

Matrix ADT

Generated by Doxygen 1.9.1

1 Class Index	1
1.1 Class List	1
2 Class Documentation	3
2.1 matrix_adt.Matrix Class Reference	3
2.1.1 Detailed Description	4
2.1.2 Constructor & Destructor Documentation	4
2.1.2.1 __init__()	5
2.1.3 Member Function Documentation	5
2.1.3.1 __eq__()	5
2.1.3.2 __str__()	5
2.1.3.3 add()	6
2.1.3.4 Adjoint()	6
2.1.3.5 cofactor()	6
2.1.3.6 cols()	7
2.1.3.7 Copy()	7
2.1.3.8 det()	7
2.1.3.9 get_val()	7
2.1.3.10 Identity()	8
2.1.3.11 Inverse()	8
2.1.3.12 is_diagonal()	8
2.1.3.13 is_invertable()	9
2.1.3.14 is_lower_triangular()	9
2.1.3.15 is_same_size()	9
2.1.3.16 is_skew_symmetric()	9
2.1.3.17 is_square()	10
2.1.3.18 is_symmetric()	10
2.1.3.19 is_upper_triangular()	10
2.1.3.20 map()	10
2.1.3.21 mat_mul()	11
2.1.3.22 max_val()	11
2.1.3.23 min_val()	11
2.1.3.24 minor()	12
2.1.3.25 mul()	12
2.1.3.26 Random()	12
2.1.3.27 rows()	13
2.1.3.28 set_val()	13
2.1.3.29 tr()	13
2.1.3.30 Transpose()	14
Index	15

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

matrix_adt.Matrix	
This class represents a matrix	3

Chapter 2

Class Documentation

2.1 `matrix_adt.Matrix` Class Reference

This class represents a matrix.

Public Member Functions

- `def __init__ (self, rows, cols)`
Constructor for `Matrix`.
- `def __str__ (self)`
Informal string representation of `Matrix`.
- `def __eq__ (self, other)`
Equivalence comparison of `Matrix` objects.
- `def rows (self)`
Get the number of rows in the matrix.
- `def cols (self)`
Get the number of columns in the matrix.
- `def get_val (self, i, j)`
Get the value stored in the given row and column of the matrix.
- `def set_val (self, i, j, val)`
Set the value stored in the given row and column of the matrix.
- `def max_val (self)`
Get the maximum value stored in the matrix.
- `def min_val (self)`
Get the minimum value stored in the matrix.
- `def map (self, f)`
Apply a function to every element stored in the matrix.
- `def add (self, addend)`
Add a value to every element stored in the matrix or add another `Matrix` object element-wise.
- `def mul (self, multiplier)`
Multiply a value to every element stored in the matrix or multiply another `Matrix` object element-wise.
- `def mat_mul (self, multiplier)`
Perform matrix multiplication with another `Matrix` object.
- `def is_square (self)`
Check if matrix is square.

- def `is_invertable` (self)
Check if matrix is invertible.
- def `is_symmetric` (self)
Check if matrix is symmetric.
- def `is_skew_symmetric` (self)
Check if matrix is skew symmetric.
- def `is_upper_triangular` (self)
Check if matrix is upper triangular.
- def `is_lower_triangular` (self)
Check if matrix is lower triangular.
- def `is_diagonal` (self)
Check if matrix is diagonal.
- def `is_same_size` (self, other)
Check if matrix has the same number of rows and columns as another [Matrix](#) object.
- def `minor` (self, i, j)
Calculate the minor from deleting the given row and column of the matrix.
- def `cofactor` (self, i, j)
Calculate the cofactor from deleting the given row and column of the matrix using the minor.
- def `det` (self)
Calculate the determinant of the matrix if it is square using cofactor expansion.
- def `tr` (self)
Calculate the trace of the matrix if it is square.
- def `Copy` (self)
Make a copy of the [Matrix](#) object (doesn't change original).
- def `Transpose` (self)
Transpose the [Matrix](#) object (doesn't change original).
- def `Adjoint` (self)
Calculate the matrix adjoint (AKA adjugate) using the cofactors if matrix is square (doesn't change original).
- def `Inverse` (self)
Calculate the matrix inverse if matrix is invertible (doesn't change original).

Static Public Member Functions

- def `Identity` (rows)
Static method to generate an identity matrix of the given dimensions (ex.
- def `Random` (rows, cols, min, max)
Static method to generate an matrix with random values (ex.

