Язык Python.
Описание базовых конструкций.

Евгений Борисов

Python: **типы данных**

Логические

Числовые

Строки

Списки

Множества

Словари

None

Python: **тип данных логический**

Boolean Type:

True

False

Python: **типы данных числовые**

Numeric Type:

int – целое число

7

float – число с плавающей точкой

7.5, 75e-1

Python: **тип данных строки**

Text Sequence Type

'привет'

"медвед"

' ' 'превед
Медвед' ' '

Python: **типы данных списки**

Sequence Type:

list – список

[1, 2, 'a', [4,'a', 5,] ,]

tuple – кортеж

(1, 2, 'a',)

Python: **типы данных множества**

Set Types:

set – множество

set([1,2,2,3,4,2,3,4]) → {1,2,3,4}

frozenset – неизменяемое множество

Python: **типы данных словарь**

Mapping Types:

dict – словарь

{'a':1, 'b':2, 'zzz':7,}

Python: изменяемые типы данных

всё есть объекты

присваивание создаёт новый объект

immutable:
int float bool string tuple frozenset

mutable:
list dict set

Python: операции

Операции с данными: арифметические, логические, строковые, битовые

управление

циклы

ввод / вывод

Python: операции с данными

присваивание, арифметика и сравнения

```
a,b = 1,2
a,b = b,a
  a = 10
  a += 7
```

```
a / b
a // 3
a % 3
a - b
a + b
a * b
a**2
```

```
a<10
b<=7
a>2
a!=b
a==1
```

Python: операции с данными

логические

a = True
b = False

a or b
a and b
not b

Python: операции с данными

битовые

```
a = 255
b = 7

a^b
a&b
a|b
a>>3
```

Python: операции с данными

строковые

s = 'abc'

s*3 → 'abcabcabc'

s + 'dmr' → 'abcdmr'

Python: операции управления

```
if not x:
    print('x')
elif y:
    print('y')
else:
    print('z')
```

отступ в качестве операторных скобок

Python: цикл while

```
i=0
while i<5:
    print(i)
    i+=1
```

```
i=0
while i<5:
    i+=1
    if i<3:
        continue
    print(i)
```

```
i=0
while True:
    print(i)
    i+=1
    if i>5:
        break
```

Python: цикл for

```
for x in [1,2,3,4]:
    print(x)
```

Python: списки (list)

s=[1,7,3,4,['a','b']]

s=[1,5,3,4,]

len(s) **sorted**(s)

s[2] s[2:] s[2:4]

2 **in** s

s.**append**(9)

s.**insert**(5,'a')

s.**index**(2)

s = **list**(**range**(10))

s = [i/2 **for** i **in range**(10) **if** i!=3]

Python: кортежи (tuple)

```
c = (1,2,3,5)
```

Python: словари (dict)

```
d = { 'a':1, 'b':44, 'c':45, 'cvc':-1,  }

d['c']→ 45

d.keys()      d.values()
```

Python: множества (set)

s = **set**([1,2,3,1,3,4,5])

{1,2,3,4,5}

s[2] → error

операции: & | -

Python: менеджер контекстов (with)

```
with open('temp.txt','r') as f:
    x = f.read()
```

```
with open('temp.txt','r') as f:
    x = [ s for s in f.read().split('\n') if s ]
```

Python: функции

```
def myfunc(x,y=1):
    print(x)
    return x+1,y/2
```

возможность определения значения параметра по умолчанию

```
a,b = myfunc(y=5,x=-1)
```

возможность именования параметров

Python: итераторы

объект перечислитель

реализует навигацию по элементам другого объекта

выдаёт следующий элемент __next__()

если элементов больше нет
то «бросает» исключение

```
s='abcdef'
it_s = iter(s)
it_s.__next__()
for c in it_s:
    print(c)
```

```
s='abcdef'
for c in s:
    print(c)
```

Python: генераторы

генерируем последовательность

```
def ones(n):
    while n > 0:
        n -= 1
        yield 1

for o in ones(4):
    print(o)
```

