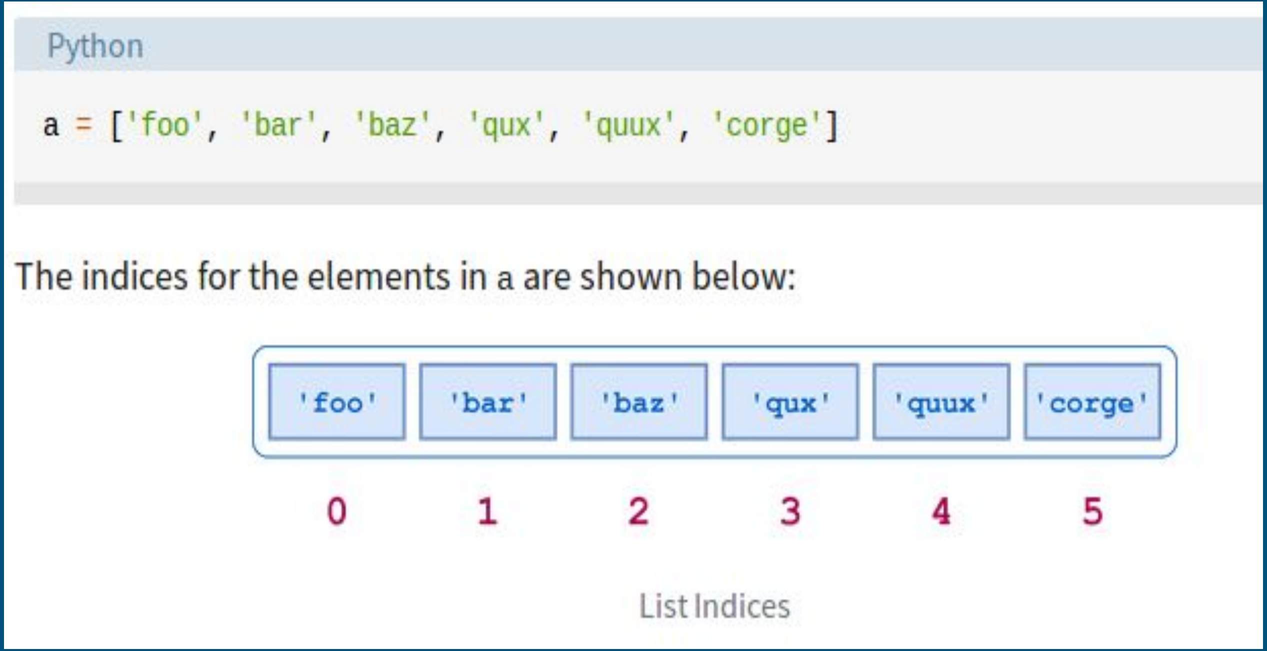
Big Data with Python

By Odin Outsourcing

List / Array



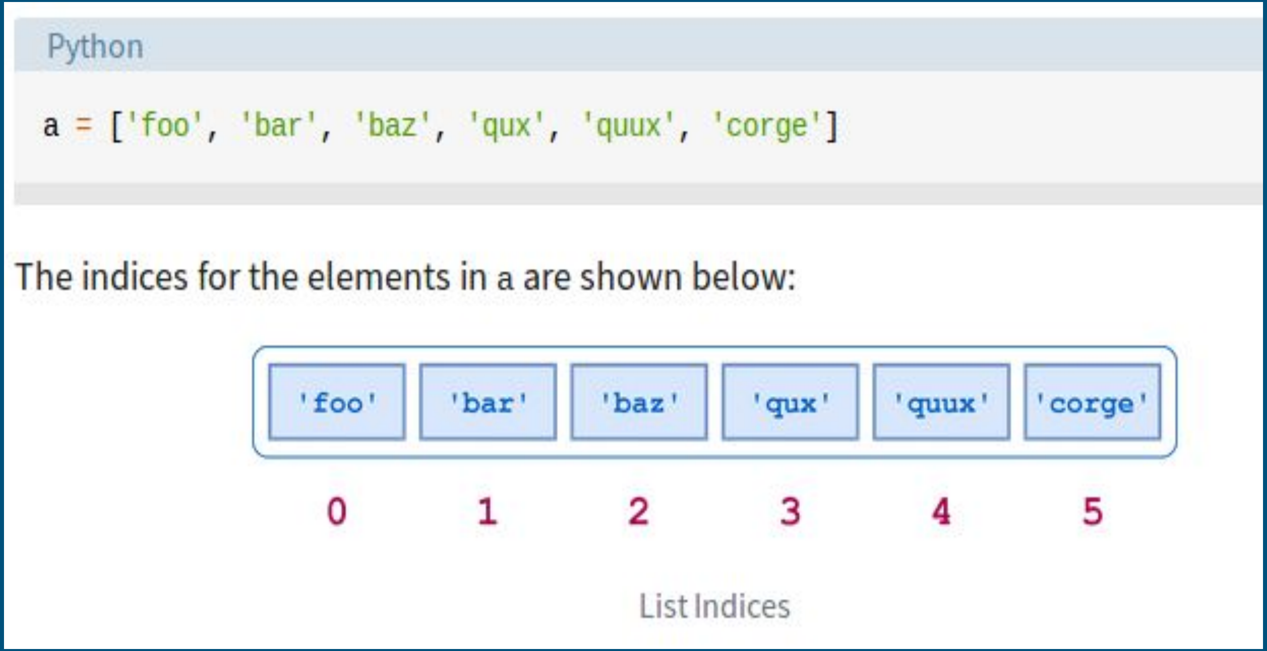
Python

```python
a = ['foo', 'bar', 'baz', 'qux', 'quux', 'corge']
```

The indices for the elements in a are shown below:

List Indices

List / Array (Con.)



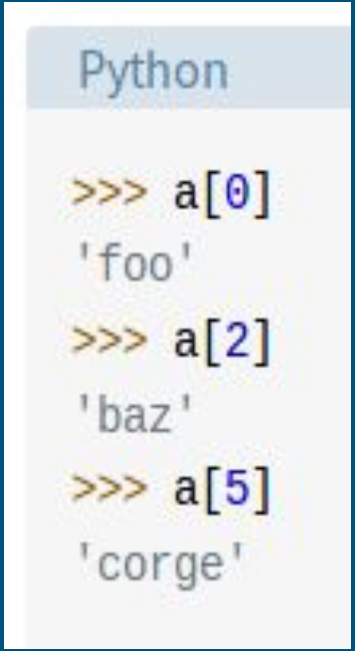
Python

```python
a = ['foo', 'bar', 'baz', 'qux', 'quux', 'corge']
```

The indices for the elements in a are shown below:

```
'foo'  'bar'  'baz'  'qux'  'quux'  'corge'
  0      1      2      3      4       5
```

List Indices

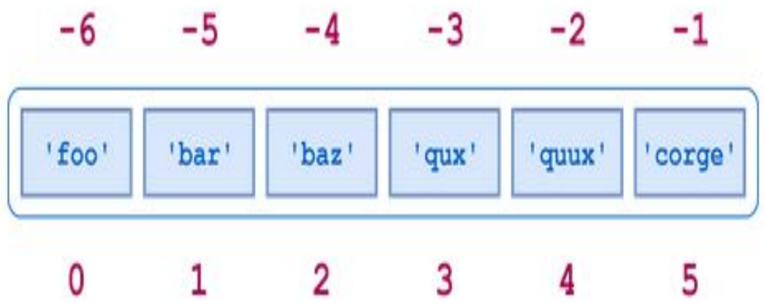


Python

```
>>> a[0]
'foo'
>>> a[2]
'baz'
>>> a[5]
'corge'
```

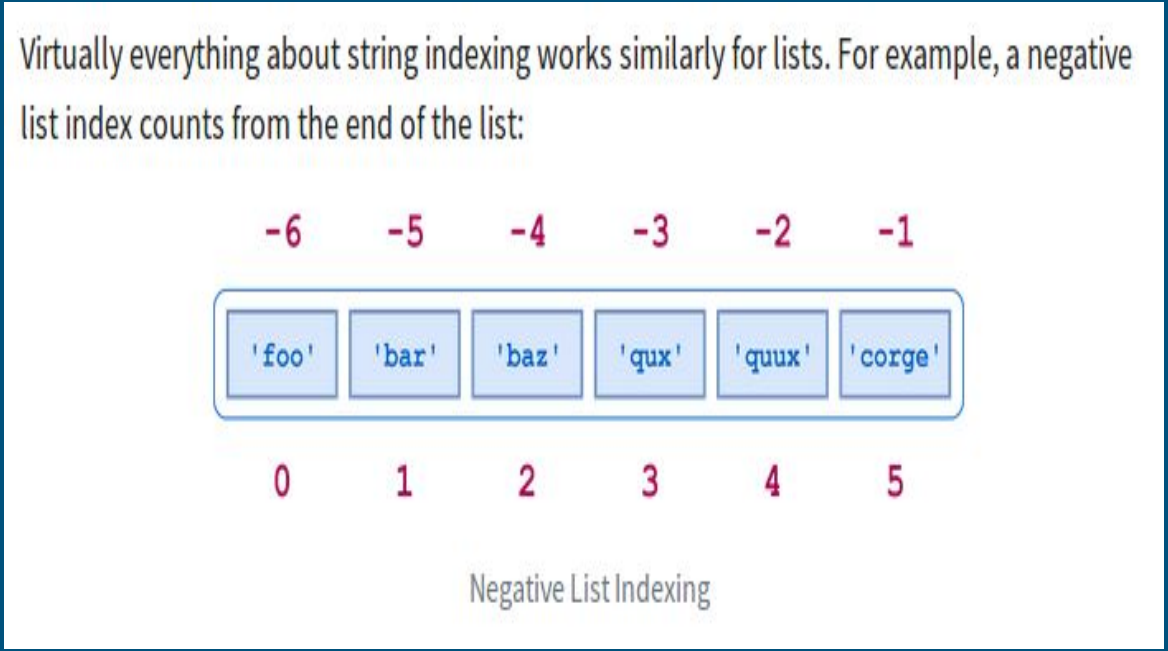
List / Array (Con.)

