numpy_functions_12

```
>>
import numpy as np
```

arange()

```
>>
# np.arange(start,end,step)
a=np.arange(1,13) # returns values  within given interval, here excluding 13
print(a)

[ 1  2  3  4  5  6  7  8  9 10 11 12]
```

```
>>
a=np.arange(1,13,2) # return values with step 2
print(a)

[ 1  3  5  7  9 11]
```

linspace()

```
>>
a=np.linspace(1,5,4) # generates 4 number between 1 and 5 and all the 4
number are equally spaced, means the differences between them are same
print(a)

[1.         2.33333333 3.66666667 5.        ]
```

reshape()

```
>>
a=np.arange(16)
print(a,'\n')
print(a.reshape(2,8)) # reshapes the 16x1 array into 2x8 array
a.shape

[ 0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15]

[[ 0  1  2  3  4  5  6  7]
 [ 8  9 10 11 12 13 14 15]]

(16,)
```

```
>>
print(a.reshape(2,8).ndim) # showing the dimention after reshaping

2
```

```
>>
print(a.reshape(4,2,2),'\n')
print(a.reshape(4,2,2).ndim)
```

```
[[[ 0  1]
  [ 2  3]]

 [[ 4  5]
  [ 6  7]]

 [[ 8  9]
  [10 11]]

 [[12 13]
  [14 15]]]

3
```

```
>>
a=np.arange(1,13).reshape(6,2)
print(a)
```

```
[[ 1  2]
 [ 3  4]
 [ 5  6]
 [ 7  8]
 [ 9 10]
 [11 12]]
```

ravel()

```
>>
a.ndim
```

```
2
```

```
>>
a.ravel() # converts a multy dimentional array into 1d array with all the
elements
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

```
>>
b=a.reshape(3,2,2)
print(b)
```

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

 [[ 5  6]
  [ 7  8]]
 [[ 9 10]
  [11 12]]]

>>
b.ravel()

array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])
```

flatten()

```
>>
b.ndim

3

>>
b.flatten() # flatten a multy dimentional array into 1d array
# flatten takes argument(order), by default it is b.flatten(order='c') - this
flatten an array by their row major values

array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12])

>>
b.flatten(order='F') # this flatten a multy dimentional array into column
major values

array([ 1,  5,  9,  3,  7, 11,  2,  6, 10,  4,  8, 12])
```

transpose()

```
>>
print(a)

[[ 1  2]
 [ 3  4]
 [ 5  6]
 [ 7  8]
 [ 9 10]
 [11 12]]

>>
a.transpose() # this simply transpose a matrix, means interchanges the row
and columns

array([[ 1,  3,  5,  7,  9, 11],
       [ 2,  4,  6,  8, 10, 12]])

print(np.shape(a))
print(np.shape(a.transpose()))

(6, 2)
(2, 6)
```