In [1]:
```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

PYTHON 3

Функции, генераторы, декораторы

MIPT 2020

Еще немного о **collections**

In [2]:
```
from collections import OrderedDict

d = OrderedDict([(1, 3), (3, 1), (2, 2)])
dict(d)
d.popitem(last=False)
d.popitem()
d.popitem()
d.popitem()
```

Out[2]: {1: 3, 3: 1, 2: 2}

Out[2]: (1, 3)

Out[2]: (2, 2)

Out[2]: (3, 1)

```
---------------------------------------------------------------------------
KeyError                                  Traceback (most recent call last)
<ipython-input-2-586f29d7e993> in <module>
      6 d.popitem()
      7 d.popitem()
----> 8 d.popitem()

KeyError: 'dictionary is empty'
```

In [3]:
```
d = dict([(1, 3), (3, 1), (2, 2)])

for k, v in d.items():
    k, v
```

Out[3]: (1, 3)

Out[3]: (3, 1)

Out[3]: (2, 2)

Из документации:

Performing list(d) on a dictionary returns a list of all the keys used in the dictionary, in insertion order

Функции

Аргументы

In [4]:
```
def hello():
    print("hello world")

hello()
```

```
hello world
```

In [5]:
```
def i_return_two_plus_x(x: int) -> int:  # just for static linters
    return 2 + x

i_return_two_plus_x(4)
```

Out[5]:
```
6
```

In [6]:
```
def some_args(a, b, c, d=2, e=4):
    return a + b + c + d + e

some_args(1, 2, 3)
some_args(1, 2, 3, 4, 5)
some_args(1, 2, c=3, d=4, e=5)
some_args(1, 2, 3, 5, d=7)
```

Out[6]:
```
12
```

Out[6]:
```
15
```

Out[6]:
```
15
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-6-e95701289b87> in <module>
      5 some_args(1, 2, 3, 4, 5)
      6 some_args(1, 2, c=3, d=4, e=5)
----> 7 some_args(1, 2, 3, 5, d=7)

TypeError: some_args() got multiple values for argument 'd'
```

In [7]:
```
def more_args(a, b=3, *, d, c=3):
    return a + b + c + d

more_args(1, b=2, c=3, d=4)
more_args(1, c=2, d=1)
more_args(1, 2, 3, 4)
```

Out[7]:
```
10
```

Out[7]:
```
7
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-7-ae7f756f875e> in <module>
      4 more_args(1, b=2, c=3, d=4)
      5 more_args(1, c=2, d=1)
----> 6 more_args(1, 2, 3, 4)

TypeError: more_args() takes from 1 to 2 positional arguments but 4 were given
```

In [8]:
```
# python3.8
def args(a, /, b, *, c):
    ...
```

In [9]:
```
def f(x):
    return x

def f(x, y):
    return x, y

f(1, 2)
f(1)
```

Out[9]:
```
(1, 2)

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-9-302dc7c40a5c> in <module>
      6
      7 f(1, 2)
----> 8 f(1)

TypeError: f() missing 1 required positional argument: 'y'
```

In [10]:
```
def anything(f, *args, **kwargs):
    print(args)
    print(kwargs, kwargs['a'], sep='\n')

anything(1, 2, 3, a=4, b=5)
```

```
(2, 3)
{'a': 4, 'b': 5}
4
```

In [11]:
```
def test_mutability(x):
    x.append(2)
    print(id(x))
    x = 173
    print(id(x))

x = [1]
id(x)
test_mutability(x)
id(x)
x
```

Out[11]:
```
139743454915648
```

```
139743454915648
11502144
```

Out[11]:
```
139743454915648
```

Out[11]:
```
[1, 2]
```

Звездочка **только** распаковывает iterable в аргументы функции

In [12]:
```
def so_many_args(a, b, c, d, e, f, g, h, i, *args, j=0, k=0):
    return a + b + c + d + e + f + g + h + i + j + k

so_many_args(*[1, 2, 3, 4, 5, 6, 7, 8, 9], **{'j': 10, 'k': 11})
```

Out[12]:
```
66
```

Рекурсия

In [13]:
```python
import sys

