Unblock Me Solver

API Documentation

July 23, 2017

Contents

1 Package UnBlockMeSolver

1.1 Modules

1.2 Variables

Name	Description
__package__	**Value:** `None`

2 Package UnBlockMeSolver.Map

2.1 Modules

- **Map** *(Section 3, p. 4)*
- **MapReader** *(Section 4, p. 9)*
- **Move** *(Section 5, p. 11)*
- **Piece** *(Section 6, p. 14)*

2.2 Variables

Name	Description
__package__	**Value:** `None`

3 Module UnBlockMeSolver.Map.Map

3.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.Map'`

3.2 Class Map

object
└ **UnBlockMeSolver.Map.Map.Map**

3.2.1 Methods

__init__(*self*, *graph*, *delimeter='\n'*)

Initialize the class by setting the graph and delimeters variables.

Parameters

`graph:` Represents the board that the game will be played on
(type=string | MapReader)

`delimeter:` how to split the string
(type=string)

Overrides: object.__init__

convertToMap(*self*)

Convert the graph into a matrix that can be viewed and modified.

setUpPieces(*self*)

Find every piece available in the board and add it to an array to keep track of them for future use.

setUp(*self*)

Set up the Map class and set the graph by either getting a a string from either a map reader or string as the graph and breaking if not valid.

isWallOrGoal(*self*, *point*, *num_goals_found*)

This will check if the point is a wall or goal and if not return False. Additionally, it will update the number of goals that have been found.

Parameters

`point:` the point being checked in the graph.
(type=character)

`num_goals_found:` number of goals in the graph currently found
(type=integer)

Return Value

if the point is a wall or a goal and number of goals currently found
(type=boolean, integer)

largeRowValid(*self*, *row_index*, *num_goals_found*)

This will test on the larger rows to see if it is valid. A larger row indicates that it is either the top or the bottom of the graph.

Parameters

`row_index:` index for the row to be checked
(type=integer)

`num_goals_found:` the number of goals that have been found
(type=integer)

Return Value

If a large row provided is valid and number of goals currently found
(type=boolean, integer)

numColumnsMatch(*self*)

Ensure that the number of columns matches for each row.

Return Value

if the number of columns matches for each row
(type=boolean)

topBottomRowsValid(*self*, *num_goals_found*)

Make sure that both the top and bottom rows are valid according to the topBottomRowsValid function.

Parameters

`num_goals_found:` number of goals presently found
(type=integer)

Return Value

If the top and bottom rows are valid and number of goals currently found
(type=boolean, integer)

midRowsValid(*self*, *num_goals_found*)

Ensure that all the middle rows are valid entries with walls or a goal on both sides.

Parameters

`num_goals_found:` number of goals presently found

(type=integer)

Return Value

if the middle rows are valid and the number of goals currently found

(type=boolean, integer)

playerFound(*self*)

Check to see if the player, defined in Map.py, is in the map.

Return Value

player found in map

(type=boolean)

isValid(*self*)

Test if the graph in the map class is valid or not.

Return Value

graph is valid or not

(type=boolean)

validAdditionMoves(*self*, *move*)

This will handle move verificatin for moving to the right or down in the board. It will return an array of the the moves that can be made in the direction given.

Parameters

`move:` Array of moves in the direction of the move given

(type=[Move])

validSubtractionMoves(*self*, *move*)

This will handle move verificatin for moving to the left or up in the board. It will return an array of the the moves that can be made in the direction given.

Parameters

`move:` Array of moves in the direction of the move given

(type=[Move])

isValidMove(*self*, *move*)

Check if the move given is valid or not to make the move.

Return Value

if the move is valid or not

(type=boolean)

makeConfidentMove(*self*, *move*)

Make a move on the board without checking for its validity.