2.1.1 Detailed Description

This class represents a matrix.

This class represents a matrix object with a 2D array containing the float values stored in the matrix and two integer values representing the number of rows and the number of columns

2.1.2 Constructor & Destructor Documentation

2.1.2.1 `__init__()`

```
def matrix_adt.Matrix.__init__ (
    self,
    rows,
    cols )
```

Constructor for [Matrix](#).

Constructor creates a matrix of zeros and accepts two parameters for the number of rows and the number of columns (ex. `Matrix(2,3)`).

Parameters

<i>rows</i>	integer for number of rows
<i>cols</i>	integer for number of cols

2.1.3 Member Function Documentation

2.1.3.1 `__eq__()`

```
def matrix_adt.Matrix.__eq__ (
    self,
    other )
```

Equivalence comparison of [Matrix](#) objects.

Used for comparing if two matrix objects are equivalent (ex. `matrix1 == matrix2`).

Parameters

<i>other</i>	Matrix object to compare if equal with.
--------------	---

Returns

Returns True if both [Matrix](#) objects are equivalent.

2.1.3.2 `__str__()`

```
def matrix_adt.Matrix.__str__ (
    self )
```

Informal string representation of [Matrix](#).

Used for printing the matrix in a readable form with even spacing (ex. `print(matrix)`)

Returns

String of matrix values in an organized table.

2.1.3.3 add()

```
def matrix_adt.Matrix.add (
    self,
    addend )
```

Add a value to every element stored in the matrix or add another [Matrix](#) object element-wise.

(Note: changes original matrix as well)

Parameters

<i>addend</i>	float or Matrix object of same dimensions to be added element-wise
---------------	--

Returns

A copy of the manipulated [Matrix](#) object.

2.1.3.4 Adjoint()

```
def matrix_adt.Matrix.Adjoint (
    self )
```

Calculate the matrix adjoint (AKA adjugate) using the cofactors if matrix is square (doesn't change original).

Returns

[Matrix](#) object that is the adjoint of the original matrix.

2.1.3.5 cofactor()

```
def matrix_adt.Matrix.cofactor (
    self,
    i,
    j )
```

Calculate the cofactor from deleting the given row and column of the matrix using the minor.

Parameters

<i>i</i>	integer of the row index (must be greater than or equal to 0 and less than number of rows).
<i>j</i>	integer of the column index (must be greater than or equal to 0 and less than number of columns).

Returns

float value of the cofactor.

2.1.3.6 cols()

```
def matrix_adt.Matrix.cols (
    self )
```

Get the number of columns in the matrix.

Returns

integer for the number of columns in the matrix.

2.1.3.7 Copy()

```
def matrix_adt.Matrix.Copy (
    self )
```

Make a copy of the [Matrix](#) object (doesn't change original).

Returns

[Matrix](#) object that is identical in dimensions and values.

2.1.3.8 det()

```
def matrix_adt.Matrix.det (
    self )
```

Calculate the determinant of the matrix if it is square using cofactor expansion.

Returns

float value of the determinant.

2.1.3.9 get_val()

```
def matrix_adt.Matrix.get_val (
    self,
    i,
    j )
```

Get the value stored in the given row and column of the matrix.

Parameters

<i>i</i>	integer of the row index (must be greater than or equal to 0 and less than number of rows).
<i>j</i>	integer of the column index (must be greater than or equal to 0 and less than number of columns).

Returns

float of value stored in the i-th row and j-th column of matrix.

2.1.3.10 Identity()

```
def matrix_adt.Matrix.Identity (
    rows ) [static]
```

Static method to generate an identity matrix of the given dimensions (ex.

Matrix.Identity(5)).

Parameters

<i>rows</i>	integer for number of rows (or cols) of the square identity matrix.
-------------	---

Returns

[Matrix](#) object that is an identity matrix.

2.1.3.11 Inverse()

```
def matrix_adt.Matrix.Inverse (
    self )
```

Calculate the matrix inverse if matrix is invertible (doesn't change original).

Returns

[Matrix](#) object that is the inverse of the original matrix.

2.1.3.12 is_diagonal()

```
def matrix_adt.Matrix.is_diagonal (
    self )
```

Check if matrix is diagonal.

Returns

True if matrix is diagonal.