Virtually everything about string indexing works similarly for lists. For example, a negative list index counts from the end of the list:



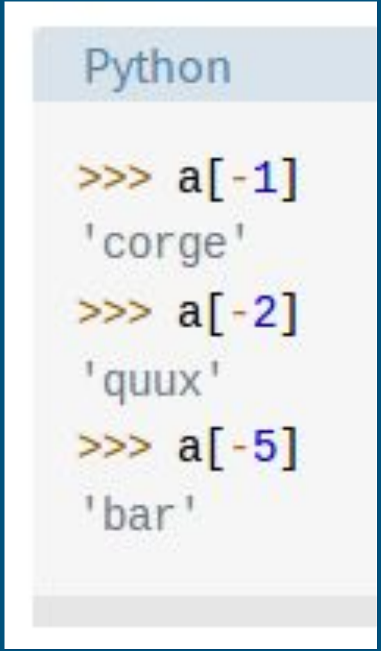
Negative List Indexing

List / Array (Con.)



Virtually everything about string indexing works similarly for lists. For example, a negative list index counts from the end of the list:

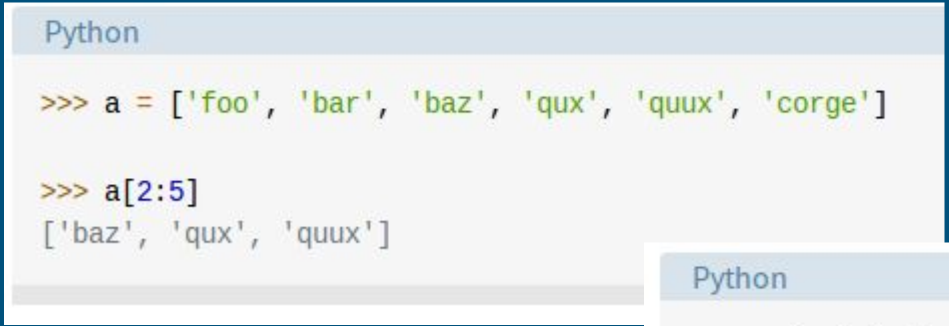
Negative List Indexing



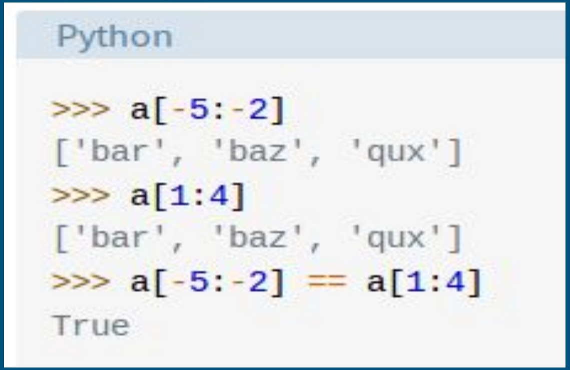
Python

```
>>> a[-1]
'corge'
>>> a[-2]
'quux'
>>> a[-5]
'bar'
```

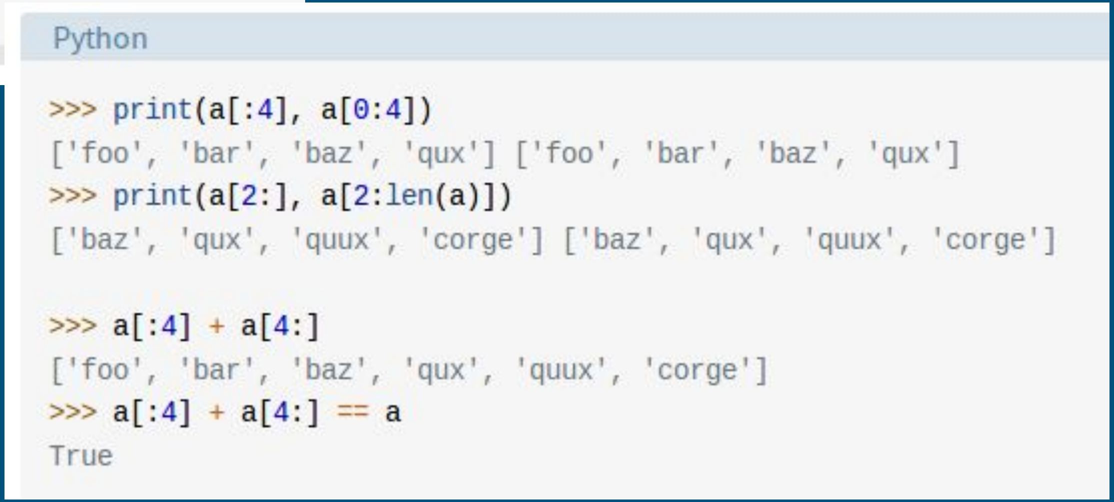
List / Array Slicing


Python

```
>>> a = ['foo', 'bar', 'baz', 'qux', 'quux', 'corge']

>>> a[2:5]
['baz', 'qux', 'quux']
```

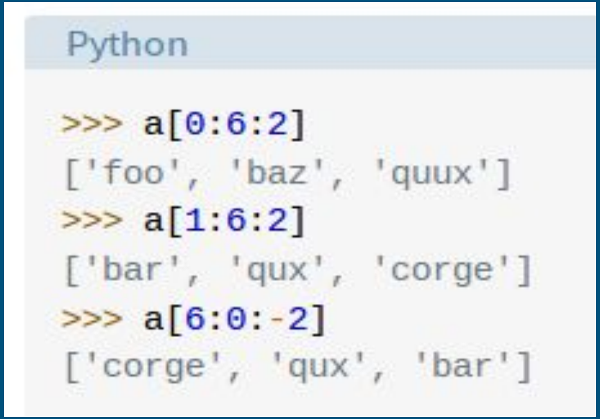

Python

```
>>> a[-5:-2]
['bar', 'baz', 'qux']
>>> a[1:4]
['bar', 'baz', 'qux']
>>> a[-5:-2] == a[1:4]
True
```

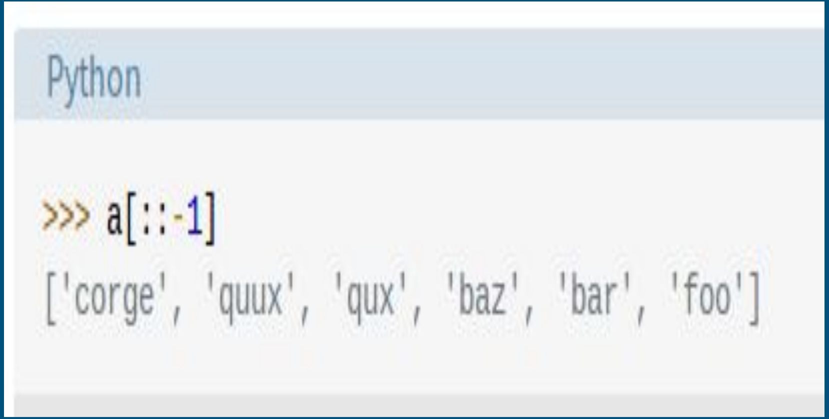

Python

```
>>> print(a[:4], a[0:4])
['foo', 'bar', 'baz', 'qux'] ['foo', 'bar', 'baz', 'qux']
>>> print(a[2:], a[2:len(a)])
['baz', 'qux', 'quux', 'corge'] ['baz', 'qux', 'quux', 'corge']

>>> a[:4] + a[4:]
['foo', 'bar', 'baz', 'qux', 'quux', 'corge']
>>> a[:4] + a[4:] == a
True
```

List / Array Slicing (Con.)

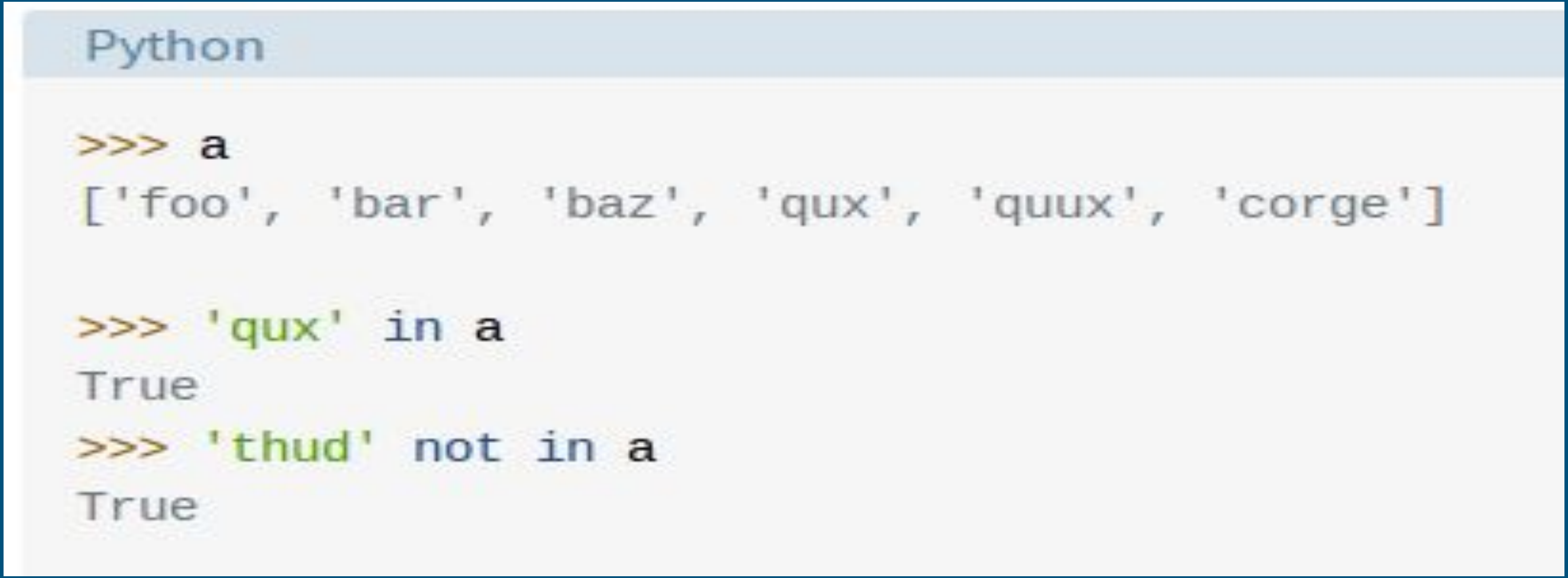

Python

```
>>> a[0:6:2]
['foo', 'baz', 'quux']
>>> a[1:6:2]
['bar', 'qux', 'corge']
>>> a[6:0:-2]
['corge', 'qux', 'bar']
```