Parameters

`move:` move to make on the board

(type=Move)

makeMove(*self*, *move*)

Make a move on the board. Raise syntax error on bad move given

Parameters

`move:` move to make on the board

(type=Move)

getMoves(*self*)

Get the moves available on the board.

Return Value

Array of moves available

(type=[Move])

isSolved(*self*)

If there is a goal found then the board is not solved. Complete checking is done as there may be a goal in any location.

Return Value

if the game has been solved or not

(type=boolean)

copy(*self*)

Create a copy of this map and return it

Return Value

Copy of this map

(type=Map)

copyMove(*self*, *move*)

Create a copy of the map and make a move on it; return the result.

Parameters

`move:` move to be bade on copied board

(type=Move)

Return Value

The new map with the move made on it

(type=Map)

copyConfidentMove(*self*, *move*)

Create a copy of the map and make a move on it without checking for its validity. Return the result.

Parameters

`move:` move to be bade on copied board

(type=Move)

Return Value

The new map with the move made on it

(type=Map)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

3.2.2 Properties

Name	Description
Inherited from object	
__class__	

3.2.3 Class Variables

Name	Description
wall	**Value:** `'|'`
goal	**Value:** `'$'`
empty	**Value:** `'0'`
player	**Value:** `'**'`
playerPiece	**Value:** `'*'`

4 Module UnBlockMeSolver.Map.MapReader

4.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.Map'`

4.2 Class MapReader

object
└ **UnBlockMeSolver.Map.MapReader.MapReader**

4.2.1 Methods

__init__(*self*)

Initialize MapReader class

Overrides: object.__init__

load(*self*, *file_name*)

Set flags and test if the file given exists. Throws IOError on file not existing.

Parameters

`file_name`: Name of the file that contains definition of the map.
(type=string)

get(*self*)

Read the file and return the string inside

Return Value

string inside of the specified file
(type=string)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

4.2.2 Properties

Name	Description
Inherited from object	
__class__	

4.2.3 Class Variables

Name	Description
loaded	**Value:** `False`
file_found	**Value:** `False`

5 Module UnBlockMeSolver.Map.Move

5.1 Variables

Name	Description
__package__	**Value:** `None`

5.2 Class Move

object
└ **UnBlockMeSolver.Map.Move.Move**

5.2.1 Methods

__init__(*self*, *piece*, *right*, *up*)

Initialize data structure with required variables. Throws an error on incorrect value ranges and types given

Parameters

`piece`: The piece to be moved
(type=character)

`right`: If the piece should be to the right by a space
(type=integer)

`up`: If the piece should be moved up by a space
(type=integer)

Overrides: object.__init__

isValid(*self*)

Ensure that the move is valid for the board and the piece is a character.

equals(*self*, *move*)

Test whether the two moves are equal.

Parameters

`move`: Move being tested for equality

(type=Move)

Return Value

Whether the two moves are equal

(type=boolean)

toStr(*self*)

Convert self to string.

Return Value

String representation of the move

(type=str)

left(*piece*, *size*=`1`)

Instantiate a move that will go left

Parameters

`piece`: Character that represents the piece to be moved

(type=character)

right(*piece*, *size*=`1`)

Instantiate a move that will go right

Parameters

`piece`: Character that represents the piece to be moved

(type=character)

up(*piece*, *size*=`1`)

Instantiate a move that will go up

Parameters

`piece`: Character that represents the piece to be moved

(type=character)

down(*piece*, *size*=1)

Instantiate a move that will go down

Parameters

`piece`: Character that represents the piece to be moved

(type=character)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

5.2.2 Properties

Name	Description
Inherited from object	
__class__	

6 Module UnBlockMeSolver.Map.Piece

6.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.Map'`

6.2 Class Piece

object
└ **UnBlockMeSolver.Map.Piece.Piece**

6.2.1 Methods

__init__(*self*, *x*, *y*, *vertical*)

Initialize piece class

Parameters

`x:` x coordinate
(type=integer)

`y:` y coordinate
(type=integer)

`vertical:` marks whether this piece can move vertically or not
(type=boolean)

Overrides: object.__init__

move(*self*, *move*)

Move the piece if the given move is valid

Parameters

`move:` Move that will change the location of the piece
(type=Move)

copy(*self*)

Create a copy of self.

