



GeekBrains

Курс Основы Python



GeekBrains

Урок 2

Встроенные типы и операции с ними

На этом уроке

1. Встроенные типы данных в Python:
 - NoneType.
 - Числа.
 - Исключения.
 - Строки.
 - Байты.
 - Множества.
 - Списки.
 - Кортежи.
 - Словари.
2. О цикле for in для обхода последовательностей.
3. Понятие тернарного оператора.
4. Оператор is.
5. Десятка лучших трюков в Python.

Встроенные типы данных в Python

NoneType

Числа

Исключения

Строки

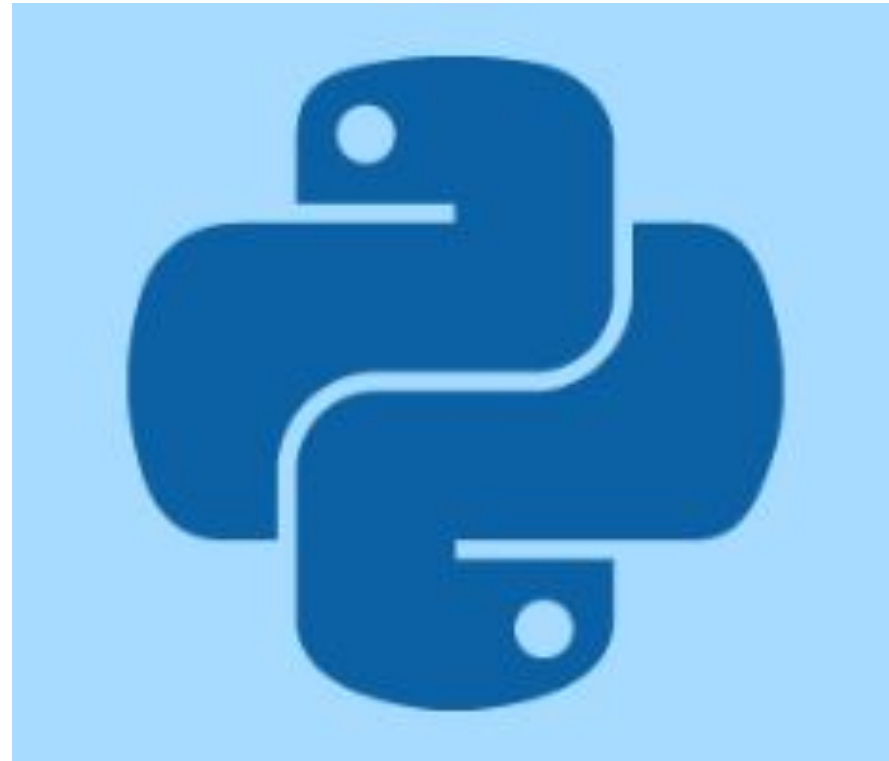
Байты

Множества

Списки

Кортежи

Словари



Тип данных: число

Перевод между системами счисления

Битовые операции

Операция	Пример
Взятие по модулю	<code>print(abs(-6)) -> 6</code>
Побитовое И	<code>print(4 & 6) -> 4</code>
Побитовое ИЛИ	<code>print(4 6) -> 6</code>
Побитовое исключающее ИЛИ	<code>print(4 ^ 6) -> 2</code>
Битовый сдвиг влево	<code>print(4 << 6) -> 256</code>
Битовый сдвиг вправо	<code>print(4 >> 6) -> 0</code>

Функция	Описание	Пример
<code>int()</code>	Преобразовать к целому числу в десятичном формате (по умолчанию). Допускается выбор другой системы счисления с помощью дополнительного параметра (от 2 до 36)	<code>print(int(17.5)) -> 17</code> <code>print(int('10001', 2)) -> 17</code>
<code>bin()</code>	Преобразовать к двоичному формату	<code>print(bin(17)) -> 0b10001</code>
<code>oct()</code>	Преобразовать к восьмеричному формату	<code>print(oct(17)) -> 0o21</code>
<code>hex()</code>	Преобразовать к шестнадцатеричному формату	<code>print(hex(17)) -> 0x11</code>

Тип данных: строка

- ❖ Конкатенация
- ❖ Взятие элемента по индексу
- ❖ Извлечение среза
- ❖ Обратная итерация
- ❖ Реверс на месте