Python

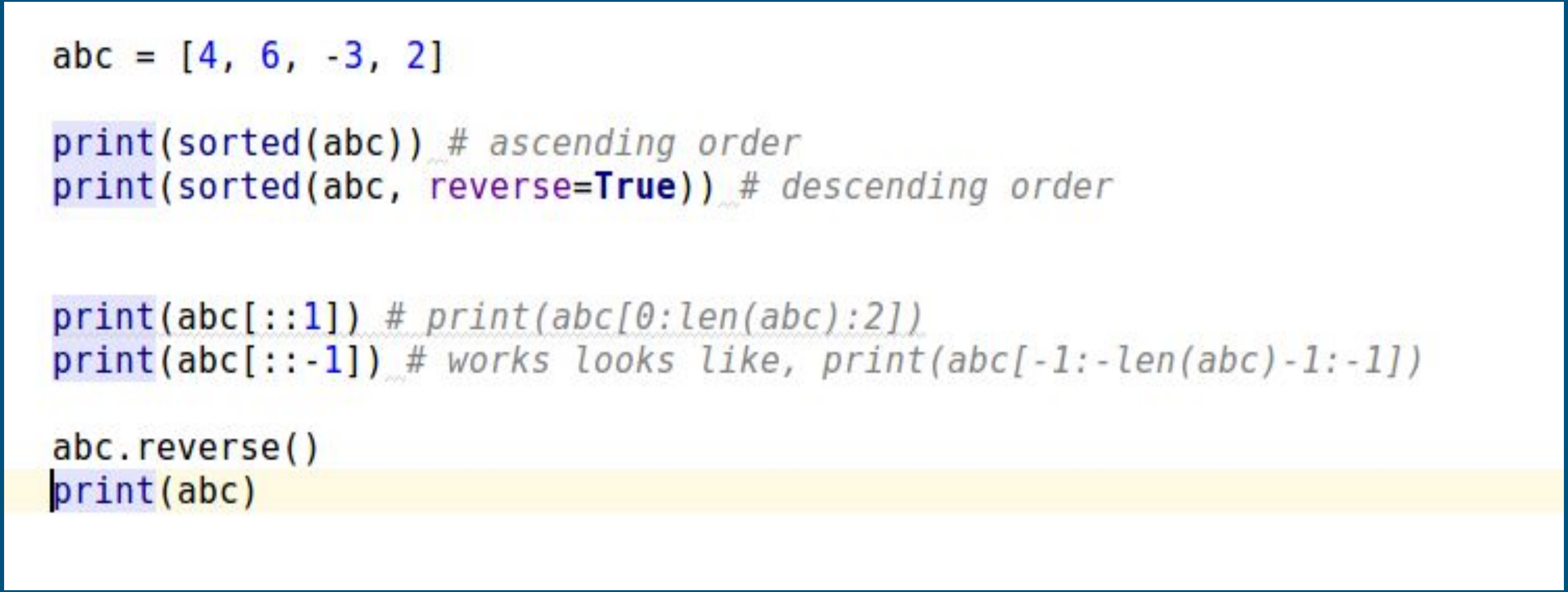
```
>>> a[::-1]
['corge', 'quux', 'qux', 'baz', 'bar', 'foo']
```

List / Array (in side /out side)


Python

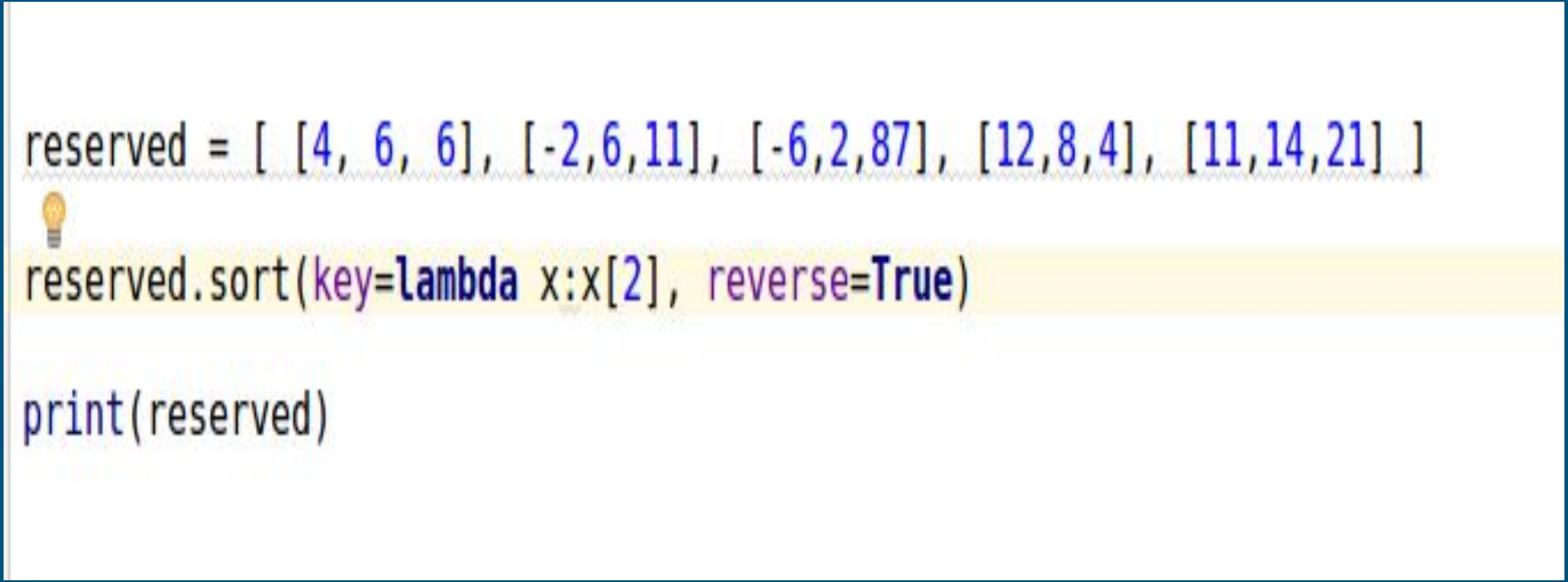
```
>>> a
['foo', 'bar', 'baz', 'qux', 'quux', 'corge']

>>> 'qux' in a
True
>>> 'thud' not in a
True
```

List / Array (Sorting)



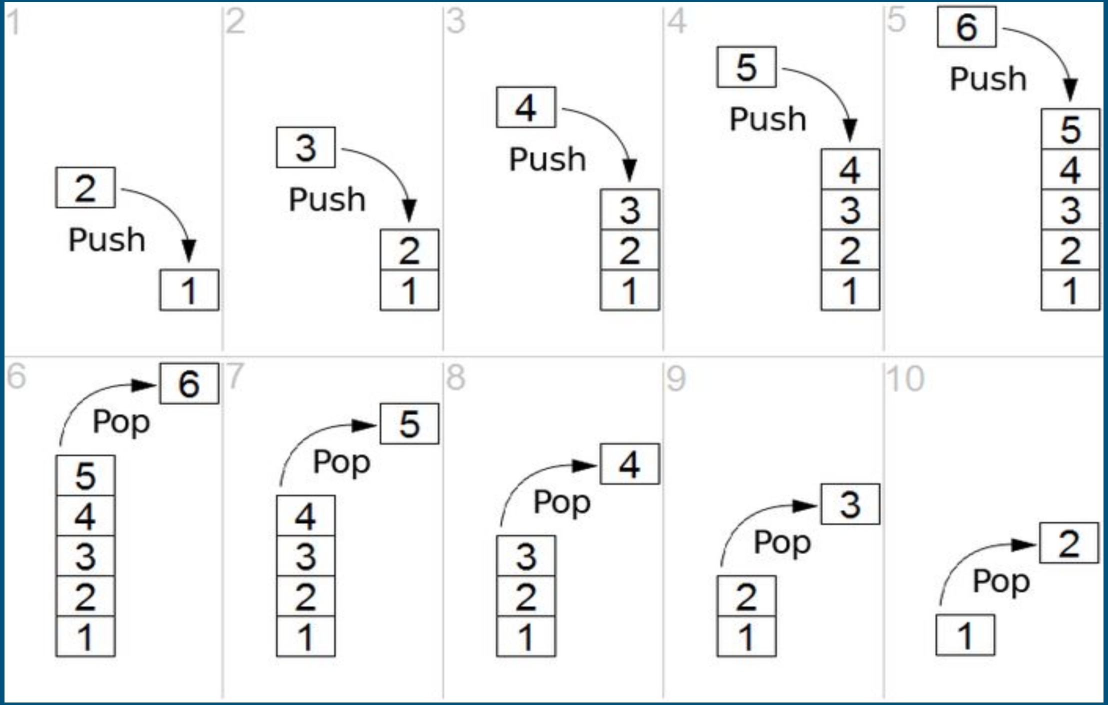
```python
abc = [4, 6, -3, 2]

print(sorted(abc)) # ascending order
print(sorted(abc, reverse=True)) # descending order


print(abc[::1]) # print(abc[0:len(abc):2])
print(abc[::-1]) # works looks like, print(abc[-1:-len(abc)-1:-1])

abc.reverse()
print(abc)
```

List / Array (Specific element-wise Sorting)