Return Value

Copy of this piece

(type=Piece)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

6.2.2 Properties

Name	Description
Inherited from object	
__class__	

7 Package UnBlockMeSolver.PathFinder

7.1 Modules

7.2 Variables

Name	Description
__package__	**Value:** `None`

8 Module UnBlockMeSolver.PathFinder.AStar

8.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.PathFinder'`

8.2 Class AStar

object
└
UnBlockMeSolver.PathFinder.PathFinder.PathFinder
└
UnBlockMeSolver.PathFinder.AStar.ASta

8.2.1 Methods

populateQueue(*self*, *graph*, *queue*, *parent*)

Populate the priority queue with nodes and their cost.

Parameters

`graph`: Graph to get moves from to populate queue
(type=Map)

`queue`: Queue with nodes being added to it with the cost
(type=Queue)

`parent`: Parent for the nodes that will be created
(type=AStarNode)

getPath(*self*, *heuristic*)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle
(type=[Move])

Overrides: UnBlockMeSolver.PathFinder.PathFinder.PathFinder.getPath

Inherited from UnBlockMeSolver.PathFinder.PathFinder.PathFinder(Section 12.2)

__init__()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

8.2.2 Properties

Name	Description
Inherited from object	
__class__	

9 Module UnBlockMeSolver.PathFinder.AStarNode

9.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.PathFinder'`

9.2 Class AStarNode

object
└
UnBlockMeSolver.PathFinder.Node.Node
└
UnBlockMeSolver.PathFinder.AStarNode.AStarNod

9.2.1 Methods

__init__(*self*, *graph*, *move*, *parent*, *g*, *h*)

Initialize AStarNode class

Parameters

`graph`: Map that moves can be made on
(type=Map)

`move`: Move that resulted in the above graph
(type=Move)

`parent`: Parent node of this. Contains previous moves to get the current board.
(type=AStarNode)

`g`: Step cost for this node
(type=number)

`h`: Heuristic cost for this node
(type=number)

Overrides: object.__init__

cost(*self*)

Cost for this node.

Return Value

The cost to get to this state

(type=number)

Inherited from UnBlockMeSolver.PathFinder.Node.Node(Section 11.2)

reconstructPath()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

9.2.2 Properties

Name	Description
Inherited from object	
__class__	

9.2.3 Class Variables

Name	Description
g	**Value:** `0`
h	**Value:** `0`

10 Module UnBlockMeSolver.PathFinder.Heuristics

10.1 Functions

manhattan(*x1*, *y1*, *x2*, *y2*)

euclidian(*x1*, *y1*, *x2*, *y2*)

10.2 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.PathFinder'`

11 Module UnBlockMeSolver.PathFinder.Node

11.1 Variables

Name	Description
__package__	**Value:** `None`

11.2 Class Node

object
└ **UnBlockMeSolver.PathFinder.Node.Node**

Known Subclasses: UnBlockMeSolver.PathFinder.AStarNode.AStarNode

11.2.1 Methods

__init__(*self*, *graph*, *move*, *parent*)

Initialize AStarNode class

Parameters

`graph:` Map that moves can be made on
(type=Map)

`move:` Move that resulted in the above graph
(type=Move)

`parent:` Parent node of this. Contains previous moves to get the current board.
(type=Node)

Overrides: object.__init__

reconstructPath(*self*)

Construct the path from here that it took to make it here.