sys.setrecursionlimit(3000)

def fact(n):
    if n <= 1:
        return 1
    return n * fact(n - 1)

fact(10)
fact(100)
fact(2950)
```

Out[13]:
```
3628800
```

Out[13]:
```
9332621544394415268169923885626670049071596826438162146859296389521759999322991
5608941463976156518286253697920827223758251185210916864000000000000000000000000
```

Out[13]:
```
8714424896231723533236416994855066399501649438061404064312046054708657597802511
3094005316813387253233918824761027823220193224411432506537861735630651739690844
5480199457125827431878909595337211956991823884581123194940201472281909989825218
4822267366266441575822077543043363318103464494099394030992460462060990245603269
7423471842607702869889993931638206179934969400770095329575996340043050371531159
5884864409713551053373046977135758116074641798793932004246095412192497926880171
9874178741189533269077079538306104205481993227539587540596638971391075752927631
4956708640058616750861299719225853231172377711344581605022107094346459639472229
6153998444202069304114419266833196432014943315616307939255051449481389651070428
5993433131249713569516355805314348401946916377392814806279690076280346857615514
0846093304215833940256882313656130255935841986124053880273129010591807353576256
0720401734116002001563178647294266540768625302835568429505597711814971218596694
8138875658190266987357692570485398881568900862921459226675530945729818261453139
6381697582671137306843013261900760019964494514455622366948394012157861913526910
2799292271140283896077614936780976854623150907766077131866958993889088075304733
6516989676333241692878891056662319295338099548901525255568874722996757007525712
4148089981412338439103014824866115851078608141404411928236488742026382712282520
2543242159576875208209193504004795535319316677955099702666830701655755599769380
9652278980816149896284988924021997235909036024919080417675158179023341703844258
3167785643752362146477888967346265832924825935280469297459611870569361765843847
2904441118641443030290982389541016596003896437989969323302415389676566567027537
8742056682500408701782522877260098521455849925612510095069425901915901779119521
8426874503459702469299590864658119839753677543377484462317338993831355286915196
0919313486266421894428221008309742655396435716843093679907885657254780858357132
5633571104610506318616058688404737492487679371145345149875298185780107268462706
4787718859626210820444254588084049291027402994449318170317262256693451131586901
2941244926229209949056086840379719435152210578689616069282600458490773623354260
7661826567964441925998462642526622266257230642067372461251128323333174667243099
0637081325591212188480074398815071564203054900704583608367821098639642520325875
2589790201041926171026857005240075671951015557420302413311857428473728006907397
3598283248000746532700768885608889554089595702986124007764139600695160095234575
5149706287866852312409192640436676622911081664421299730494546788189812643918935
4353909488052468580023030848455081865916421650798807278980874754219181832287543
6353677071852304494390622014927407595893476965111301901873926092446801911390226
2074879161559765816263630784345554321501203820058876903161279092620614511699002
4859193454772773537048767055930373653809803711781396595125346804224556408476172
0186952933045854771312345400238295240297234362802496702285105666174899954486082
2182129604620170708858603366799016857818231471025444800260134195370492639596594
8449477579002121393663613039648048018334662457962810530516814487147457077057871
4698548585679214201971976904194208471306439699707413677369212834111962391140740
0620733065226955156882356070436202968608084269737414907053232343790331745651625
4542275791469100556211001982437662734318539846896937202405369816972914613713353
6615475841330796440664915286748745630714164439398763673696338195456012604922750
3794296420098108955228376299263478878968926409618388806650296400482035722790836
0060613824994661433741064205934753347977772869698294241949730160060967540080679
5416199355803925126409829942803565210779891114430814607146761220487433293146044
4891461716608761312320316045627874930228095452330184399976943159663703564604887
7197809707959117627133011398355147741830134894890623142348696282548712922130694
9541035787627686247273102369977590260627549309933802759503068501284984963377875
9139135588082149222765206174152880443267999617573754697338469017122818151074073
```

In [14]:
```
sys.getrecursionlimit()
sys.setrecursionlimit(100)
```

Out[14]:
```
3000
```

In [15]:
```
fact(200)
```