2.1.3.13 is_invertable()

```
def matrix_adt.Matrix.is_invertable (
    self )
```

Check if matrix is invertible.

Returns

True if matrix is invertible.

2.1.3.14 is_lower_triangular()

```
def matrix_adt.Matrix.is_lower_triangular (
    self )
```

Check if matrix is lower triangular.

Returns

True if matrix is lower triangular.

2.1.3.15 is_same_size()

```
def matrix_adt.Matrix.is_same_size (
    self,
    other )
```

Check if matrix has the same number of rows and columns as another [Matrix](#) object.

Parameters

<i>other</i>	Matrix object to compare sizes with.
--------------	--

Returns

True if both matrices have the same size.

2.1.3.16 is_skew_symmetric()

```
def matrix_adt.Matrix.is_skew_symmetric (
    self )
```

Check if matrix is skew symmetric.

Returns

True if matrix is skew symmetric.

2.1.3.17 is_square()

```
def matrix_adt.Matrix.is_square (
    self )
```

Check if matrix is square.

Returns

True if matrix is square.

2.1.3.18 is_symmetric()

```
def matrix_adt.Matrix.is_symmetric (
    self )
```

Check if matrix is symmetric.

Returns

True if matrix is symmetric.

2.1.3.19 is_upper_triangular()

```
def matrix_adt.Matrix.is_upper_triangular (
    self )
```

Check if matrix is upper triangular.

Returns

True if matrix is upper triangular.

2.1.3.20 map()

```
def matrix_adt.Matrix.map (
    self,
    f )
```

Apply a function to every element stored in the matrix.

(Note: changes original matrix as well)

Parameters

<i>f</i>	A function that takes in the float value of an element and returns a float.
----------	---

Returns

A copy of the manipulated [Matrix](#) object.

2.1.3.21 mat_mul()

```
def matrix_adt.Matrix.mat_mul (
    self,
    multiplier )
```

Perform matrix multiplication with another [Matrix](#) object.

Parameters

<i>multiplier</i>	Matrix object with same number of rows.
-------------------	---

Returns

A [Matrix](#) object that is the product of matrix multiplication.

2.1.3.22 max_val()

```
def matrix_adt.Matrix.max_val (
    self )
```

Get the maximum value stored in the matrix.

Returns

float of maximum value stored in matrix.

2.1.3.23 min_val()

```
def matrix_adt.Matrix.min_val (
    self )
```

Get the minimum value stored in the matrix.

Returns

float of minimum value stored in matrix.

2.1.3.24 minor()

```
def matrix_adt.Matrix.minor (
    self,
    i,
    j )
```

Calculate the minor from deleting the given row and column of the matrix.

Parameters

<i>i</i>	integer of the row index (must be greater than or equal to 0 and less than number of rows).
<i>j</i>	integer of the column index (must be greater than or equal to 0 and less than number of columns).

Returns

float value of the minor.

2.1.3.25 mul()

```
def matrix_adt.Matrix.mul (
    self,
    multiplier )
```

Multiply a value to every element stored in the matrix or multiply another [Matrix](#) object element-wise.

(Note: changes original matrix as well)

Parameters

<i>multiplier</i>	float or Matrix object of same dimensions to be multiplied element-wise
-------------------	---

Returns

A copy of the manipulated [Matrix](#) object.

2.1.3.26 Random()

```
def matrix_adt.Matrix.Random (
    rows,
    cols,
    min,
    max ) [static]
```

Static method to generate an matrix with random values (ex.

[Matrix.Random](#)(8,9,-10,10)).

Parameters

<i>rows</i>	integer for number of rows.
<i>cols</i>	integer for number of cols.
<i>min</i>	int for minimum value in random range (inclusive).
<i>max</i>	float for maximum value in random range (exclusive).

Returns

[Matrix](#) object with random values.

2.1.3.27 rows()

```
def matrix_adt.Matrix.rows (
    self )
```

Get the number of rows in the matrix.

Returns

integer for the number of rows in the matrix.

2.1.3.28 set_val()

```
def matrix_adt.Matrix.set_val (
    self,
    i,
    j,
    val )
```

Set the value stored in the given row and column of the matrix.

Parameters

<i>i</i>	integer of the row index (must be greater than or equal to 0 and less than number of rows).
<i>j</i>	integer of the column index (must be greater than or equal to 0 and less than number of columns).
<i>val</i>	float of value stored to be stored in the i-th row and j-th column of matrix.