Return Value

array of moves to get to this point from the root node
(type=[Move])

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

11.2.2 Properties

Name	Description
Inherited from object	
__class__	

12 Module UnBlockMeSolver.PathFinder.PathFinder

12.1 Variables

Name	Description
__package__	**Value:** `None`

12.2 Class PathFinder

object
└ **UnBlockMeSolver.PathFinder.PathFinder.PathFinder**

Known Subclasses: UnBlockMeSolver.PathFinder.TreeSearch.TreeSearch, UnBlockMeSolver.PathFinder.AStar.AStar, UnBlockMeSolver.PathFinder.dfs.DFS, UnBlockMeSolver.PathFinder.bfs

12.2.1 Methods

__init__(*self*, *board*)

Initialize Pathfinder class instance.

Parameters

`board`: board to be solved

(type=Map)

Overrides: object.__init__

getPath(*self*)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle

(type=[Move])

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

12.2.2 Properties

Name	Description
Inherited from object	
__class__	

13 Module UnBlockMeSolver.PathFinder.Solver

13.1 Variables

Name	Description
__package__	**Value:** `None`

14 Module UnBlockMeSolver.PathFinder.TreeSearch

14.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.PathFinder'`

14.2 Class TreeSearch

object

UnBlockMeSolver.PathFinder.PathFinder.PathFinder

UnBlockMeSolver.PathFinder.TreeSearch.

14.2.1 Methods

__init__(*self*, *board*, *bfs*=`True`)

Initialize TreeSearch class.

Parameters

`board`: Map to be solved

(type=Map)

`bfs`: Whether this class will use breadth-first search or depth-first search

(type=boolean)

Return Value

Initialized TreeSearch class

(type=TreeSearch)

Overrides: object.__init__

populateQueue(*self*, *graph*, *move*, *queue*, *parent*)

Populate the queue with nodes and their cost.

Parameters

`graph`: Graph to get moves from to populate queue
(type=Map)

`move`: Move to get to the
(type=Move)

`queue`: Stack with nodes being added to it with the cost
(type=Queue)

`parent`: Parent for the nodes that will be created
(type=Node)

getPath(*self*)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle
(type=[Move])

Overrides: UnBlockMeSolver.PathFinder.PathFinder.PathFinder.getPath

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

14.2.2 Properties

Name	Description
Inherited from object	
__class__	

15 Module UnBlockMeSolver.PathFinder.bfs

15.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.PathFinder'`

15.2 Class BFS

object
└
UnBlockMeSolver.PathFinder.PathFinder.PathFinder
└
UnBlockMeSolver.PathFinder.bfs.BFS

15.2.1 Methods

getPath(*self*)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle

(type=[Move])

Overrides: UnBlockMeSolver.PathFinder.PathFinder.PathFinder.getPath

populateQueue(*self*, *graph*, *move*, *queue*, *parent*)

Populate the queue with nodes and their cost.

Parameters

`graph:` Graph to get moves from to populate queue
(type=Map)

`move:` Move to get to the
(type=Move)

`queue:` Stack with nodes being added to it with the cost
(type=Queue)

`parent:` Parent for the nodes that will be created
(type=Node)

Inherited from UnBlockMeSolver.PathFinder.PathFinder.PathFinder(Section 12.2)

__init__()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

15.2.2 Properties

Name	Description
Inherited from object	
__class__	

16 Module UnBlockMeSolver.PathFinder.dfs

16.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.PathFinder'`

16.2 Class DFS

object
└
UnBlockMeSolver.PathFinder.PathFinder.PathFinder
└
UnBlockMeSolver.PathFinder.dfs.DFS

16.2.1 Methods

getPath(*self*)

Get the best path to solve the given graph.

Return Value

Array of moves which represent the path found to solve the puzzle

(type=[Move])

Overrides: UnBlockMeSolver.PathFinder.PathFinder.PathFinder.getPath

populateQueue(*self*, *graph*, *move*, *stack*, *parent*)

Populate the stack with nodes and their cost.