```
---------------------------------------------------------------------------
RecursionError                            Traceback (most recent call last)
<ipython-input-15-327a90dad84d> in <module>
----> 1 fact(200)

<ipython-input-13-32caae5ecee4> in fact(n)
      6     if n <= 1:
      7         return 1
----> 8     return n * fact(n - 1)
      9
     10 fact(10)

<ipython-input-13-32caae5ecee4> in fact(n)
      6     if n <= 1:
      7         return 1
----> 8     return n * fact(n - 1)
      9
     10 fact(10)

<ipython-input-13-32caae5ecee4> in fact(n)
```

In [16]:
```
def f(n):
    if n < 2:
        return n
    return g(n - 1) + 1

def g(n):
    if n < 2:
        return n
    return f(n - 1) + 2

f(5)
```

Out[16]:
```
7
```

Генераторы

In [17]:
```
def squared(a):
    for x in a:
        yield x ** 2

a = list(range(10))
for elem in squared(a):
    print(elem, end=' ')
```

```
0 1 4 9 16 25 36 49 64 81
```

In [18]:
```
func = squared
func
generator = func(a)
generator
next(generator, None)
```

Out[18]: <function __main__.squared(a)>

Out[18]: <generator object squared at 0x7f188efdd820>

Out[18]: 0

In [19]:
```
while True:
    print(next(generator), end=' ')
```

```
1 4 9 16 25 36 49 64 81

---------------------------------------------------------------------------
StopIteration                             Traceback (most recent call last)
<ipython-input-19-e51527406cfe> in <module>
      1 while True:
----> 2     print(next(generator), end=' ')

StopIteration:
```

In [20]:
```
generator = func(a)
while True:
    try:
        elem = next(generator)
        # loop body
        print(elem, end=' ')
    except StopIteration:
        break
```

```
0 1 4 9 16 25 36 49 64 81
```

In [21]:
```
def yielder3():
    for i in range(10):
        yield i

def yielder2():
    for elem in yielder3():
        yield elem

def yielder1():
    yield from yielder2()  # Передает управление второму генератору,
                           # пока он не исчерпается

for x in yielder1():
    print(x, end=' ')
```

```
0 1 2 3 4 5 6 7 8 9
```

In [22]:
```
def cumsum():
    csum = 0.0
    while True:
        csum += yield csum  # Вообще говоря это уже корутина

arr = [1.0 + x / 2 for x in range(10)]
arr
gen = cumsum()
gen
gen.send(None) # equal to next(gen)
for elem in arr:
    print(gen.send(elem), end=' ')

for elem in gen:
    print("Can reach?")
```

Out[22]:
```
[1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5]
```

Out[22]:
```
<generator object cumsum at 0x7f188efddb30>
```

Out[22]:
```
0.0
```

```
1.0 2.5 4.5 7.0 10.0 13.5 17.5 22.0 27.0 32.5

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-22-113f639d2730> in <module>
     12     print(gen.send(elem), end=' ')
     13
---> 14 for elem in gen:
     15     print("Can reach?")

<ipython-input-22-113f639d2730> in cumsum()
      2     csum = 0.0
      3     while True:
----> 4         csum += yield csum  # Вообще говоря это уже корутина
      5
      6 arr = [1.0 + x / 2 for x in range(10)]

TypeError: unsupported operand type(s) for +=: 'float' and 'NoneType'
```

Области видимости

In [23]:
```
x = 2  # global

def func(a):
    x = 4  # local
    return a + x

func(1)
x
```

Out[23]:
```
5
```

Out[23]:
```
2
```

In [24]:
```
hw = 'hello world'
def hello():
    global hw  # можно явно указать и менять отсюда
    hw = 'broken, fix pls'
    print(hw)

hello()
hw

def hello2():
    global hw
    hw = 'fixed, dont worry'

hello2()
hw

def hello2():
    nonlocal hw
    hw = 'fixed, dont worry'
```

```
broken, fix pls
```

Out[24]:
```
'broken, fix pls'
```

Out[24]:
```
'fixed, dont worry'
```