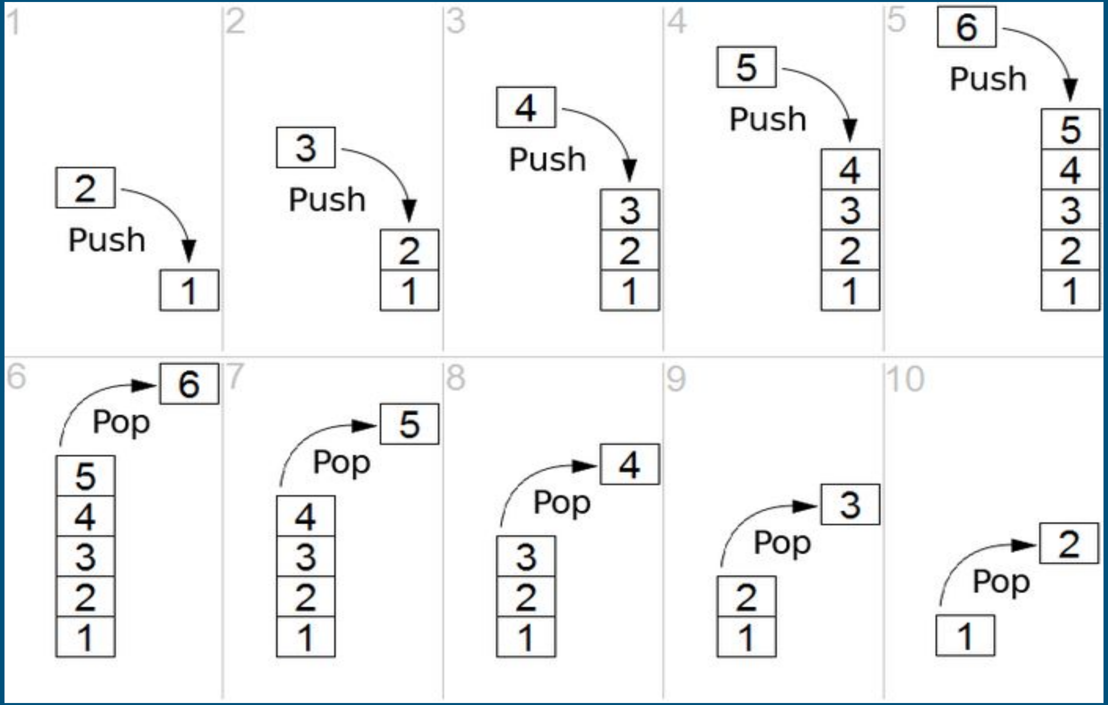


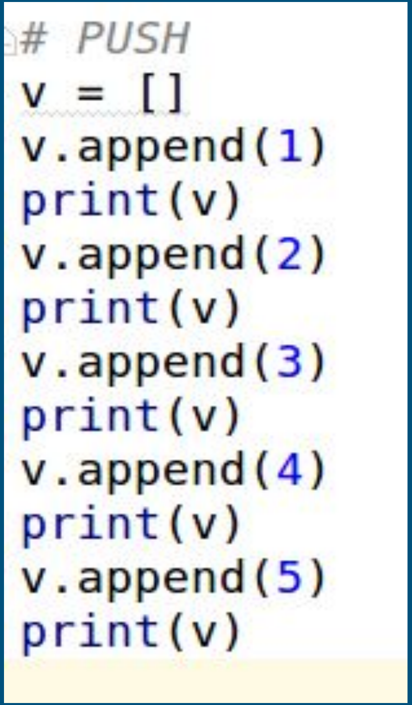
```
reserved = [ [4, 6, 6], [-2,6,11], [-6,2,87], [12,8,4], [11,14,21] ]

reserved.sort(key=lambda x:x[2], reverse=True)

print(reserved)
```

List / Array (Stack)



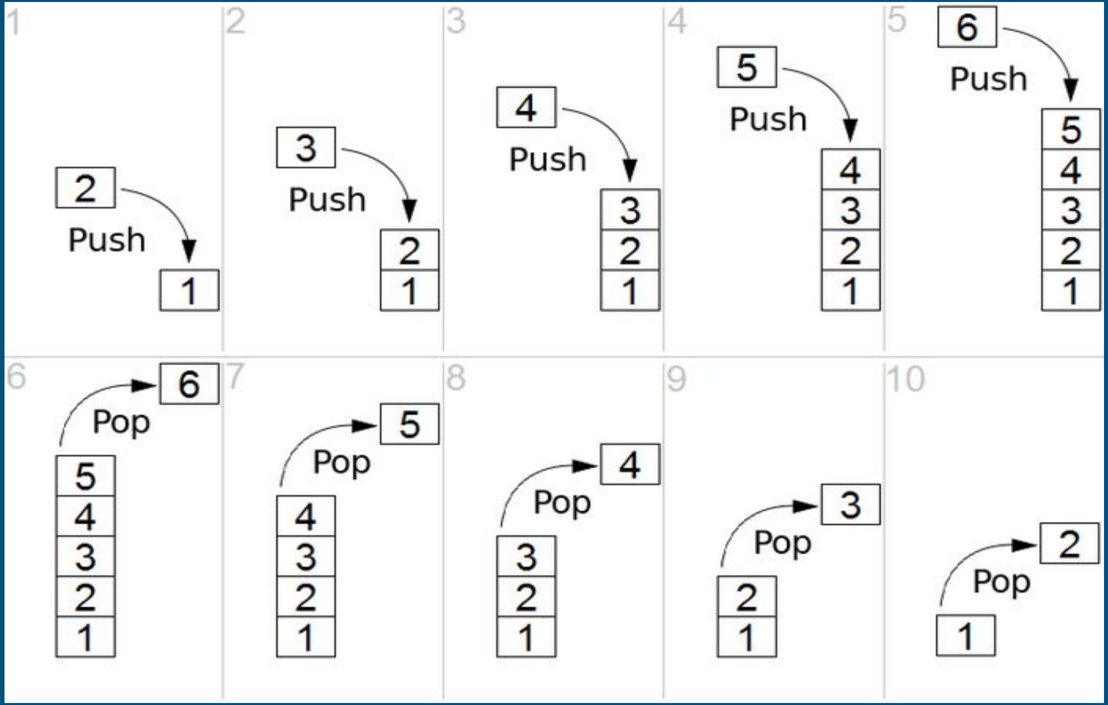
List / Array (Stack): push / append ( )

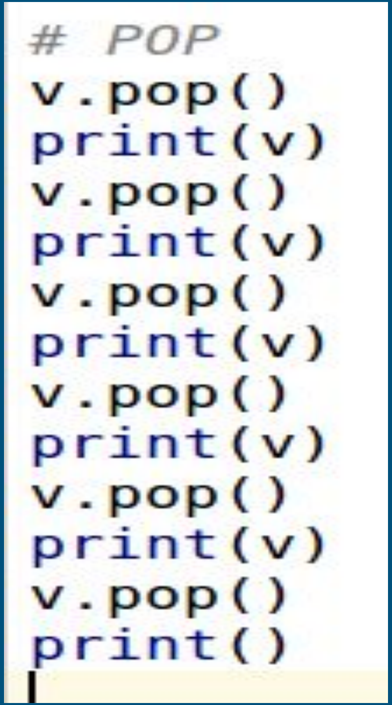




```
# PUSH
v = []
v.append(1)
print(v)
v.append(2)
print(v)
v.append(3)
print(v)
v.append(4)
print(v)
v.append(5)
print(v)
```

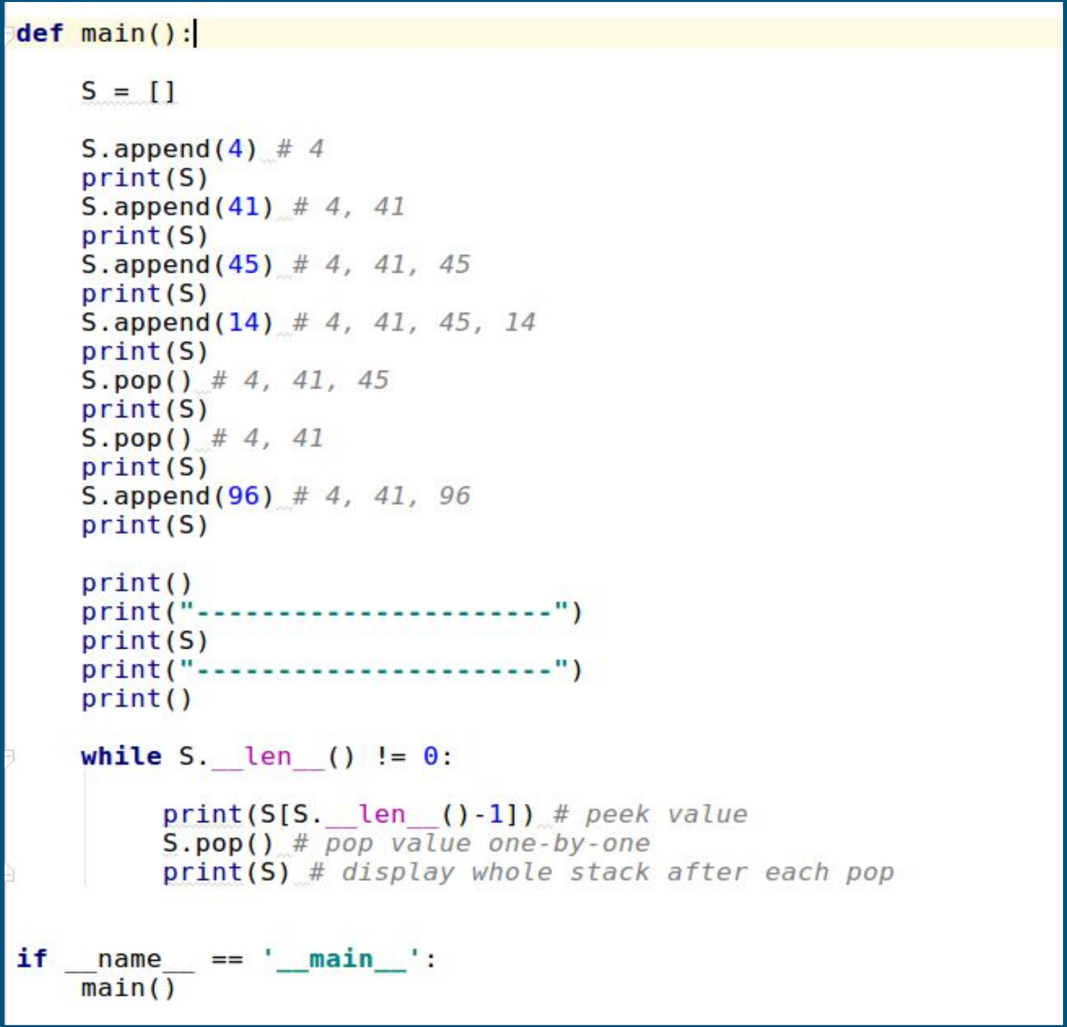
List / Array (Stack): pop( )