Parameters

`graph:` Graph to get moves from to populate queue

(type=Map)

`move:` Move to get to the

(type=Move)

`stack:` Stack with nodes being added to it with the cost

(type=[Node])

`parent:` Parent for the nodes that will be created

(type=Node)

Inherited from UnBlockMeSolver.PathFinder.PathFinder.PathFinder(Section 12.2)

__init__()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

16.2.2 Properties

Name	Description
Inherited from object	
__class__	

17 Package UnBlockMeSolver.Utility

17.1 Modules

- **MovesTo** *(Section 18, p. 34)*
- **Stack** *(Section 19, p. 35)*

17.2 Variables

Name	Description
__package__	**Value:** `None`

18 Module UnBlockMeSolver.Utility.MovesTo

18.1 Variables

Name	Description
__package__	**Value:** `None`

19 Module UnBlockMeSolver.Utility.Stack

19.1 Variables

Name	Description
__package__	**Value:** `None`

19.2 Class Stack

object
└ **UnBlockMeSolver.Utility.Stack.Stack**

19.2.1 Methods

__init__(*self*)

x.__init__(...) initializes x; see help(type(x)) for signature

Overrides: object.__init__ extit(inherited documentation)

put(*self*, *obj*)

Put object into stack.

Parameters

`obj`: object to be placed into the stack

(type=any type)

get(*self*)

Gets the most recent item and pops it.

rtype: any type

Return Value

Object from stack that was popped.

empty(*self*)

Checks whether this stack is empty or not

Return Value

Stack is empty

(type=boolean)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

19.2.2 Properties

Name	Description
Inherited from object	
__class__	

20 Package UnBlockMeSolver.test

20.1 Modules

20.2 Variables

Name	Description
__package__	**Value:** `None`

21 Package UnBlockMeSolver.test.MapTests

21.1 Modules

- **mapTests** *(Section 22, p. 39)*
- **moveTests** *(Section 23, p. 43)*
- **pieceTests** *(Section 24, p. 45)*
- **readerTests** *(Section 25, p. 47)*

21.2 Variables

Name	Description
__package__	**Value:** `None`

22 Module UnBlockMeSolver.test.MapTests.mapTests

22.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.test.MapTests'`

22.2 Class FakeMapReader

object
└
UnBlockMeSolver.test.MapTests.mapTests.FakeMapReader

Fake object with method and member similar to mapreader class that the map class relies on

22.2.1 Methods

__init__(*self*)

x.__init__(...) initializes x; see help(type(x)) for signature

Overrides: object.__init__ extit(inherited documentation)

get(*self*)

Inherited from object

__delattr__(), __format__(), __getattribute__(), __hash__(), __new__(), __reduce__(), __reduce_ex__(), __repr__(), __setattr__(), __sizeof__(), __str__(), __subclasshook__()

22.2.2 Properties

Name	Description
Inherited from object	
__class__	

22.3 Class TestMap

object
unittest.case.TestCase
UnBlockMeSolver.test.MapTests.mapTests.TestMap

22.3.1 Methods

test_setUpPieces(*self*)

test_init(*self*)

test_convertToMap(*self*)

test_setUp(*self*)

test_isWallOrGoal(*self*)

test_largeRowValid(*self*)

test_numColumnsMatch(*self*)

test_topBottomRowsValid(*self*)

test_midRowsValid(*self*)

test_playerFound(*self*)

test_isValid(*self*)

validate_move(*self*, *move1*, *move2*)

validate_moves(*self*, *moves1*, *moves2*)

test_isValidMove(*self*)

test_makeConfidentMove(*self*)

test_makeMove(*self*)

test_getMoves(*self*)

test_isSolved(*self*)

test_copy(*self*)

test_copyMove(*self*)

test_copyConfidentMove(*self*)

test_validSubractionMoves(*self*)

test_validAdditionMoves(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

22.3.2 Properties

Name	Description
Inherited from object	
__class__	

22.3.3 Class Variables

Name	Description
sample_string	**Value:** '|||||\n|000|\n|**0$\n||||'
default_delimeter	**Value:** '\n'
new_delimeter	**Value:** ','
Inherited from unittest.case.TestCase longMessage, maxDiff	

