Day_29_241123

January 23, 2024

Logical Functions

```
[1]: import numpy as np
```

```
[2]: a  = np.array([6,4,5,0])
     b = np.array([4,3,2,1])
```

```
[3]: a < b
```

```
[3]: array([False, False, False,  True])
```

any function will return true if one condition is True otherwise it will return False

```
[4]: np.any(a<b)
```

```
[4]: True
```

all function will return False if any one condition is False

```
[5]: np.all(a>b)
```

```
[5]: False
```

Defining 2D Arrays

```
[6]: a = np.array([1,2,3,4,5,6,7,8,9,0,1,1,22,1,22,1])
```

```
[7]: b = np.array([[1,2,3],[4,5,6]])
     b
```

```
[7]: array([[1, 2, 3],
            [4, 5, 6]])
```

```
[8]: b.shape
```

```
[8]: (2, 3)
```

```
[9]: a.size
```

```
[9]: 16
```

Reshaping a 1D array into 2D Array

```
[10]: a = a.reshape(4,4)
```

```
[11]: a
```

```
[11]: array([[ 1,  2,  3,  4],
             [ 5,  6,  7,  8],
             [ 9,  0,  1,  1],
             [22,  1, 22,  1]])
```

```
[12]: d = np.arange(12).reshape(2,6)
```

```
[13]: d
```

```
[13]: array([[ 0,  1,  2,  3,  4,  5],
             [ 6,  7,  8,  9, 10, 11]])
```

Transpose of a Matrix

```
[14]: d.T
```

```
[14]: array([[ 0,  6],
             [ 1,  7],
             [ 2,  8],
             [ 3,  9],
             [ 4, 10],
             [ 5, 11]])
```

```
[15]: a.T
```

```
[15]: array([[ 1,  5,  9, 22],
             [ 2,  6,  0,  1],
             [ 3,  7,  1, 22],
             [ 4,  8,  1,  1]])
```

Quiz

```
[16]: c = np.array([1,2,3,4,5])
      mask = (c%2 ==0)
      c[mask] = -1
      c
```

```
[16]: array([ 1, -1,  3, -1,  5])
```

Decreasing the Dimensions (Flatten) 2D to 1D

```
[17]: a
```

```
[17]: array([[ 1,  2,  3,  4],
             [ 5,  6,  7,  8],
             [ 9,  0,  1,  1],
        [22,  1, 22,  1]])
```

```
[18]: a.flatten()
```

```
[18]: array([ 1,  2,  3,  4,  5,  6,  7,  8,  9,  0,  1,  1, 22,  1, 22,  1])
```

Indexing/Slicing over arrays

```
[19]: a[[0,1,2],[0,1,2]]
```

```
[19]: array([1, 6, 1])
```

```
[20]: a
```

```
[20]: array([[ 1,  2,  3,  4],
             [ 5,  6,  7,  8],
             [ 9,  0,  1,  1],
             [22,  1, 22,  1]])
```

```
[21]: a[:]
```

```
[21]: array([[ 1,  2,  3,  4],
             [ 5,  6,  7,  8],
             [ 9,  0,  1,  1],
             [22,  1, 22,  1]])
```

```
[22]: a[0:2] # a[rows : columns]
```

```
[22]: array([[1, 2, 3, 4],
             [5, 6, 7, 8]])
```

```
[23]: a[1:3,2:3]
```

```
[23]: array([[7],
             [1]])
```

Masking (Fancy indexing)

```
[24]: a[a>2].reshape(3,3)
```

```
[24]: array([[ 3,  4,  5],
             [ 6,  7,  8],
             [ 9, 22, 22]])
```

```
[25]: np.max(a,axis=1) # When axis = 0 it will consider Column, axis = 1 it will␣
      ↪consider row
```

```
[25]: array([ 4,  8,  9, 22])
```

```
[26]: a
```

```
[26]: array([[ 1,  2,  3,  4],
              [ 5,  6,  7,  8],
              [ 9,  0,  1,  1],
              [22,  1, 22,  1]])
```

```
[27]: np.sum(a,axis=1)
```

```
[27]: array([10, 26, 11, 46])
```

```
[28]: np.sort(a) # Default axis is 1 Which is Row
```

```
[28]: array([[ 1,  2,  3,  4],
              [ 5,  6,  7,  8],
              [ 0,  1,  1,  9],
              [ 1,  1, 22, 22]])
```

```
[29]: np.argmin(a) # argmin will return the index of min value
```

```
[29]: 9
```

```
[32]: a = a.reshape(1,16)
      np.argsort(a)
```

```
[32]: array([[ 9,  0, 10, 11, 13, 15,  1,  2,  3,  4,  5,  6,  7,  8, 12, 14]],
            dtype=int64)
```

```
[ ]:
```