Lecture_3

January 19, 2023

1 LEC 3: Revisiting Lists and NumPy Continued

1.1 Back to Lists

```
[26]: a = [ 21, 5, 6.0, 7, 90, 20, 23, 43, 100, 67, 9.5 ]
```

1.1.1 Slicing

```
[12]: a[1:7]     # a[start:stop:steps]
```

```
[12]: [5, 6.0, 7, 90, 20, 23]
```

```
[13]: a[1:7:2]
```

```
[13]: [5, 7, 20]
```

```
[14]: a[:7]
```

```
[14]: [21, 5, 6.0, 7, 90, 20, 23]
```

```
[15]: a[2:]
```

```
[15]: [6.0, 7, 90, 20, 23, 43, 100, 67, 9.5]
```

```
[33]: a[5:-1]
```

```
[33]: [20, 23, 43, 100, 67]
```

```
[34]: a[-1:-3]
```

```
[34]: []
```

```
[35]: a[-1:4]
```

```
[35]: []
```

```
[22]: a[:-1]
```

```
[22]: [21, 5, 6.0, 7, 90, 20, 23, 43, 100, 67]
```

```
[23]: a[::-1]
```

```
[23]: [9.5, 67, 100, 43, 23, 20, 90, 7, 6.0, 5, 21]
```

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

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

1.1.2 Few applications

```
[17]: len(a)
```

```
[17]: 11
```

```
[18]: sum(a)
```

```
[18]: 391.5
```

```
[20]: mean = sum(a)/len(a)
      mean
```

```
[20]: 35.59090909090909
```

```
[21]: max(a)
```

```
[21]: 100
```

```
[36]: a.index(max(a))
```

```
[36]: 8
```

```
[22]: min(a)
```

```
[22]: 5
```

```
[23]: sorted(a)    # it does not change the original list
```

```
[23]: [5, 6.0, 7, 9.5, 20, 21, 23, 43, 67, 90, 100]
```

```
[24]: sorted(a, reverse = True)
```

```
[24]: [100, 90, 67, 43, 23, 21, 20, 9.5, 7, 6.0, 5]
```

```
[30]: a.sort()   # it changes the original list
      a
```

```
[30]: [5, 6.0, 7, 9.5, 20, 21, 23, 43, 67, 90, 100]
```

```
[32]: a.sort(reverse = True)
      a
```

```
[32]: [100, 90, 67, 43, 23, 21, 20, 9.5, 7, 6.0, 5]
```

```
[27]: b = ['zakir','abhi','biki','jaya']
      sorted(b)
```

```
[27]: ['abhi', 'biki', 'jaya', 'zakir']
```

```
[37]: c = [2,6,8,6,6,8,8,9,0,1,4]
      c.count(6)
```

```
[37]: 3
```

```
[51]: a = [ 21, 5, 6.0, 7, 90, 20, 23, 43, 100, 67, 9.5 ]
      b = [ 54, 7, 78, 89.0, 0.1, 32, 12, 89.9, 32, 6, 2 ]
```

```
[58]: print(a+b)
```

```
[21, 5, 6.0, 7, 90, 20, 23, 43, 100, 67, 9.5, 54, 7, 78, 89.0, 0.1, 32, 12,
89.9, 32, 6, 2] [21, 5, 6.0, 7, 90, 20, 23, 43, 100, 67, 9.5, 21, 5, 6.0, 7, 90,
20, 23, 43, 100, 67, 9.5]
```

```
[59]: print(2*a)
```

```
[21, 5, 6.0, 7, 90, 20, 23, 43, 100, 67, 9.5, 21, 5, 6.0, 7, 90, 20, 23, 43,
100, 67, 9.5]
```

1.2 Back to NumPy

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

1.2.1 Iterator on array

```
[4]: a = np.array([2,5,7,9])
```

```
[67]: for i in a:
          print(i**2)
```

```
4
25
49
81
```

[66]:
```
a*a
```

[66]:
```
array([ 4, 25, 49, 81])
```

[68]:
```
# iterating over 2d array

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

for i in a:
    print(i)
```

```
[1 2 3]
[4 5 6]
```

[69]:
```
for i in np.nditer(a, order = 'F'):
    print(i)
```

```
1
4
2
5
3
6
```

[70]:
```
for i in np.nditer(a, order = 'C'):
    print(i)
```

```
1
2
3
4
5
6
```

[10]:
```
for i in np.nditer(a, order = 'C'):
    print(i**2,end = ',')
```

```
4,25,49,81,
```

1.2.2 Indexing of arrays

[50]:
```
a = np.array([2,5,8,11,14,32,67,0])
print(a[0], a[-1], a[1])
```

```
2 0 5
```

[85]:
```
b = np.array([[1,2,3,4], [4,5,6,9], [3,6,9,0], [4,0,1,6],[0,0,9,3]])
print(b[0],b[1],b[-1],b[-2])
```

```
[1 2 3 4] [4 5 6 9] [0 0 9 3] [4 0 1 6]
```

1.2.3 Slicing of Arrays

```
[51]: a[0:3]    # same as lists
```

```
[51]: array([2, 5, 8])
```

```
[52]: a[::-1]
```

```
[52]: array([ 0, 67, 32, 14, 11,  8,  5,  2])
```

