NumPy - Matrix Library

NumPy package contains a Matrix library **numpy.matlib**. This module has functions that return matrices instead of ndarray objects.

matlib.empty()

The **matlib.empty()** function returns a new matrix without initializing the entries. The function takes the following parameters.

```
numpy.matlib.empty(shape, dtype, order)
```

Where,

Sr.No.	Parameter & Description
1	**shape** **int** or tuple of **int** defining the shape of the new matrix
2	**Dtype** Optional. Data type of the output
3	**order** C or F

Example

Live Demo

```
import numpy.matlib

import numpy as np


print np.matlib.empty((2,2))

# filled with random data
```

It will produce the following output –

```
[[ 2.12199579e-314,   4.24399158e-314]
 [ 4.24399158e-314,   2.12199579e-314]]
```

numpy.matlib.zeros()

This function returns the matrix filled with zeros.

Live Demo

```
import numpy.matlib

import numpy as np

print np.matlib.zeros((2,2))
```

It will produce the following output –

```
[[ 0.  0.]
 [ 0.  0.]]
```

numpy.matlib.ones()

This function returns the matrix filled with 1s.

Live Demo

```
import numpy.matlib

import numpy as np

print np.matlib.ones((2,2))
```

It will produce the following output –

```
[[ 1.  1.]
 [ 1.  1.]]
```

numpy.matlib.eye()

This function returns a matrix with 1 along the diagonal elements and the zeros elsewhere. The function takes the following parameters.

```
numpy.matlib.eye(n, M,k, dtype)
```

Where,

Sr.No.	Parameter & Description

1	**n** The number of rows in the resulting matrix
2	**M** The number of columns, defaults to n
3	**k** Index of diagonal
4	**dtype** Data type of the output

Example

Live Demo

```
import numpy.matlib

import numpy as np

print np.matlib.eye(n = 3, M = 4, k = 0, dtype = float)
```

It will produce the following output –

```
[[ 1.  0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]]
```

numpy.matlib.identity()

The **numpy.matlib.identity()** function returns the Identity matrix of the given size. An identity matrix is a square matrix with all diagonal elements as 1.

Live Demo

```
import numpy.matlib

import numpy as np

print np.matlib.identity(5, dtype = float)
```

It will produce the following output –

```
[[ 1.  0.  0.  0.  0.]
 [ 0.  1.  0.  0.  0.]
 [ 0.  0.  1.  0.  0.]
 [ 0.  0.  0.  1.  0.]
 [ 0.  0.  0.  0.  1.]]
```

numpy.matlib.rand()

The **numpy.matlib.rand()** function returns a matrix of the given size filled with random values.

Example

Live Demo

```
import numpy.matlib

import numpy as np

print np.matlib.rand(3,3)
```

It will produce the following output –

```
[[ 0.82674464  0.57206837  0.15497519]
 [ 0.33857374  0.35742401  0.90895076]
 [ 0.03968467  0.13962089  0.39665201]]
```

Note that a matrix is always two-dimensional, whereas ndarray is an n-dimensional array. Both the objects are inter-convertible.

Example

Live Demo

```
import numpy.matlib

import numpy as np



i = np.matrix('1,2;3,4')

print i
```

It will produce the following output –

```
[[1  2]
 [3  4]]
```

Example

```
import numpy.matlib

import numpy as np



j = np.asarray(i)

print j
```

It will produce the following output −

```
[[1  2]
 [3  4]]
```

Example

```
import numpy.matlib

import numpy as np



k = np.asmatrix (j)

print k
```

It will produce the following output −

```
[[1  2]
 [3  4]]
```