Python 3

Типы данных

Типы данных Python

- **Логический**, может принимать одно из двух значений — True (истина) или False (ложь).
- **Числа**, могут быть целыми (1 и 2), с плавающей точкой (1.1 и 1.2), дробными (1/2 и 2/3), и даже комплексными.
- **Строки** — последовательности символов Юникода, например, HTML-документ.
- **Байты** и **массивы байтов**, например, файл изображения в формате JPEG.
- **Списки** — упорядоченные последовательности значений.
- **Кортежи** — упорядоченные неизменяемые последовательности значений.
- **Множества** — неупорядоченные наборы значений.
- **Словари** — неупорядоченные наборы пар вида ключ-значение.

Логический

- Логический тип данных может принимать одно из двух значений: истина или ложь (True/False)
- Результатом вычисления выражений также может быть логическое значение.

```
if size < 0:
    raise ValueError('число должно быть неотрицательным')
```

- Из-за некоторых обстоятельств, связанных с наследием оставшимся от Python 2, логические значения могут трактоваться как числа. True как 1, и False как 0.

Числа

- Python поддерживает как целые числа, так и с плавающей точкой.
- Нет необходимости объявлять тип для их различия; Python определяет его по наличию или отсутствию десятичной точки.
- Можно использовать функцию type() для проверки типа любого значения или переменной.

```
>>> type(1000)
<class 'int'>
```

- Функцию isinstance(переменная, тип) тоже можно использовать для проверки принадлежности значения или переменной определенному типу.

Целые числа и числа с плавающей точкой

- float(целое) – преобразование в число с плавающей точкой
- int(дробь) – преобразоывание дроби в целое отбрасывая дробную часть
- Точность чисел с плавающей точкой равна 15 десятичным знакам в дробной части.
- Целые числа могут быть сколь угодно большими.
- Если ваше целое число больше чем 2**32 – 1, операции с ним будут медленными.

Основные операции с числами

- + Сложение двух чисел:
  print(6 + 2) # 8
- - Вычитание двух чисел:
  print(6 - 2) # 4
- * Умножение двух чисел:
  print(6 * 2) # 12
- / Деление двух чисел:
  print(6 / 2) # 3.0
- // Целочисленное деление двух чисел:
  print(7 / 2) # 3.5
  print(7 // 2) # 3
  Данная операция возвращает целочисленный результат деления, отбрасывая дробную часть
- ** Возведение в степень:
  print(6 ** 2) # Возводим число 6 в степень 2. Результат - 36
- % Получение остатка от деления:
  print(7 % 2) # Получение остатка от деления числа 7 на 2. Результат - 1

Дроби

```
>>> import fractions
>>> x = fractions.Fraction(1, 3)
>>> x
Fraction(1, 3)
>>> x * 2
Fraction(2, 3)
>>> fractions.Fraction(6, 4)
Fraction(3, 2)
>>> fractions.Fraction(0, 0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
  File "fractions.py", line 96, in __new__
    raise ZeroDivisionError('Fraction(%s, 0)' % numerator)
ZeroDivisionError: Fraction(0, 0)
```

Тригонометрия

- Тригонометрические, а также многие другие математические функции, находятся в модуле math

```
>>> import math
>>> math.pi
3.141592653589793
>>> math.sin(math.pi / 2)
1.0
>>> math.tan(math.pi / 4)
0.9999999999999989
```

Списки

- Существует четыре способа добавить элементы в список.
- >>> a_list = ['a']
  >>> a_list = a_list + [2.0, 3]
  >>> a_list
  ['a', 2.0, 3]
  >>> a_list.append(True)
  >>> a_list
  ['a', 2.0, 3, True]
  >>> a_list.extend(['four', 'Ω'])
  >>> a_list
  ['a', 2.0, 3, True, 'four', 'Ω']
  >>> a_list.insert(0, 'Ω')
  >>> a_list
  ['Ω', 'a', 2.0, 3, True, 'four', 'Ω']

Разрезание списков

- >>> a_list
  ['a', 'b', 'mpilgrim', 'z', 'example']
  >>> a_list[1:3]
  ['b', 'mpilgrim']
  >>> a_list[1:-1]
  ['b', 'mpilgrim', 'z']
  >>> a_list[0:3]
  ['a', 'b', 'mpilgrim']
  >>> a_list[:3]
  ['a', 'b', 'mpilgrim']
  >>> a_list[3:]
  ['z', 'example']
  >>> a_list[:]
  ['a', 'b', 'mpilgrim', 'z', 'example']

Поиск в списке

- >>> a_list = ['a', 'b', 'new', 'mpilgrim', 'new']
  >>> a_list.count('new')
  2
  >>> 'new' in a_list
  True
  >>> 'c' in a_list
  False
  >>> a_list.index('mpilgrim')
  3
  >>> a_list.index('new')
  2
  >>> a_list.index('c')
  Traceback (innermost last):
  File "<interactive input>", line 1, in ?
  ValueError: list.index(x): x not in list

Удаление элементов из списка

- >>> a_list = ['a', 'b', 'new', 'mpilgrim', 'new']
  >>> a_list[1]
  'b'
  >>> del a_list[1]
  >>> a_list
  ['a', 'new', 'mpilgrim', 'new']
  >>> a_list[1]
  'new'

Удаление элементов по значению

- >>> a_list.remove('new')
  >>> a_list
  ['a', 'mpilgrim', 'new']
  >>> a_list.remove('new')
  >>> a_list
  ['a', 'mpilgrim']
  >>> a_list.remove('new')
  Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
  ValueError: list.remove(x): x not in list

Кортежи (tuple)

- Кортеж — это неизменяемый список.