23 Module UnBlockMeSolver.test.MapTests.moveTests

23.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.test.MapTests'`

23.2 Class TestMove

object
└
unittest.case.TestCase
└
UnBlockMeSolver.test.MapTests.moveTests.TestMove

23.2.1 Methods

test_init(*self*)

test_isValid(*self*)

test_equals(*self*)

test_toStr(*self*)

test_left(*self*)

test_right(*self*)

test_up(*self*)

test_down(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(),

assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

23.2.2 Properties

Name	Description
Inherited from object	
__class__	

23.2.3 Class Variables

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

24 Module UnBlockMeSolver.test.MapTests.pieceTests

24.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.test.MapTests'`

24.2 Class TestPiece

object
└
unittest.case.TestCase
└
UnBlockMeSolver.test.MapTests.pieceTests.TestPiece

24.2.1 Methods

test_init(*self*)

test_move(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

24.2.2 Properties

Name	Description
Inherited from object	
__class__	

24.2.3 Class Variables

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

25 Module UnBlockMeSolver.test.MapTests.readerTests

25.1 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.test.MapTests'`

25.2 Class TestReader

object
└
unittest.case.TestCase
└
UnBlockMeSolver.test.MapTests.readerTests.TestReader

25.2.1 Methods

test_is_loaded(*self*)

test_is_found(*self*)

test_get(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(),

__setattr__(), __sizeof__(), __subclasshook__()

25.2.2 Properties

Name	Description
Inherited from object	
__class__	

25.2.3 Class Variables

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

26 Package UnBlockMeSolver.test.PathFinding

26.1 Modules

- **AStarNodeTests** *(Section 27, p. 50)*
- **heuristicTests** *(Section 28, p. 52)*
- **nodeTests** *(Section 29, p. 54)*
- **pathFindingTests** *(Section 30, p. 56)*

26.2 Variables

Name	Description
__package__	**Value:** `None`

27 Module UnBlockMeSolver.test.PathFinding.AStarNodeTests

27.1 Variables

Name	Description
__package__	**Value:** 'UnBlockMeSolver.test.PathFinding'

27.2 Class NodeTest

object

unittest.case.TestCase

UnBlockMeSolver.test.PathFinding.AStarNodeTests.NodeTest

27.2.1 Methods

test_cost(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

27.2.2 Properties

Name	Description
Inherited from object	
__class__	

27.2.3 Class Variables

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

28 Module UnBlockMeSolver.test.PathFinding.heuristicTests

28.1 Variables

Name	Description
__package__	**Value:** 'UnBlockMeSolver.test.PathFinding'

28.2 Class TestMap

object

unittest.case.TestCase

UnBlockMeSolver.test.PathFinding.heuristicTests.TestMap

28.2.1 Methods

test_manhattan(*self*)

test_euclidian(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

28.2.2 Properties

Name	Description
Inherited from object	
__class__	

28.2.3 Class Variables

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

29 Module UnBlockMeSolver.test.PathFinding.nodeTests

29.1 Variables

Name	Description
__package__	**Value:** 'UnBlockMeSolver.test.PathFinding'

29.2 Class NodeTest

object

unittest.case.TestCase

UnBlockMeSolver.test.PathFinding.nodeTests.NodeTest

29.2.1 Methods

test_reconstructPath(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

29.2.2 Properties

Name	Description
Inherited from object	
__class__	

29.2.3 Class Variables

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

30 Module UnBlockMeSolver.test.PathFinding.pathFindingTests

30.1 Functions

testPaths(*self*, *solver*, *isBFS*=`False`, *checkLength*=`True`)

30.2 Variables

Name	Description
__package__	**Value:** `'UnBlockMeSolver.test.PathFinding'`

30.3 Class PathFindingTest

object

unittest.case.TestCase

UnBlockMeSolver.test.PathFinding.pathFindingTests.PathFindingTes

30.3.1 Methods

solve(*self*, *board*, *moves*)

test_pathFinder(*self*)

test_dfs(*self*)

test_bfs(*self*)

heuristic(*self*, *board*)

Calculate heuristic cost for the given board.