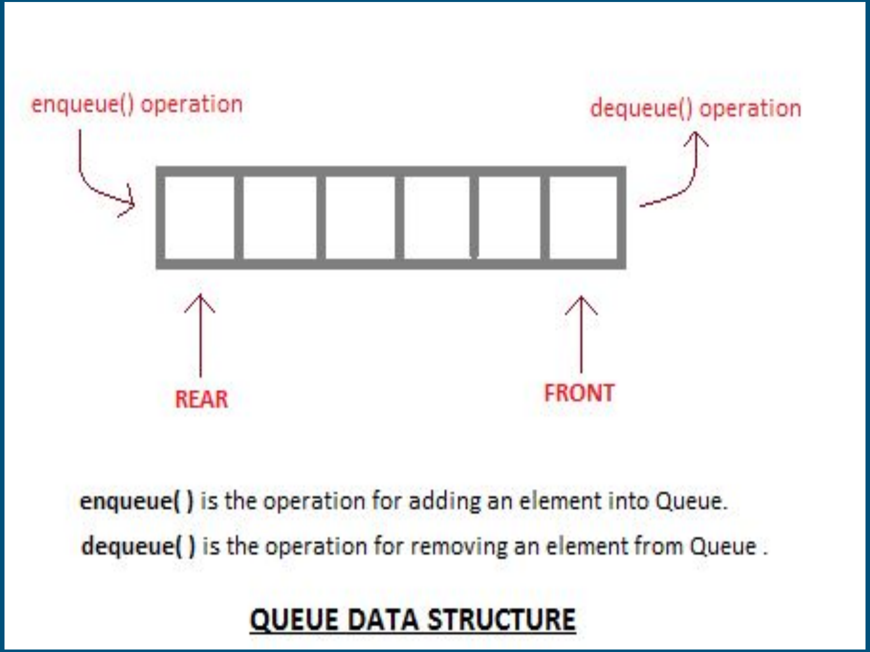
```
# POP
v.pop()
print(v)
v.pop()
print(v)
v.pop()
print(v)
v.pop()
print(v)
v.pop()
print()
```

Stack (Hands-On)



```python
def main():

    S = []

    S.append(4) # 4
    print(S)
    S.append(41) # 4, 41
    print(S)
    S.append(45) # 4, 41, 45
    print(S)
    S.append(14) # 4, 41, 45, 14
    print(S)
    S.pop() # 4, 41, 45
    print(S)
    S.pop() # 4, 41
    print(S)
    S.append(96) # 4, 41, 96
    print(S)

    print()
    print("----------------------")
    print(S)
    print("----------------------")
    print()

    while S.__len__() != 0:

        print(S[S.__len__()-1]) # peek value
        S.pop() # pop value one-by-one
        print(S) # display whole stack after each pop


if __name__ == '__main__':
    main()
```

Queue

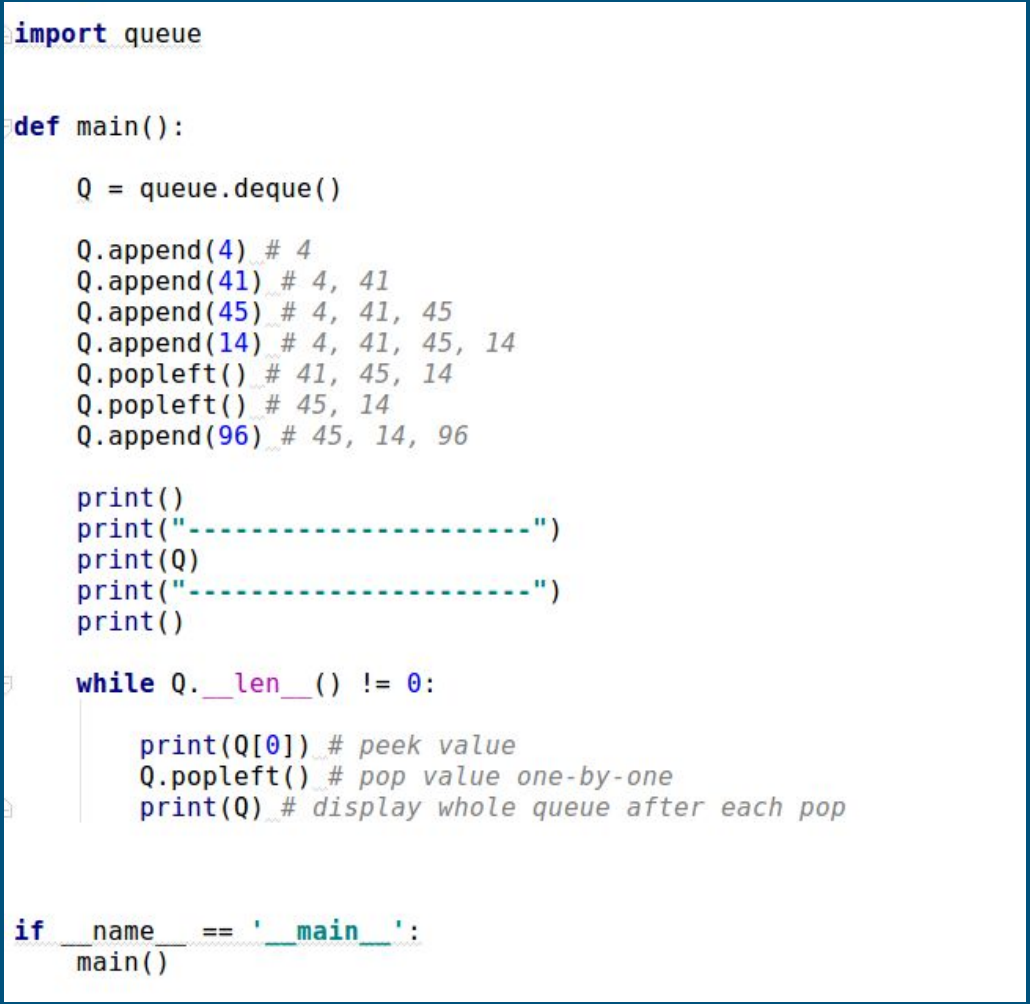


enqueue( ) is the operation for adding an element into Queue.

dequeue( ) is the operation for removing an element from Queue .

<u>**QUEUE DATA STRUCTURE**</u>

Queue (Hands-On)

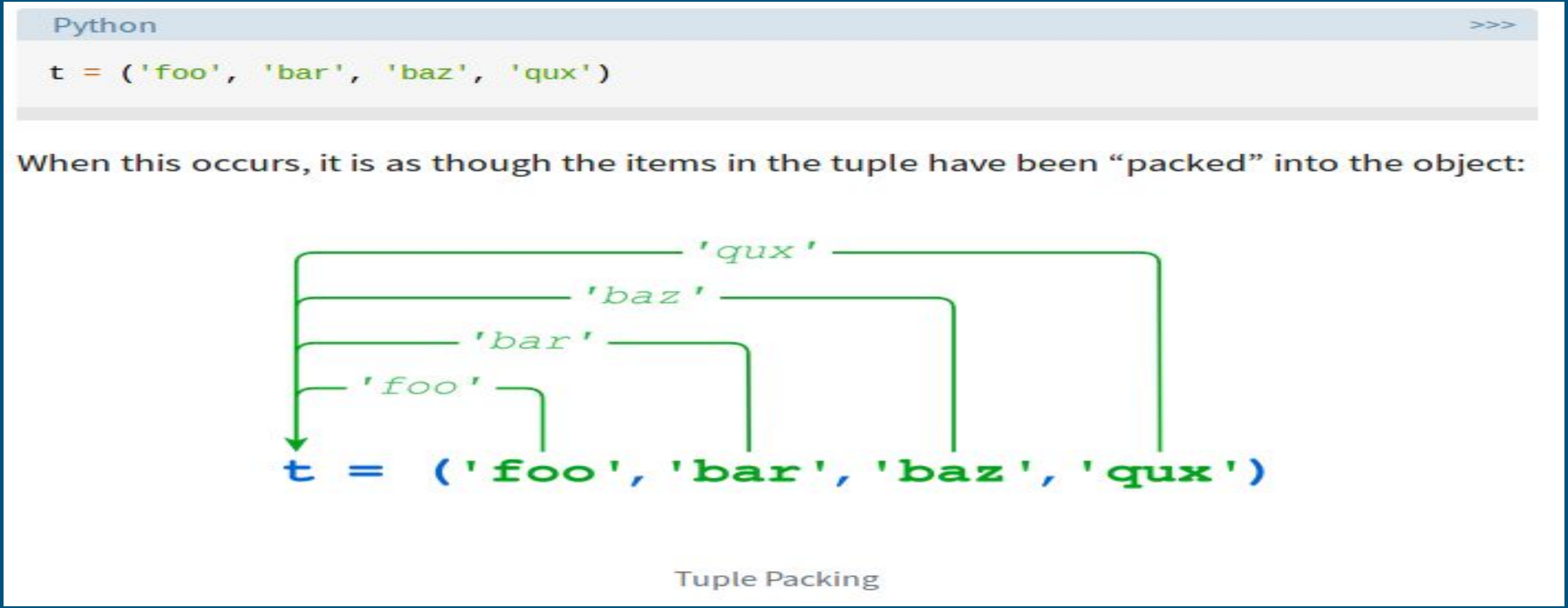


```
import queue


def main():

    Q = queue.deque()

    Q.append(4) # 4
    Q.append(41) # 4, 41
    Q.append(45) # 4, 41, 45
    Q.append(14) # 4, 41, 45, 14
    Q.popleft() # 41, 45, 14
    Q.popleft() # 45, 14
    Q.append(96) # 45, 14, 96

    print()
    print("----------------------")
    print(Q)
    print("----------------------")
    print()

    while Q.__len__() != 0:

        print(Q[0]) # peek value
        Q.popleft() # pop value one-by-one
        print(Q) # display whole queue after each pop


if __name__ == '__main__':
    main()
```

Learning Resources (List / Array)

1. https://developers.google.com/edu/python/lists
2. https://realpython.com/python-lists-tuples/ (*****)
3. https://www.geeksforgeeks.org/python-list/ (***)
4. http://thomas-cokelaer.info/tutorials/python/lists.html
5. https://www.w3schools.com/python/python_lists.asp
6. https://www.pythonforbeginners.com/lists/python-lists-cheat-sheet
7. https://www.programiz.com/python-programming/list

