disk) – The disk that will be moved.
- **start** (Rod) – The state of the start_rod prior to the move.
- **end** (Rod) – The state of the end_rod prior to the move.
- **moves** (*int*) – The number of moves prior to the move.

CHAPTER 8

Example

```
>>> tower = Towers(height=3)
>>> print(tower)
Towers(Rods(3 - start([***, **, *]), end([]), tmp([])))

>>> print('moves required: {moves}'.format(moves=tower.moves_for_height(height)))
moves required: 7

>>> with tower:
...     for i in tower:
...         print(i)
Move(disk=*, start=Rod(name='start', disks=[***, **, *], height=3), end=Rod(name='end
↪', disks=[], height=3), moves=0)
Move(disk=**, start=Rod(name='start', disks=[***, **], height=3), end=Rod(name='tmp',␣
↪disks=[], height=3), moves=1)
Move(disk=*, start=Rod(name='end', disks=[*], height=3), end=Rod(name='tmp',␣
↪disks=[**], height=3), moves=2)
Move(disk=***, start=Rod(name='start', disks=[***], height=3), end=Rod(name='end',␣
↪disks=[], height=3), moves=3)
Move(disk=*, start=Rod(name='tmp', disks=[**, *], height=3), end=Rod(name='start',␣
↪disks=[], height=3), moves=4)
Move(disk=**, start=Rod(name='tmp', disks=[**], height=3), end=Rod(name='end',␣
↪disks=[***], height=3), moves=5)
Move(disk=*, start=Rod(name='start', disks=[*], height=3), end=Rod(name='end',␣
↪disks=[***, **], height=3), moves=6)

>>> print(tower)
Towers(Rods(3 - start([]), end([***, **, *]), tmp([])))

>>> print('moves taken: {moves}'.format(moves=tower.moves))
moves taken: 7
```

CHAPTER 9

Installation

Instructions can be found here

CHAPTER 10

Contributions

Guidelines can be found here

Authors can be found here

CHAPTER 11

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

Index

Symbols

A

C

D

E

F