```
  File "<ipython-input-24-52da261a4c65>", line 18
    nonlocal hw
    ^
SyntaxError: no binding for nonlocal 'hw' found
```

Замыкания

In [25]:
```
def outer(x):
    z = x
    def inner(y):
        nonlocal z
        return z * y
    return inner

multiply_by_3 = outer(3)
multiply_by_3(4)
outer(3)(4)
```

Out[25]:
```
12
```

Out[25]:
```
12
```

Функция тоже объект!

In [26]:
```
multiply_by = [outer(x) for x in range(10)]
multiply_by[3](5)
multiply_by[6](10)
```

Out[26]:
```
15
```

Out[26]:
```
60
```

Декораторы

In [27]:
```
def salad(f):
    def internal(*args):
        print("salad")
        f(*args)
    return internal

def bread(f):
    def internal():
        print("bread")
        f()
        print("bread")
    return internal

def tomato(f):
    def internal():
        print("tomato")
        f()
    return internal

def f():
    print("beef")

sandwich = bread(salad(tomato(f)))
sandwich()
```

```
bread
salad
tomato
beef
bread
```

Так люди придумали декораторы

In [28]:
```
@bread
@salad
@tomato
def g():
    print("chicken")

g()
# то же самое что g = bread(salad(tomato(g)))
```

```
bread
salad
tomato
chicken
bread
```

In [29]:
```
g.__name__, g.__repr__()
```

Out[29]:
```
('internal', '<function bread.<locals>.internal at 0x7f188efe7ee0>')
```

Лучше не затирать эти поля

In [30]:
```
from functools import wraps

def bread(h):
    @wraps(h)
    def internal():
        print("bread")
        h()
        print("bread")
    return internal

@bread
def h():
    print("fish")

h()
h.__name__, h.__repr__()
```

```
bread
fish
bread
```

Out[30]:
```
('h', '<function h at 0x7f188ee61940>')
```

Лямбда-функции

In [31]:
```
applier1 = lambda f, x: f(x)

def apply(f, x):
    return f(x)

applier2 = apply

applier1(sum, [1, 2, 3])
applier2(sum, [1, 2, 3])
```

Out[31]:
```
6
```

Out[31]:
```
6
```

In [32]:
```
a = [(1, 5), (3, 1), (7, 20), (9, 11)]
sorted(a, key=sum)
```

Out[32]:
```
[(3, 1), (1, 5), (9, 11), (7, 20)]
```

Summary

In [33]:
```
# function
def f():
    return 'f'
f.__call__()
f.__call__ = lambda: 'f new'
f.__call__()

# generator
def g():  # function
    yield 'g'
it = g.__call__()  # get generator
it.__next__()  # iterate

# coroutine
def h():
    s = yield 'h'
    yield s
it = h.__call__() # get coroutine
it.__next__() # first iteration
it.send('hey')
```

Out[33]: 'f'

Out[33]: 'f new'

Out[33]: 'g'

Out[33]: 'h'

Out[33]: 'hey'

In [34]:
```
dir(f)
```

Out[34]:
```
['__annotations__',
 '__call__',
 '__class__',
 '__closure__',
 '__code__',
 '__defaults__',
 '__delattr__',
 '__dict__',
 '__dir__',
 '__doc__',
 '__eq__',
 '__format__',
 '__ge__',
 '__get__',
 '__getattribute__',
 '__globals__',
 '__gt__',
 '__hash__',
 '__init__',
 '__init_subclass__',
 '__kwdefaults__',
 '__le__',
 '__lt__',
 '__module__',
 '__name__',
 '__ne__',
 '__new__',
 '__qualname__',
 '__reduce__',
 '__reduce_ex__',
 '__repr__',
 '__setattr__',
 '__sizeof__',
 '__str__',
 '__subclasshook__']
```

In [35]:
```
f.__code__
```

Out[35]:
```
<code object f at 0x7f188ee633a0, file "<ipython-input-33-76ad423ce8de>", line
2>
```