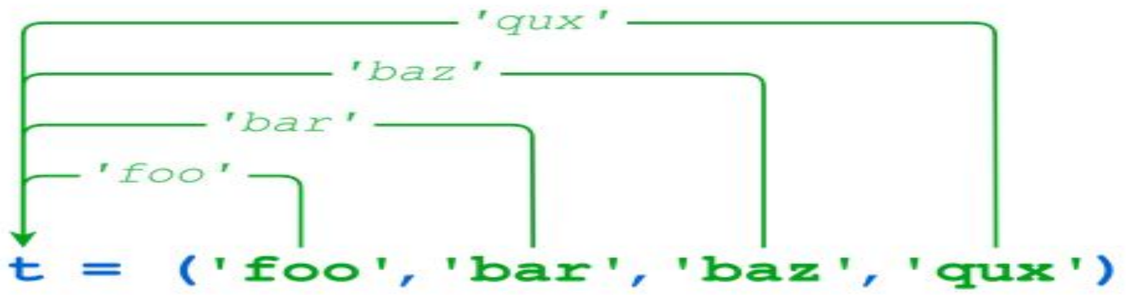
Tuple



Python >>>

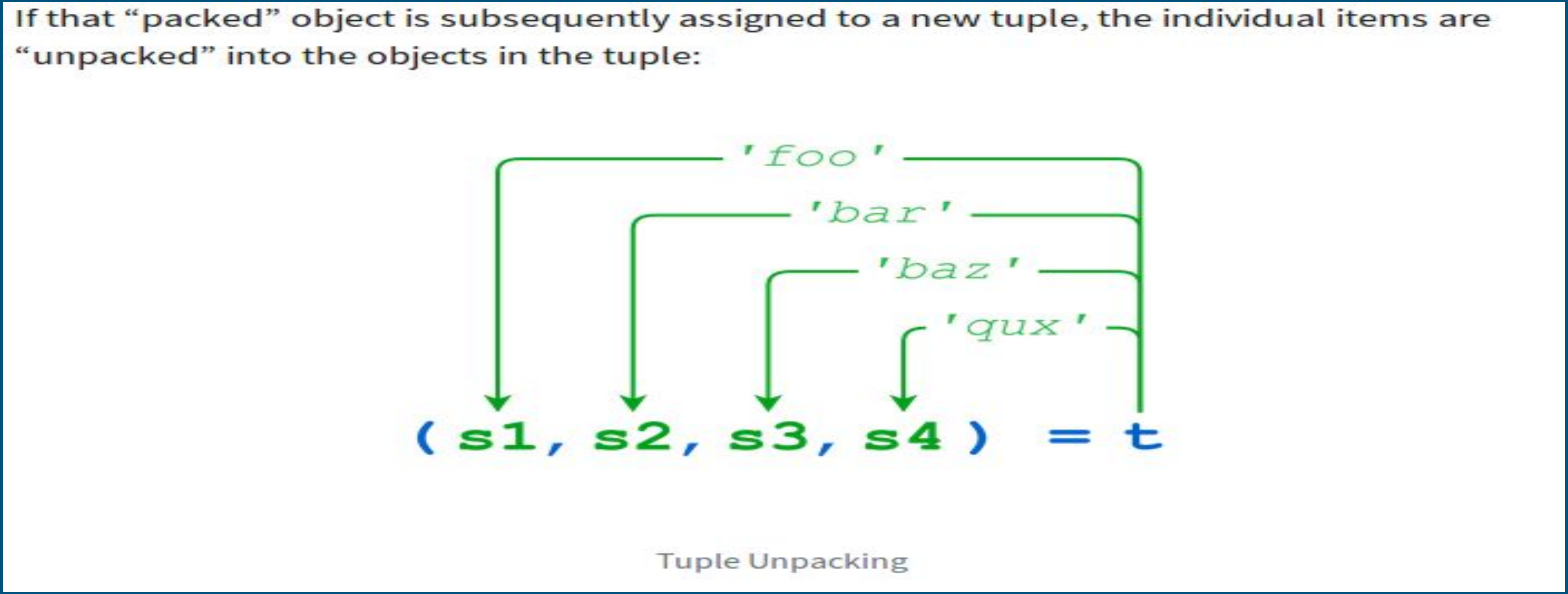
```python
t = ('foo', 'bar', 'baz', 'qux')
```

When this occurs, it is as though the items in the tuple have been “packed” into the object:

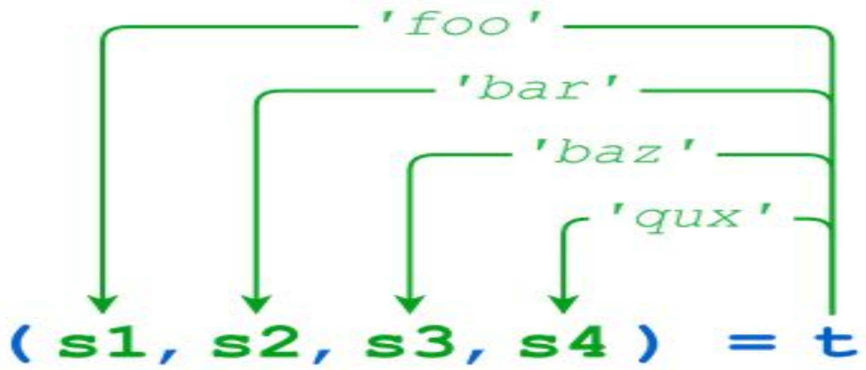


Tuple Packing

Tuple (Con.)

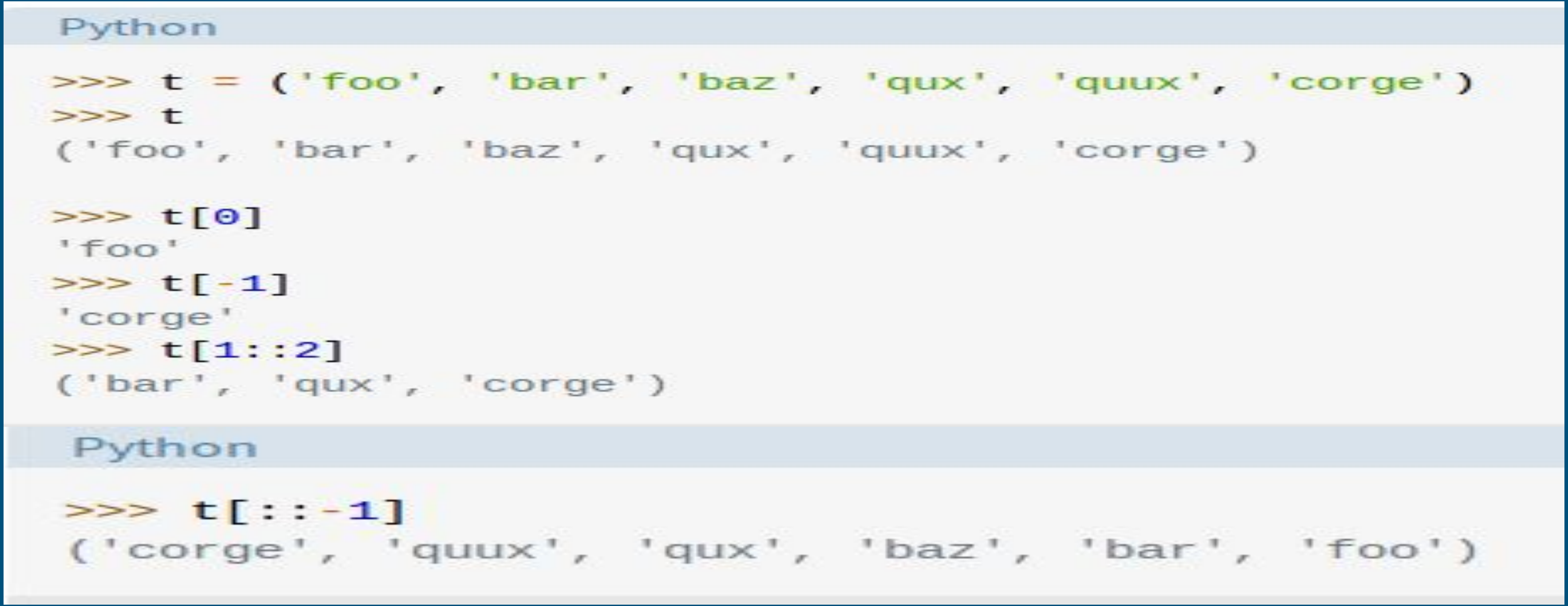


If that "packed" object is subsequently assigned to a new tuple, the individual items are "unpacked" into the objects in the tuple:



Tuple Unpacking

Tuple (Con.)

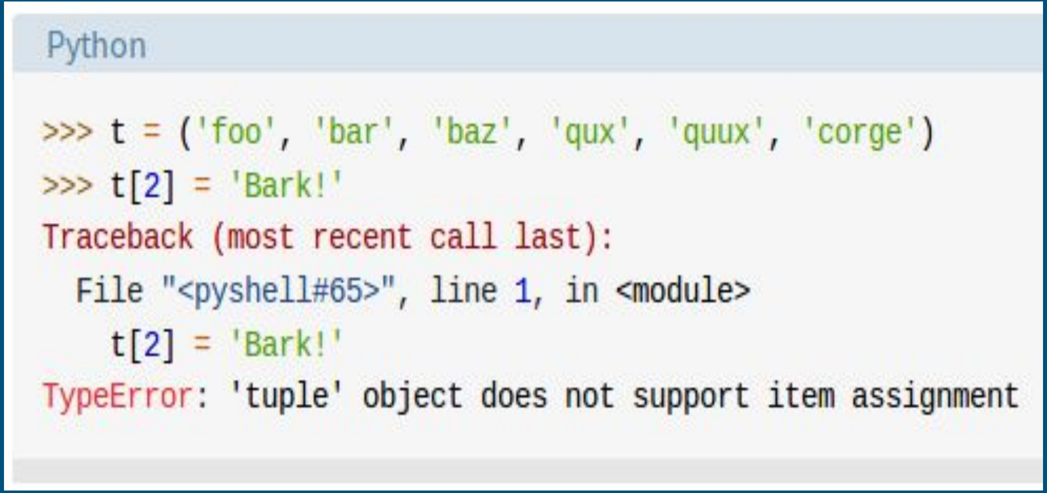

Python

```
>>> t = ('foo', 'bar', 'baz', 'qux', 'quux', 'corge')
>>> t
('foo', 'bar', 'baz', 'qux', 'quux', 'corge')

>>> t[0]
'foo'
>>> t[-1]
'corge'
>>> t[1::2]
('bar', 'qux', 'corge')
```

Python

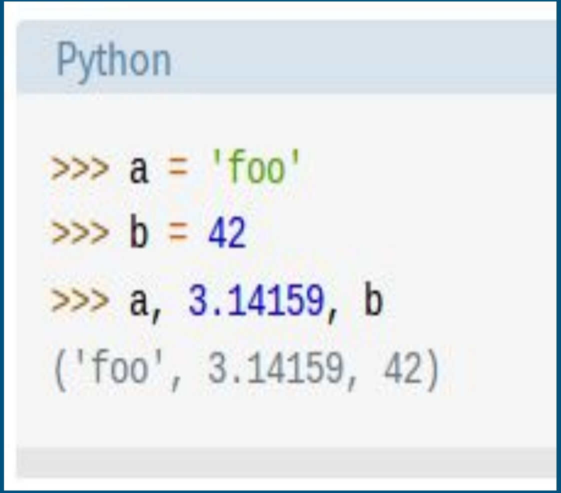
```
>>> t[::-1]
('corge', 'quux', 'qux', 'baz', 'bar', 'foo')
```

Tuple (Con.)



