lecture_22

October 16, 2022

1 Lecture 22

[1]:
```
d = {
    'a': 1,
    'b': False,
}
```

[2]:
```
d['a']
```

[2]:
```
1
```

[3]:
```
d['c']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Input In [3], in <cell line: 1>()
----> 1 d['c']

KeyError: 'c'
```

[4]:
```
d['c'] = 24
```

[5]:
```
d['c']
```

[5]:
```
24
```

[6]:
```
from collections import defaultdict
```

[7]:
```
d = defaultdict(int)
```

[8]:
```
d['a']
```

[8]:
```
0
```

[9]:
```
d['a'] = 42
```

[10]:
```
d['a']
```

[10]:
```
42
```

[11]:
```
d['b']
```

[11]:
```
0
```

[12]:
```
d['b'] = "test"
```

[13]:
```
d['b']
```

[13]:
```
'test'
```

[14]:
```
d['c']
```

[14]:
```
0
```

[15]:
```
qualified_students = defaultdict(bool)
```

[16]:
```
qualified_students['Henry']
```

[16]:
```
False
```

[17]:
```
qualified_students['Henry'] = {'a': 42}
```

[18]:
```
if qualified_students['Henry']:
    print(qualified_students['Henry'])
```

```
{'a': 42}
```

[19]:
```
d = defaultdict(list)
```

[20]:
```
d['a']
```

[20]:
```
[]
```

[21]:
```
p = {}
```

[22]:
```
p['a']
```

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
Input In [22], in <cell line: 1>()
----> 1 p['a']

KeyError: 'a'
```

[23]:
```
if 'a' not in p:
    p['a'] = []
```

[24]:
```
p['a'].append(42)
```

[25]:
```
d['a'].append(42)
```

1.1 Generators

[26]:
```
a = [1, 2, 12, 24, 42]
```

[27]:
```
b = iter(a)
```

[28]:
```
next(b)
```

[28]:
```
1
```

[31]:
```
next(b)
```

[31]:
```
2
```

[32]:
```
next(b)
```

[32]:
```
12
```

[33]:
```
next(b)
```

[33]:
```
24
```

[34]:
```
next(b)
```

[34]:
```
42
```

[35]:
```
next(b)
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Input In [35], in <cell line: 1>()
----> 1 next(b)

StopIteration:
```

[36]:
```
b
```

[36]:
```
<list_iterator at 0x1057a67f0>
```

[37]:
```
next(b)
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Input In [37], in <cell line: 1>()
----> 1 next(b)

StopIteration:
```

[38]:
```
def factorial(n):
    if n == 1 or n == 0:
        return 1
    return n * factorial(n-1)
```

[39]:
```
factorial(12)
```

[39]:
```
479001600
```

[40]:
```
def foo():
    a = 42
    yield a
    a += 24
    yield a
    a += 12
    yield a
    print("here")
    yield "Bye Bye"
```

[41]:
```
f = foo()
```

[42]:
```
f
```

[42]:
```
<generator object foo at 0x105aa97b0>
```

[43]:
```
next(f)
```

[43]:
```
42
```

[44]:
```
next(f)
```

[44]:
```
66
```

[45]:
```
next(f)
```

[45]:
```
78
```

[46]:
```
next(f)
```

```
here
```

[46]: 'Bye Bye'

[47]:
```
next(f)
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Input In [47], in <cell line: 1>()
----> 1 next(f)

StopIteration:
```

[48]:
```
iter(f)
```

[48]: <generator object foo at 0x105aa97b0>

[49]:
```
iter(f) is f
```

[49]: True

[50]:
```
for i in foo():
    print(i)
```

```
42
66
78
here
Bye Bye
```

[51]:
```
foo()
```

[51]: <generator object foo at 0x105aa9f90>

[52]:
```
map(lambda x: x*2, [1, 2, 3])
```

[52]: <map at 0x104226ee0>

[53]:
```
for i in map(lambda x: x*2, [1, 2, 3]):
    print(i)
```

```
2
4
6
```

[54]:
```
m = map(lambda x: x*2, [1, 2, 3])
```

[55]:
```
m
```

[55]:
```
<map at 0x10578bd00>
```

[56]:
```
iter(m)
```

[56]:
```
<map at 0x10578bd00>
```

[57]:
```
m is iter(m)
```

[57]:
```
True
```

[58]:
```
next(m)
```

[58]:
```
2
```

[59]:
```
next(m)
```

[59]:
```
4
```

[60]:
```
next(m)
```

[60]:
```
6
```

[61]:
```
next(m)
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Input In [61], in <cell line: 1>()
----> 1 next(m)

StopIteration:
```

[62]:
```
def foo():
    a = 42
    yield a
    a += 24
    yield a
    a += 12
    yield a
    print("here")
    yield "Bye Bye"
```

[63]:
```
def foo(name):
    a = 42
    yield a
    a += 24
    yield a
    a += 12
    yield a
    print(name)
    yield "Bye Bye"
```

[65]:
```
for i in foo("Adam"):
    print(i)
```