2.1.3.29 tr()

```
def matrix_adt.Matrix.tr (
    self )
```

Calculate the trace of the matrix if it is square.

Returns

float value of the trace.

2.1.3.30 Transpose()

```
def matrix_adt.Matrix.Transpose (
    self )
```

Transpose the [Matrix](#) object (doesn't change original).

Returns

[Matrix](#) object that is a transposition of the original matrix.

The documentation for this class was generated from the following file:

- `matrix_adt.py`

Index

- [__eq__](#)
 - [matrix_adt.Matrix](#), [5](#)
 - [__init__](#)
 - [matrix_adt.Matrix](#), [4](#)
 - [__str__](#)
 - [matrix_adt.Matrix](#), [5](#)
- [add](#)
 - [matrix_adt.Matrix](#), [6](#)
- [Adjoint](#)
 - [matrix_adt.Matrix](#), [6](#)
- [cofactor](#)
 - [matrix_adt.Matrix](#), [6](#)
- [cols](#)
 - [matrix_adt.Matrix](#), [7](#)
- [Copy](#)
 - [matrix_adt.Matrix](#), [7](#)
- [det](#)
 - [matrix_adt.Matrix](#), [7](#)
- [get_val](#)
 - [matrix_adt.Matrix](#), [7](#)
- [Identity](#)
 - [matrix_adt.Matrix](#), [8](#)
- [Inverse](#)
 - [matrix_adt.Matrix](#), [8](#)
- [is_diagonal](#)
 - [matrix_adt.Matrix](#), [8](#)
- [is_invertable](#)
 - [matrix_adt.Matrix](#), [8](#)
- [is_lower_triangular](#)
 - [matrix_adt.Matrix](#), [9](#)
- [is_same_size](#)
 - [matrix_adt.Matrix](#), [9](#)
- [is_skew_symmetric](#)
 - [matrix_adt.Matrix](#), [9](#)
- [is_square](#)
 - [matrix_adt.Matrix](#), [10](#)
- [is_symmetric](#)
 - [matrix_adt.Matrix](#), [10](#)
- [is_upper_triangular](#)
 - [matrix_adt.Matrix](#), [10](#)
- [map](#)
 - [matrix_adt.Matrix](#), [10](#)
- [mat_mul](#)
 - [matrix_adt.Matrix](#), [11](#)
- [matrix_adt.Matrix](#), [3](#)
- [__eq__](#), [5](#)
- [__init__](#), [4](#)
- [__str__](#), [5](#)
- [add](#), [6](#)
- [Adjoint](#), [6](#)
- [cofactor](#), [6](#)
- [cols](#), [7](#)
- [Copy](#), [7](#)
- [det](#), [7](#)
- [get_val](#), [7](#)
- [Identity](#), [8](#)
- [Inverse](#), [8](#)
- [is_diagonal](#), [8](#)
- [is_invertable](#), [8](#)
- [is_lower_triangular](#), [9](#)
- [is_same_size](#), [9](#)
- [is_skew_symmetric](#), [9](#)
- [is_square](#), [10](#)
- [is_symmetric](#), [10](#)
- [is_upper_triangular](#), [10](#)
- [map](#), [10](#)
- [mat_mul](#), [11](#)
- [max_val](#), [11](#)
- [min_val](#), [11](#)
- [minor](#), [11](#)
- [mul](#), [12](#)
- [Random](#), [12](#)
- [rows](#), [13](#)
- [set_val](#), [13](#)
- [tr](#), [13](#)
- [Transpose](#), [14](#)
- [max_val](#)
 - [matrix_adt.Matrix](#), [11](#)
- [min_val](#)
 - [matrix_adt.Matrix](#), [11](#)
- [minor](#)
 - [matrix_adt.Matrix](#), [11](#)
- [mul](#)
 - [matrix_adt.Matrix](#), [12](#)
- [Random](#)
 - [matrix_adt.Matrix](#), [12](#)
- [rows](#)
 - [matrix_adt.Matrix](#), [13](#)
- [set_val](#)
 - [matrix_adt.Matrix](#), [13](#)
- [tr](#)
 - [matrix_adt.Matrix](#), [13](#)

Transpose

[matrix_adt.Matrix](#), [14](#)