Python

```
>>> t = ('foo', 'bar', 'baz', 'qux', 'quux', 'corge')
>>> t[2] = 'Bark!'
Traceback (most recent call last):
  File "<pyshell#65>", line 1, in <module>
    t[2] = 'Bark!'
TypeError: 'tuple' object does not support item assignment
```



Python

```
>>> a = 'foo'
>>> b = 42
>>> a, 3.14159, b
('foo', 3.14159, 42)
```

Learning Resources (Tuple)

1. https://realpython.com/python-lists-tuples/ (*****)
2. https://www.geeksforgeeks.org/tuples-in-python (***)
3. https://www.tutorialspoint.com/python/python_tuples.htm
4. https://www.w3schools.com/python/python_tuples.asp

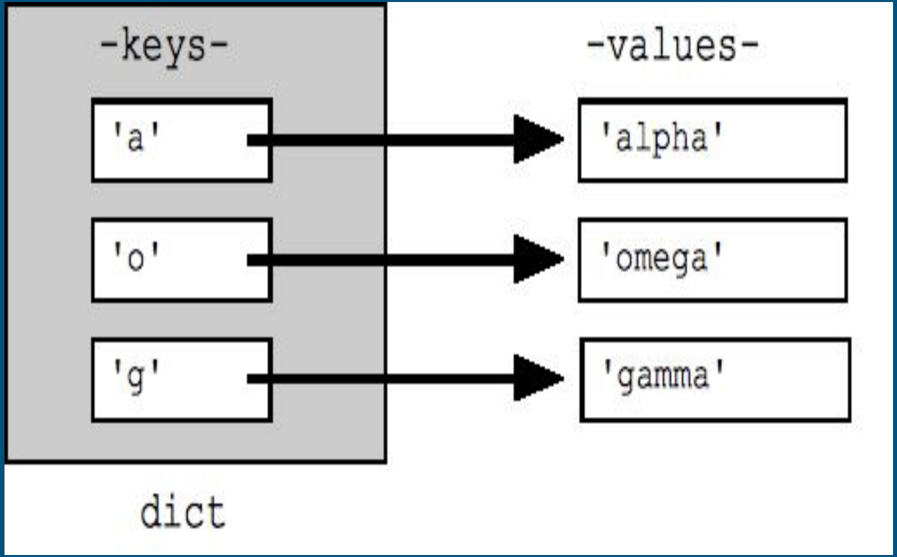
List vs. Tuple

1. https://www.programiz.com/python-programming/list-vs-tuples
2. https://www.afternerd.com/blog/difference-between-list-tuple/

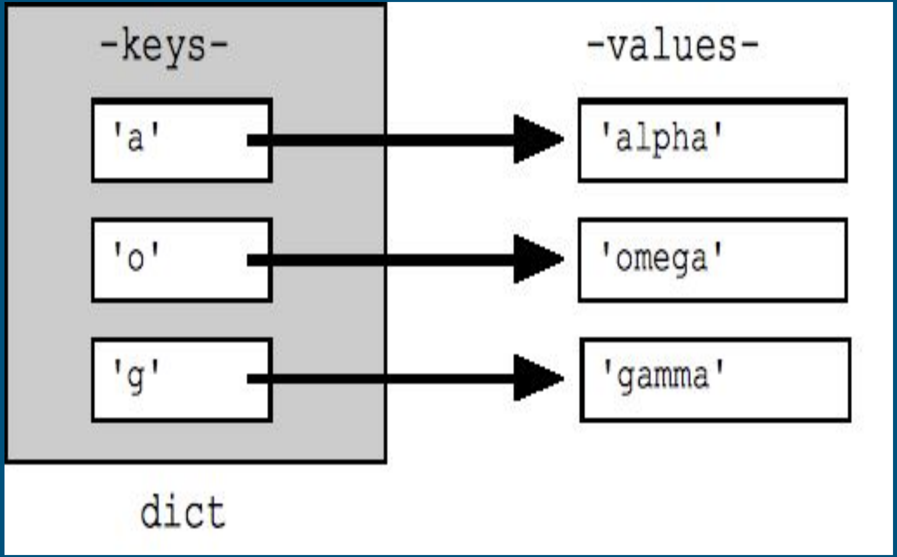
Sets

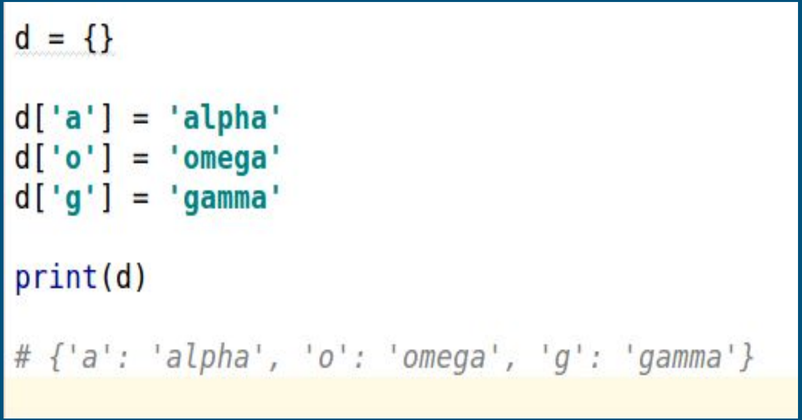
1. https://www.programiz.com/python-programming/set
2. https://realpython.com/python-sets/
3. https://www.python-course.eu/sets_frozensets.php
4. https://www.geeksforgeeks.org/sets-in-python/
5. https://snakify.org/en/lessons/sets/
6. Exercise: https://www.learnpython.org/en/Sets

Dictionary



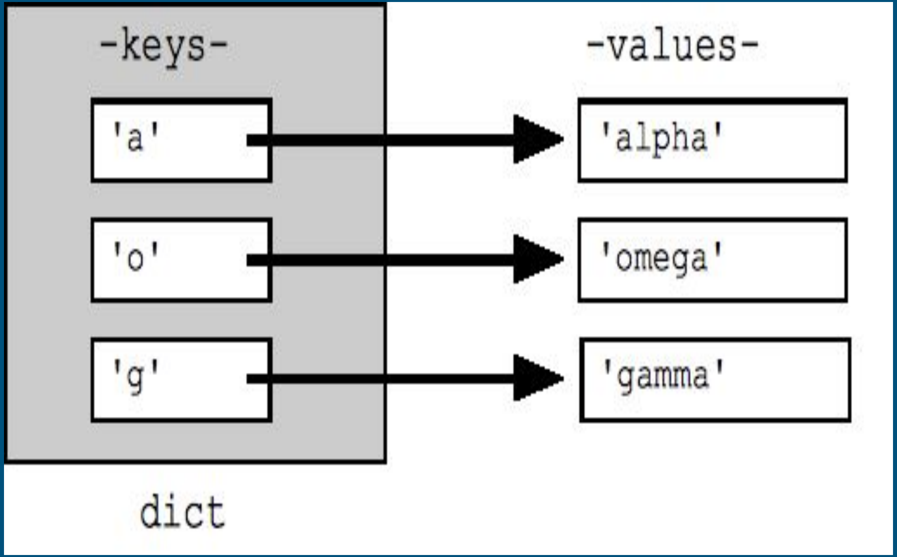
Dictionary (Con.)

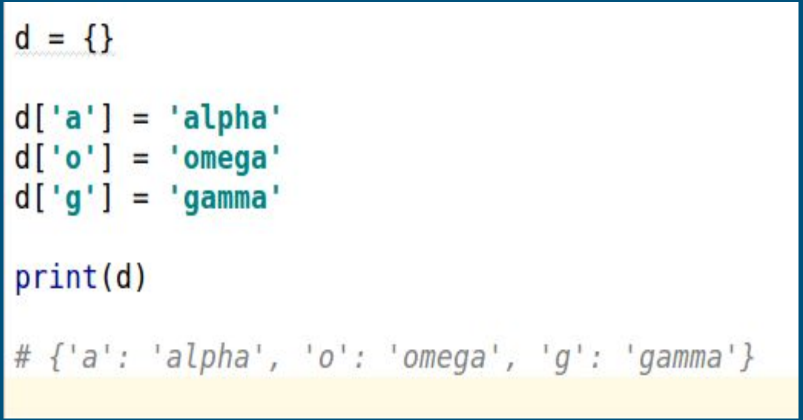




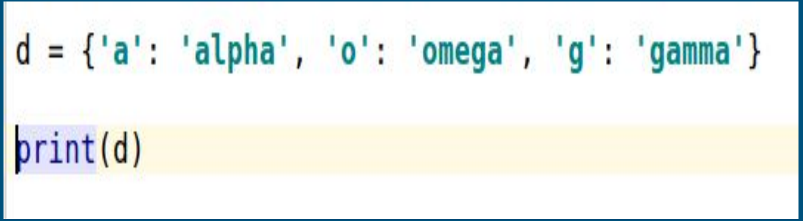
```
d = {}

d['a'] = 'alpha'
d['o'] = 'omega'
d['g'] = 'gamma'

print(d)

# {'a': 'alpha', 'o': 'omega', 'g': 'gamma'}
```

Dictionary (Con.)