Slicing 2D arrays

```
[58]: b[:]
```

```
[58]: array([[1, 2, 3, 4],
             [4, 5, 6, 9],
             [3, 6, 9, 0],
             [4, 0, 1, 6],
             [0, 0, 9, 3]])
```

```
[59]: b[:3]
```

```
[59]: array([[1, 2, 3, 4],
             [4, 5, 6, 9],
             [3, 6, 9, 0]])
```

```
[60]: b[::2]
```

```
[60]: array([[1, 2, 3, 4],
             [3, 6, 9, 0],
             [0, 0, 9, 3]])
```

```
[61]: b[:-1]
```

```
[61]: array([[1, 2, 3, 4],
             [4, 5, 6, 9],
             [3, 6, 9, 0],
             [4, 0, 1, 6]])
```

Column slicing

```
[63]: b[:]
```

```
[63]: array([[1, 2, 3, 4],
             [4, 5, 6, 9],
             [3, 6, 9, 0],
       [4, 0, 1, 6],
       [0, 0, 9, 3]])
```

```
[64]: b[:,0]
```

```
[64]: array([1, 4, 3, 4, 0])
```

```
[66]: b[:,-1]
```

```
[66]: array([4, 9, 0, 6, 3])
```

General Structure: A[: : , : :]

```
[67]: b[:, 1:3]
```

```
[67]: array([[2, 3],
             [5, 6],
             [6, 9],
             [0, 1],
             [0, 9]])
```

```
[71]: b[2:4,1:3]
```

```
[71]: array([[6, 9],
             [0, 1]])
```

Slicing by conditional statement

```
[73]: b[b>3]  #picks elements, returns an 1D array
```

```
[73]: array([4, 4, 5, 6, 9, 6, 9, 4, 6, 9])
```

1.2.4 Swap elements

```
[86]: b
```

```
[86]: array([[1, 2, 3, 4],
             [4, 5, 6, 9],
             [3, 6, 9, 0],
             [4, 0, 1, 6],
             [0, 0, 9, 3]])
```

```
[87]: b[[1,2]] = b[[2,1]]
      b
```

```
[87]: array([[1, 2, 3, 4],
             [3, 6, 9, 0],
             [4, 5, 6, 9],
             [4, 0, 1, 6],
             [0, 0, 9, 3]])
```

1.2.5 Reshape and flatten

```
[88]: b.shape
```

```
[88]: (5, 4)
```

```
[94]: c = b.reshape(4,5)
      c
```

```
[94]: array([[1, 2, 3, 4, 3],
             [6, 9, 0, 4, 5],
             [6, 9, 4, 0, 1],
             [6, 0, 0, 9, 3]])
```

```
[95]: a
```

```
[95]: array([ 2,  5,  8, 11, 14, 32, 67,  0])
```

```
[98]: d = a.reshape(2,4)
      d
```

```
[98]: array([[ 2,  5,  8, 11],
             [14, 32, 67,  0]])
```

```
[103]: e = b.flatten()
       e
```

```
[103]: array([1, 2, 3, 4, 3, 6, 9, 0, 4, 5, 6, 9, 4, 0, 1, 6, 0, 0, 9, 3])
```

1.2.6 Check elements

```
[106]: b
```

```
[106]: array([[1, 2, 3, 4],
              [3, 6, 9, 0],
              [4, 5, 6, 9],
              [4, 0, 1, 6],
              [0, 0, 9, 3]])
```

```
[105]: np.all(b,0) # along axis = 0 (column)
```

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

```
[108]: np.all(b,1) # along axis = 1 (rows)
```

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

1.2.7 Where is the element

```
[109]: np.where(b==0)  # 1st array row index and 2nd array column index
```

```
[109]: (array([1, 3, 4, 4], dtype=int64), array([3, 1, 0, 1], dtype=int64))
```

```
[110]: a
```

```
[110]: array([ 2,  5,  8, 11, 14, 32, 67,  0])
```

```
[112]: np.where(a==11)
```

```
[112]: (array([3], dtype=int64),)
```

```
[115]: r = np.random.random(size = (3,3))
       r
```

```
[115]: array([[0.68712312, 0.95305222, 0.22478599],
              [0.91777014, 0.33864868, 0.11558689],
              [0.13355812, 0.18386187, 0.3331402 ]])
```

```
[117]: np.where(r<0.5,0,r)
```

```
[117]: array([[0.68712312, 0.95305222, 0.        ],
              [0.91777014, 0.        , 0.        ],
              [0.        , 0.        , 0.        ]])
```

```
[ ]:
```

1.2.8 Insert and delete elements

```
[118]: a
```

```
[118]: array([ 2,  5,  8, 11, 14, 32, 67,  0])
```

```
[121]: np.insert(a,4,56)
       a
```

```
[121]: array([ 2,  5,  8, 11, 56, 32, 67,  0])
```

```
[122]: b
```

```
[122]: array([[1, 2, 3, 4],
              [3, 6, 9, 0],
              [4, 5, 6, 9],
              [4, 0, 1, 6],
              [0, 0, 9, 3]])
```

```
[124]: np.insert(b,1,[8,9,0,3], axis=0) # along rows
```

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

```
[126]: np.insert(b,1,[8,9,0,3,2], axis=1) # along columns
```

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

```
[128]: a
```

```
[128]: array([ 2,  5,  8, 11, 56, 32, 67,  0])
```

```
[127]: np.delete(a,2)
```

```
[127]: array([ 2,  5, 11, 56, 32, 67,  0])
```

```
[131]: b
```

```
[131]: array([[1, 2, 3, 4],
              [3, 6, 9, 0],
              [4, 5, 6, 9],
              [4, 0, 1, 6],
              [0, 0, 9, 3]])
```

```
[130]: np.delete(b,2, axis = 0)
```

```
[130]: array([[1, 2, 3, 4],
              [3, 6, 9, 0],
              [4, 0, 1, 6],
              [0, 0, 9, 3]])
```