Множества

```
>>> a_set = {1}
>>> a_set
{1}
>>> type(a_set)
<class 'set'>
>>> a_set = {1, 2}
>>> a_set
{1, 2}
```

```
>>> a_list = ['a', 'b', 'mpilgrim', True, False, 42]
>>> a_set = set(a_list)
>>> a_set
{'a', False, 'b', True, 'mpilgrim', 42}
>>> a_list
['a', 'b', 'mpilgrim', True, False, 42]
```

Операции со множествами: добавление

```
>>> a_set = {1, 2}
>>> a_set.add(4)
>>> a_set
{1, 2, 4}
>>> len(a_set)
3
>>> a_set.add(1)
>>> a_set
{1, 2, 4}
>>> len(a_set)
3
```

```
>>> a_set = {1, 2, 3}
>>> a_set
{1, 2, 3}
>>> a_set.update({2, 4, 6})
>>> a_set
{1, 2, 3, 4, 6}
>>> a_set.update({3, 6, 9}, {1, 2, 3, 5, 8, 13})
>>> a_set
{1, 2, 3, 4, 5, 6, 8, 9, 13}
>>> a_set.update([10, 20, 30])
>>> a_set
{1, 2, 3, 4, 5, 6, 8, 9, 10, 13, 20, 30}
```

Удаление элементов множества

```
>>> a_set = {1, 3, 6, 10, 15, 21, 28, 36, 45}
>>> a_set
{1, 3, 36, 6, 10, 45, 15, 21, 28}
>>> a_set.discard(10)
>>> a_set
{1, 3, 36, 6, 45, 15, 21, 28}
>>> a_set.discard(10)
>>> a_set
{1, 3, 36, 6, 45, 15, 21, 28}
>>> a_set.remove(21)
>>> a_set
{1, 3, 36, 6, 45, 15, 28}
>>> a_set.remove(21)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 21
```

pop() и clear() для множеств

```
>>> a_set = {1, 3, 6, 10, 15, 21, 28, 36, 45}
>>> a_set.pop()
1
>>> a_set.pop()
3
>>> a_set.pop()
36
>>> a_set
{6, 10, 45, 15, 21, 28}
>>> a_set.clear()
>>> a_set
set()
>>> a_set.pop()
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
KeyError: 'pop from an empty set'
```

Операции со множествами

```
>>> a_set = {2, 4, 5, 9, 12, 21, 30, 51, 76, 127, 195}
>>> 30 in a_set
True
>>> 31 in a_set
False
>>> b_set = {1, 2, 3, 5, 6, 8, 9, 12, 15, 17, 18, 21}
>>> a_set.union(b_set)
{1, 2, 195, 4, 5, 6, 8, 12, 76, 15, 17, 18, 3, 21, 30, 51, 9,
127}
>>> a_set.intersection(b_set)
{9, 2, 12, 5, 21}
>>> a_set.difference(b_set)
{195, 4, 76, 51, 30, 127}
>>>
a_set.symmetric_difference(b_set)
{1, 3, 4, 6, 8, 76, 15, 17, 18, 195, 127, 30, 51}
```

```
>>> b_set.symmetric_difference(a_set)
{3, 1, 195, 4, 6, 8, 76, 15, 17, 18, 51, 30, 127}
>>> b_set.symmetric_difference(a_set) ==
a_set.symmetric_difference(b_set)
True
>>> b_set.union(a_set) == a_set.union(b_set)
True
>>> b_set.intersection(a_set) ==
a_set.intersection(b_set)
True
>>> b_set.difference(a_set) == a_set.difference(b_set)
False
```

Словари

- >>> a_dict = {'server': 'db.diveintopython3.org', 'database': 'mysql'}
  >>> a_dict
  {'server': 'db.diveintopython3.org', 'database': 'mysql'}
  >>> a_dict['server']
  'db.diveintopython3.org'
  >>> a_dict['database']
  'mysql'
  >>> a_dict['db.diveintopython3.org']
  Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
  KeyError: 'db.diveintopython3.org'

Изменение словарей

```
>>> a_dict
{'server': 'db.diveintopython3.org', 'database': 'mysql'}
>>> a_dict['database'] = 'blog'
>>> a_dict
{'server': 'db.diveintopython3.org', 'database': 'blog'}
>>> a_dict['user'] = 'mark'
>>> a_dict
{'server': 'db.diveintopython3.org', 'user': 'mark', 'database': 'blog'}
>>> a_dict['user'] = 'dora'
>>> a_dict
{'server': 'db.diveintopython3.org', 'user': 'dora', 'database': 'blog'}
>>> a_dict['User'] = 'mark'
>>> a_dict
{'User': 'mark', 'server': 'db.diveintopython3.org', 'user': 'dora', 'database':
'blog'}
```

None

- None — это специальная константа в Python. Она обозначает пустое значение. None — это не то же самое, что False. None также и не 0. None даже не пустая строка. Если сравнивать None с другими типами данных, то результатом всегда будет False.
- None — это просто пустое значение. None имеет свой собственный тип (NoneType). Вы можете присвоить None любой переменной, но вы не можете создать других объектов типа NoneType. Все переменные, значение которых None равны друг другу.

Лабораторная работа

- Создайте и наполните значениями список целых чисел, кортеж строк, множество чисел с плавающей точкой, и словарь, ключами которого служат числа, а значениями – строки.
- Выведите все эти объекты на экран.
- Выполните слияние списка и множества, выведите результат на экран.
- Выполните преобразование списка во множество, выведите результаты на экран.