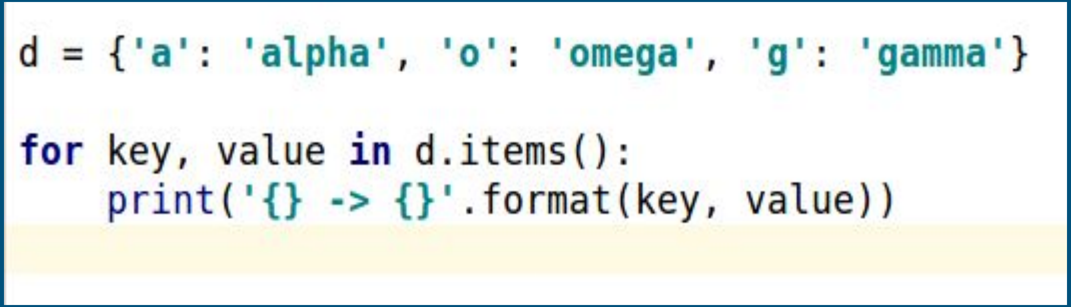


```
d = {}

d['a'] = 'alpha'
d['o'] = 'omega'
d['g'] = 'gamma'

print(d)

# {'a': 'alpha', 'o': 'omega', 'g': 'gamma'}
```



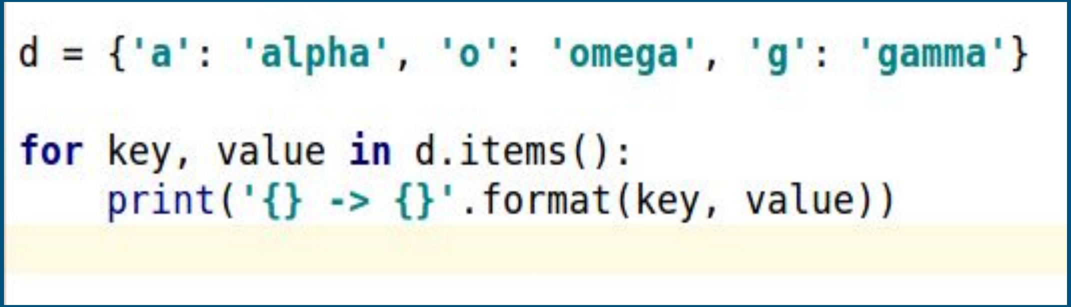
```
d = {'a': 'alpha', 'o': 'omega', 'g': 'gamma'}

print(d)
```

Dictionary (Con.)

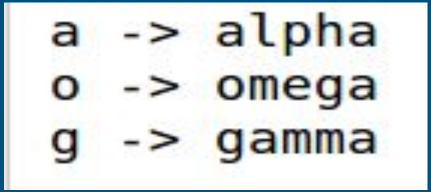


```
d = {'a': 'alpha', 'o': 'omega', 'g': 'gamma'}

for key, value in d.items():
    print('{} -> {}'.format(key, value))
```

Dictionary (Con.)

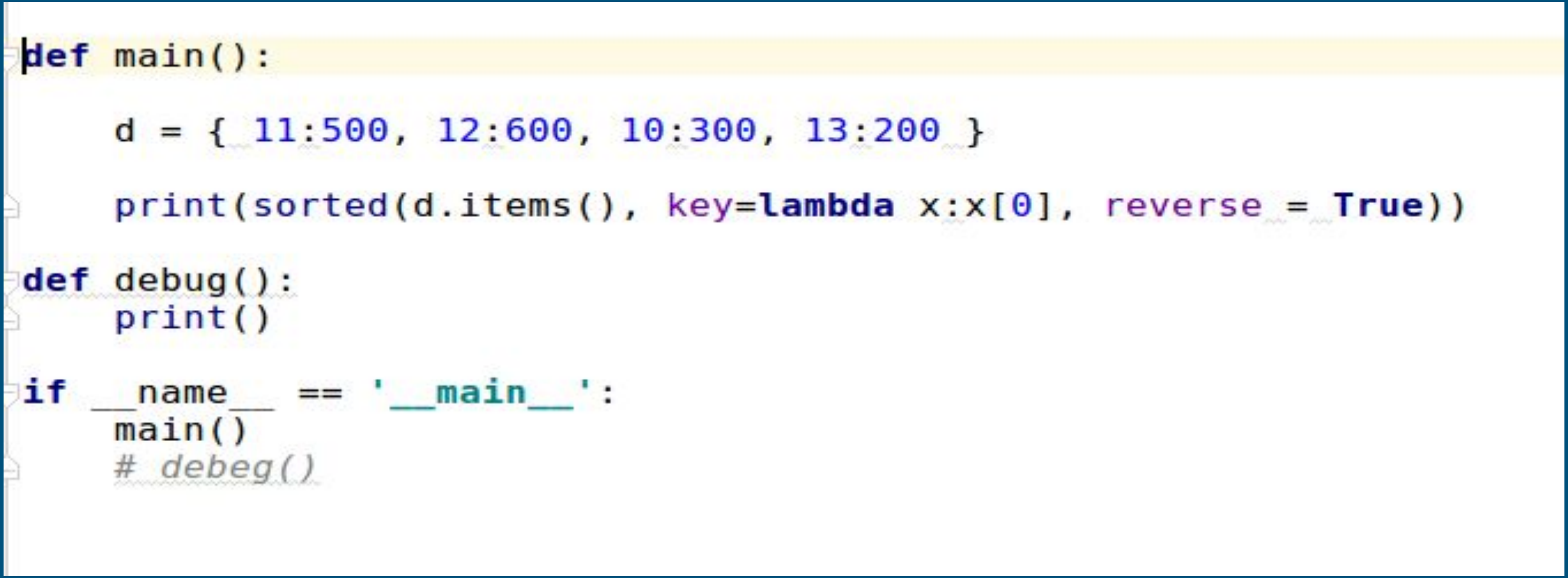


```
d = {'a': 'alpha', 'o': 'omega', 'g': 'gamma'}

for key, value in d.items():
    print('{} -> {}'.format(key, value))
```



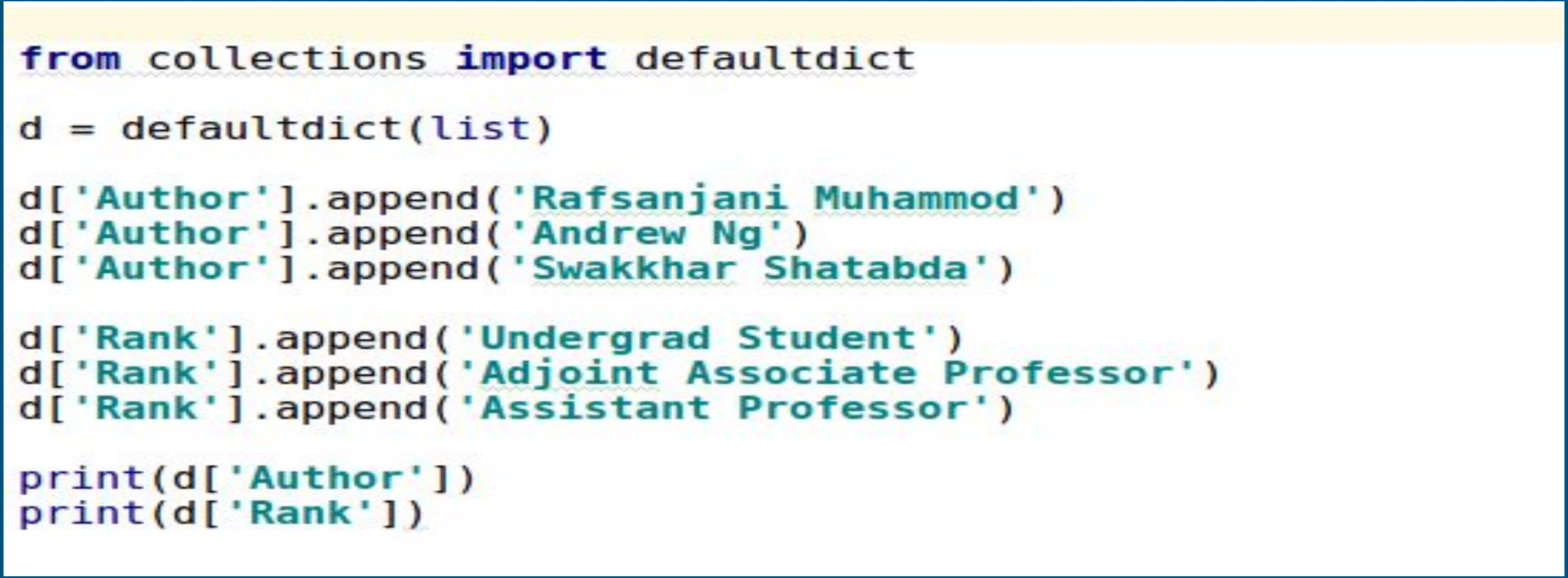
```
a -> alpha
o -> omega
g -> gamma
```

Dictionary (Con.)



```
def main():

    d = { 11:500, 12:600, 10:300, 13:200 }

    print(sorted(d.items(), key=lambda x:x[0], reverse = True))

def debug():
    print()

if __name__ == '__main__':
    main()
    # debeg()
```

Dictionary (Con.)



```
from collections import defaultdict

d = defaultdict(list)

d['Author'].append('Rafsanjani Muhammod')
d['Author'].append('Andrew Ng')
d['Author'].append('Swakkhar Shatabda')

d['Rank'].append('Undergrad Student')
d['Rank'].append('Adjoint Associate Professor')
d['Rank'].append('Assistant Professor')

print(d['Author'])
print(d['Rank'])
```

Learning Resources (Dictionary)

1. https://developers.google.com/edu/python/dict-files
2. https://www.programiz.com/python-programming/dictionary
3. https://realpython.com/python-dicts/
4. https://www.python-course.eu/dictionaries.php
5. https://www.geeksforgeeks.org/python-dictionary/

Contract your instructor!

Find Me: http://rafsanjani.pythonanywhere.com/contact

Course Website: https://mrzresearcharena.github.io/Big-Data-using-Python

Thank you!