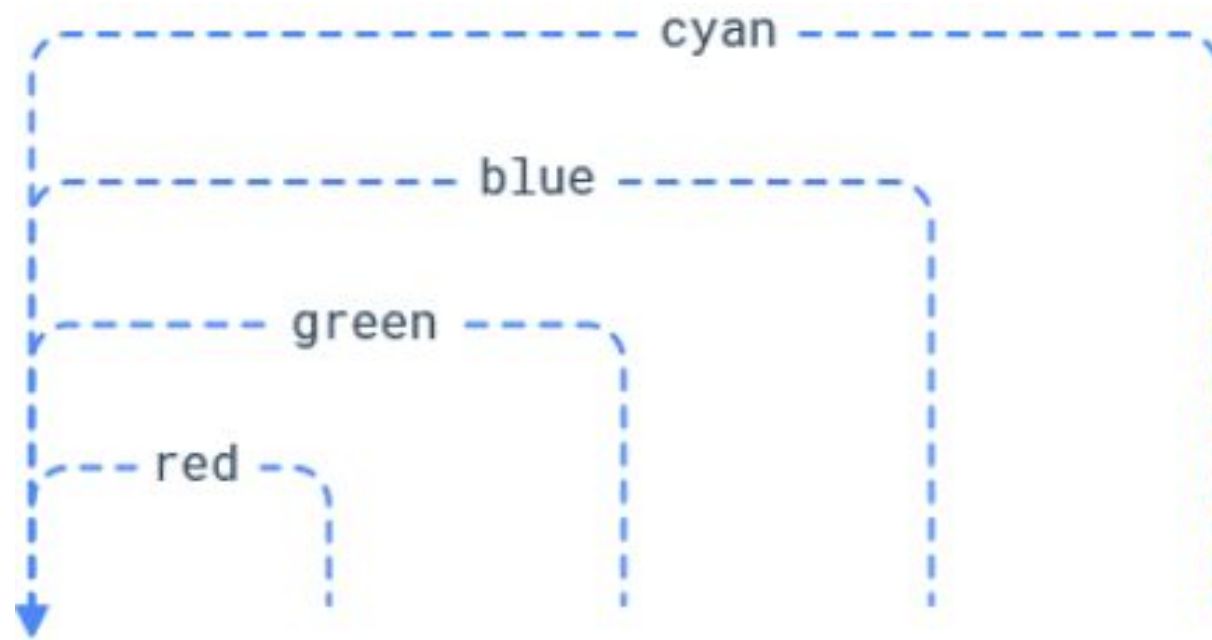


Тип данных: список

`.append(el)`
`.extend(my_list)`
`.insert(pos, el)`
`.remove(el)`
`.pop(pos)`
`.index(el)`
`.count(el)`
`.sort([key-функция])`
`.reverse()`
`.copy()`
`.clear()`

z =	[3,	7,	4,	2]
index	0	1	2	3

Тип данных: кортеж



`T = ('red', 'green', 'blue', 'cyan')`

Тип данных: множество



- .add(**el**)
- .remove(**el**)
- .discard(**el**)
- .pop()
- .copy()
- .clear()

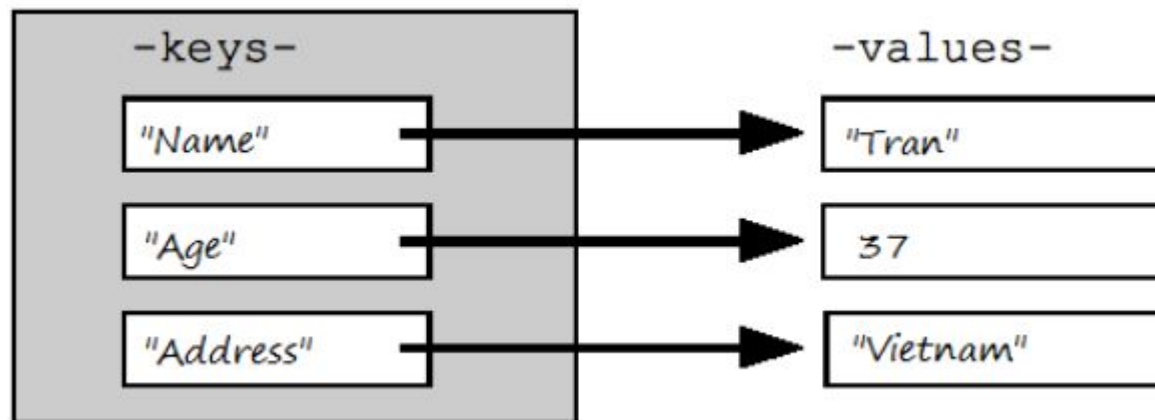
Python Множества

```
a = {1, 3, 0, 1, 3, 2}  
print(a)
```