Python: функциональное программирование

```
squares = map(lambda x: x * x, [0, 1, 2, 3, 4])

sum = reduce(lambda a, x: a + x, [0, 1, 2, 3, 4])
```

Python: ООП

```
class Animal:
  def __init__(self, name, color):
    self.name = name
    self.color = color
```

```
class Wolf(Animal):
  def bark(self):
    print("Grr...!")
```

```
class Dog(Wolf):
  def bark(self):
    super().bark()
    print("Woof!")
  def __repr__(self):
    return "Dog({})".format(self.name)
```

Python: ООП декораторы

```
class Rectangle:
    def __init__(self, width, height):
        self.width = width
        self.height = height
        self._allowed = False

    def calculate_area(self):
        return self.width * self.height

    @classmethod
    def new_square(cls, side_length):
        return cls(side_length, side_length)

    @staticmethod
    def square(a):
        return a**2
```

```
@property
def allowed(self):
    return self._allowed

@allowed.setter
def allowed(self, value):
    self._allowed = not(value)
```

```
sq = Rectangle.new_square(5)

print(sq.calculate_area())
# 25
sq.allowed=0
print(sq.allowed)
# True
print(Rectangle.square(4))
# 16
```

Python: Исключения

```
try:
   r = myfunc()
except Exception as e:
   print(e)
finally:
   print("end")


raise RuntimeError('Failed to open database')

assert x==0, 'error: x!=0'
```

Python: модули

```
import numpy as np

help(np)
np.__name__
np.__version__

from numpy.random import rand
```

numpy

matplotlib

sympy

pandas

geopandas

scikit-image

scikit-learn

Python: упражнения numpy

https://github.com/rougier/numpy-100

100 numpy exercises

This is a collection of exercises that have been collected in the numpy mailing list, on stack overflow and in the numpy documentation. The goal of this collection is to offer a quick reference for both old and new users but also to provide a set of exercises for those who teach.

If you find an error or think you've a better way to solve some of them, feel free to open an issue at https://github.com/rougier/numpy-100.

File automatically generated. See the documentation to update questions/answers/hints programmatically.

Run the `initialize.py` module, then for each question you can query the answer or an hint with `hint(n)` or `answer(n)` for n question number.

```
In [ ]: %run initialise.py
```

Python: упражнения pandas

https://github.com/ajcr/100-pandas-puzzles/

100 pandas puzzles

Inspired by 100 Numpy exerises, here are 100* short puzzles for testing your knowledge of pandas' power.

Since pandas is a large library with many different specialist features and functions, these excercises focus mainly on the fundamentals of manipulating data (indexing, grouping, aggregating, cleaning), making use of the core DataFrame and Series objects.

Many of the excerises here are stright-forward in that the solutions require no more than a few lines of code (in pandas or NumPy... don't go using pure Python or Cython!). Choosing the right methods and following best practices is the underlying goal.

The exercises are loosely divided in sections. Each section has a difficulty rating; these ratings are subjective, of course, but should be a seen as a rough guide as to how inventive the required solution is.

If you're just starting out with pandas and you are looking for some other resources, the official documentation is very extensive. In particular, some good places get a broader overview of pandas are...

- 10 minutes to pandas
- pandas basics
- tutorials
- cookbook and idioms

Python: что почитать?

Лутц М. Изучаем Python. пер. с англ. — СПб.: ООО “Диалектика”, 2019.

Дауни А. Основы Python. Научитесь думать как программист. пер. с англ. — Москва : "Манн, Иванов и Фербер", 2021.

Дейтел П., Дейтел Х. Python: Искусственный интеллект, большие данные и облачные вычисления. — СПб.: Питер, 2020.

Python Help : Tutorial https://docs.python.org/3/tutorial/index.html

SoloLearn : Python http://www.sololearn.com/Course/Python/

http://github.com/mechanoid5/ml_lectorium