Parameters

`board`: Board being analyzed

(type=Map)

Return Value

Heuristic cost

(type=number)

test_AStar(*self*)

Inherited from unittest.case.TestCase

__call__(), __eq__(), __hash__(), __init__(), __ne__(), __repr__(), __str__(), addCleanup(), addTypeEqualityFunc(), assertAlmostEqual(), assertAlmostEquals(), assertDictContainsSubset(), assertDictEqual(), assertEqual(), assertEquals(), assertFalse(), assertGreater(), assertGreaterEqual(), assertIn(), assertIs(), assertIsInstance(), assertIsNone(), assertIsNot(), assertIsNotNone(), assertItemsEqual(), assertLess(), assertLessEqual(), assertListEqual(), assertMultiLineEqual(), assertNotAlmostEqual(), assertNotAlmostEquals(), assertNotEqual(), assertNotEquals(), assertNotIn(), assertNotIsInstance(), assertNotRegexpMatches(), assertRaises(), assertRaisesRegexp(), assertRegexpMatches(), assertSequenceEqual(), assertSetEqual(), assertTrue(), assertTupleEqual(), assert_(), countTestCases(), debug(), defaultTestResult(), doCleanups(), fail(), failIf(), failIfAlmostEqual(), failIfEqual(), failUnless(), failUnlessAlmostEqual(), failUnlessEqual(), failUnlessRaises(), id(), run(), setUp(), setUpClass(), shortDescription(), skipTest(), tearDown(), tearDownClass()

Inherited from object

__delattr__(), __format__(), __getattribute__(), __new__(), __reduce__(), __reduce_ex__(), __setattr__(), __sizeof__(), __subclasshook__()

30.3.2 Properties

Name	Description
Inherited from object	
__class__	

30.3.3 Class Variables

continued on next page

Name	Description

Name	Description
Inherited from unittest.case.TestCase	
longMessage, maxDiff	

31 Module UnBlockMeSolver.test.files

31.1 Variables

Name	Description
good	**Value:** `'test/Maps/good_map.map'`
not_real	**Value:** `'test/Maps/doesnt_exist.map'`
sample	**Value:** `'test/Maps/sample_string.map'`
no_player	**Value:** `'test/Maps/no_player.map'`
empty	**Value:** `'test/Maps/empty_map.map'`
bad_col	**Value:** `'test/Maps/bad_map.map'`
impossible	**Value:** `'test/Maps/impossible.map'`
right	**Value:** `'test/Maps/right.map'`
left	**Value:** `'test/Maps/left.map'`
up	**Value:** `'test/Maps/up.map'`
down	**Value:** `'test/Maps/down.map'`
no_move	**Value:** `'test/Maps/no_move.map'`
bad_mid_row	**Value:** `'test/Maps/bad_mid_row.map'`
multiple_goals	**Value:** `'test/Maps/multiple_goals.map'`
no_goals	**Value:** `'test/Maps/no_goals.map'`
top_row_bad	**Value:** `'test/Maps/top_row_bad.map'`
bottom_row_bad	**Value:** `'test/Maps/bottom_row_bad.map'`
quick_solve	**Value:** `'test/Maps/solve.map'`
simple_solve	**Value:** `'test/Maps/simple.map'`
easy_solve	**Value:** `'test/Maps/easy.map'`
left_goal	**Value:** `'test/Maps/left_goal.map'`
__package__	**Value:** `None`

Index