```
42
66
78
Adam
Bye Bye
```

[69]:
```
from itertools import count as infinite_counter
```

[70]:
```
def factorial():
    res = 1
    for i in infinite_counter(2):
        yield res
        res *= i
```

[78]:
```
def factorial(n):
    res = 1
    for i in range(1, n+1):
        yield res
        res *= i
```

[82]:
```
for idx, val in enumerate(factorial(20)):
    print(idx, val)
```

```
0 1
1 1
2 2
3 6
4 24
5 120
6 720
7 5040
8 40320
9 362880
10 3628800
```

```
11 39916800
12 479001600
13 6227020800
14 87178291200
15 1307674368000
16 20922789888000
17 355687428096000
18 6402373705728000
19 121645100408832000
```

[83]:
```
a = range(10)
```

[84]:
```
a
```

[84]:
```
range(0, 10)
```

[85]:
```
iter(a) is a
```

[85]:
```
False
```

[87]:
```
r = iter(a)
```

[88]:
```
next(r)
```

[88]:
```
0
```

[89]:
```
next(r)
```

[89]:
```
1
```

[90]:
```
def foo():
    n = 42
    yield n
    n += 12
    yield n
    n += 24
    return n
```

[91]:
```
a = foo()
```

[92]:
```
a
```

[92]:
```
<generator object foo at 0x10415f190>
```

[93]:
```
for i in foo():
    print(i)
```

```
42
54
```

[97]:
```
next(a)
```

[97]:
```
42
```

[98]:
```
next(a)
```

[98]:
```
54
```

[99]:
```
next(a)
```

```
---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
Input In [99], in <cell line: 1>()
----> 1 next(a)

StopIteration: 78
```

[100]:
```
a
```

[100]:
```
<generator object foo at 0x10415f190>
```

[102]:
```
for i in a:
    print(i)
```

[104]:
```
a = [1, 2, 12, 24, 42]
```

[105]:
```
[i**2 for i in a]
```

[105]:
```
[1, 4, 144, 576, 1764]
```

[106]:
```
def power_maker(numbers):
    for i in numbers:
        yield i**2
```

[107]:
```
for i in power_maker(a):
    print(i)
```

```
1
4
144
576
1764
```

[108]:
```
for i in map(lambda x: x**2, a):
    print(i)
```

```
1
4
144
576
1764
```

[109]:
```
for i in [e**2 for e in a]:
    print(i)
```

```
1
4
144
576
1764
```

[111]:
```
import sys
```

[144]:
```
from random import randint
```

[145]:
```
a = [randint(1, 100) for _ in range(1000)]
```

[147]:
```
sys.getsizeof(a)
```

[147]: 8856

[148]:
```
o_1 = [i**2 for i in a]
```

[149]:
```
sys.getsizeof(o_1)
```

[149]: 8856

[150]:
```
o_2 = power_maker(a)
```

[151]:
```
sys.getsizeof(o_2)
```

[151]: 112

[152]:
```
o_3 = map(lambda x: x**2, a)
```

[154]:
```
sys.getsizeof(o_3)
```

[154]: 48

[153]:
```
o_4 = (i**2 for i in a)
```

[155]:
```
sys.getsizeof(o_4)
```

[155]: 112

[157]:
```
for i in range(1, 20, 2):
    print(i)
```

```
1
3
5
7
9
11
13
15
17
19
```

[158]:
```
for i in range(20):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

[159]:
```
def our_range(start, stop=None, step=1):
    if stop is None:
        start, stop = 0, start
    while start < stop:
        yield start
```

```
        start += step
```

[160]:
```
for i in our_range(1, 20, 2):
    print(i)
```

```
1
3
5
7
9
11
13
15
17
19
```

[162]:
```
for i in our_range(20):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
```

[163]:
```
def our_count(n, step=1):
    while True:
        yield n
        n += step
```

[164]:
```
for i, e in enumerate(our_count(20, 2)):
    print(e)
```

```
    if i == 20:
        break
```

```
20
22
24
26
28
30
32
34
36
38
40
42
44
46
48
50
52
54
56
58
60
```

[171]:
```
iterables = ([1, 2, 3], [1, 2, 3, 4], [2, 3, 4], [1, 2, 3, 4, 5], [1])

min(len(i) for i in iterables)
```

[171]: 1

[172]:
```
def our_zip(*iterables):
    min_len = min(len(i) for i in iterables)

    for i in our_range(min_len):
        yield tuple(iterable[i] for iterable in iterables)
```

[173]:
```
for x, y in our_zip([1, 2, 3], [4, 5, 6, 7]):
    print(x, y)
```

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

[174]:
```
def our_map(func, *iterables):
    for params in zip(*iterables):
        yield func(*params)
```

[175]:
```
for i in our_map(pow, [1, 2, 3], [4, 5, 6, 7]):
    print(i)
```

```
1
32
729
```

[177]:
```
def our_filter(func, iterable):
    for param in iterable:
        if func(param):
            yield param
```

[180]:
```
for i in our_filter(lambda x: x % 2 == 0, [1, 2, 3, 4, 5]):
    print(i)
```

```
2
4
```

[ ]:
```

```