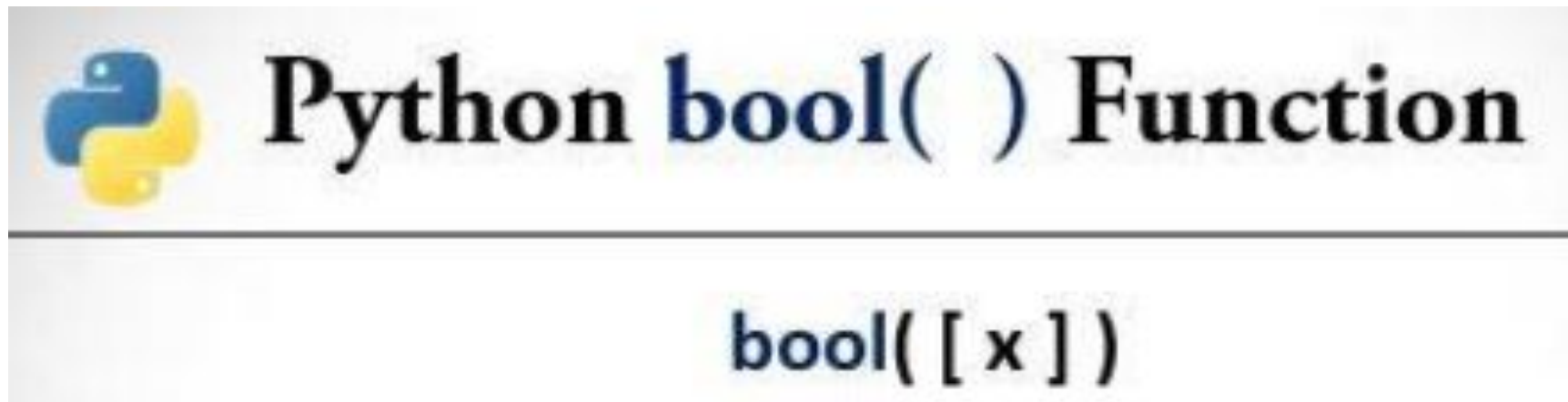
```
{0, 1, 2, 3}
```

Тип данных: словарь

`.keys()`
`.values()`
`.items()`
`.get(key)`
`.popitem()`
`.setdefault(key)`
`.pop(key)`
`.update(new_dict)`
`.copy()`
`.clear()`



Тип данных: bool



Тип данных: bytes и bytearray

BYTES

`bytes` — единица хранения информации (текстовой, графической, звуковой)

`bytearray` — массив байтов

Тип данных: NoneType



None

Один из вариантов сброса
переменной в пустое состояние

Тип данных: исключение

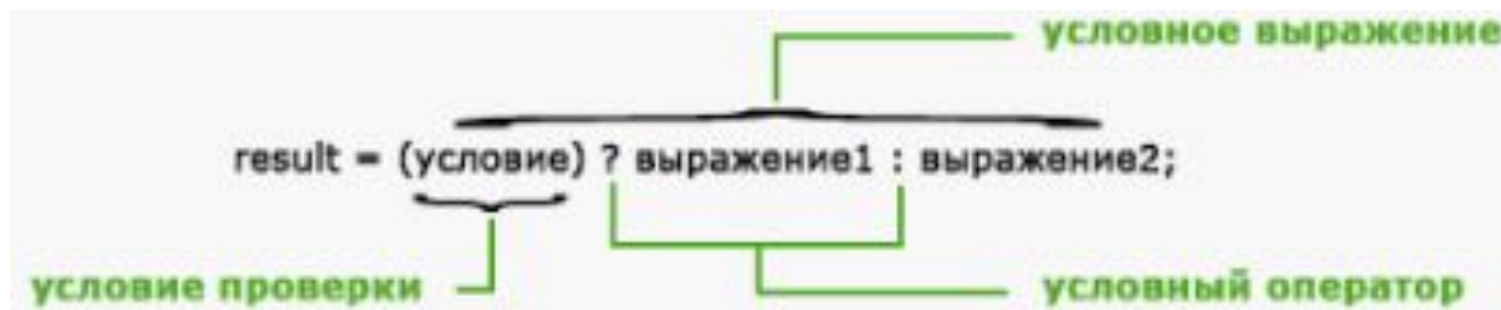
```
try:  
    print(10 / 0)  
except ZeroDivisionError:  
    print('Error')
```

Цикл for in для обхода последовательностей



```
for [переменная-итератор] in [последовательность]:  
    [действия, выполняемые для каждой переменной]
```


Понятие тернарного оператора



`(a - b) if a > b else 1 / 0`

Оператор is

Проверяет тождественность (идентичность) двух объектов в памяти. Возвращает значение True (истина), если переменные ссылаются на один и тот же объект.



Десятка лучших трюков в Python

1. Объединение списков без цикла.
2. Удаление дубликатов в списке.
3. Обмен значениями через кортежи.
4. Вывод значения несуществующего ключа в словаре.
5. Поиск самых часто встречающихся элементов списка.
6. Распаковка последовательностей при неизвестном количестве элементов.
7. Вывод с помощью функции `print()` без перевода строки.
8. Сортировка словаря по значениям.
9. Нумерованные списки.
10. Транспонирование матрицы.



ИТОГИ

В языке Python мы работаем с объектами, которые могут относиться к различным типам и, соответственно, поддерживать различные типы операций. В Python мы можем работать как с привычными типами данных, такими как числа, строки, байты, логический тип, так и со специализированными для Python типами: списками, кортежами, словарями, множествами. Объекты этих типов могут использоваться для хранения данных в процессе работы программ.