

Python Basics v2.0

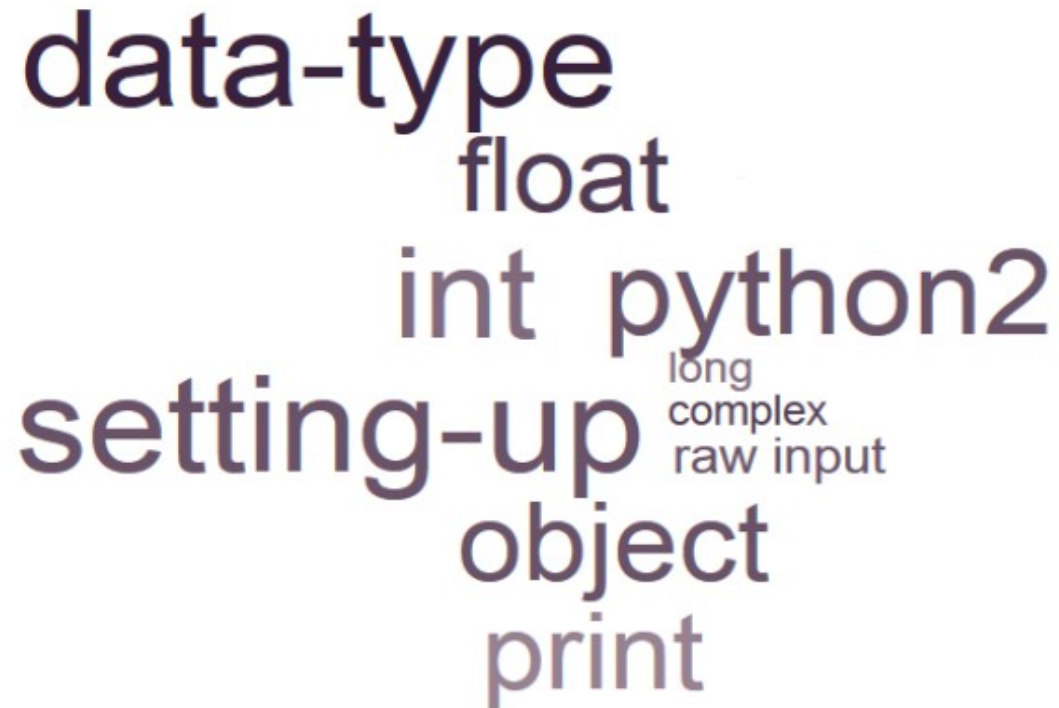
модуль I

Числа и строки: Простейшие программы

Сергей Колесник

Минск, WGU, 2017

Что было в прошлый раз?



A word cloud of Python-related terms. The words are arranged in a roughly circular pattern, with 'data-type' at the top, 'setting-up' on the left, 'object' and 'print' at the bottom, and 'python2' on the right. Other words like 'float', 'int', 'long', 'complex', and 'raw input' are interspersed between the larger words.

data-type
float
int python2
setting-up long complex raw input
object
print

Дополнения

- Последовательное присваивание

- `x = y = <выражение>`

- Множественное присваивание

- `first, second = <выражение>, <выражение>`

Практика

Вычисление корней квадратного уравнения. Коэффициенты такие, что есть два разных корня уравнения.

$$ax^2 + bx + c = 0$$

Input: вещественные коэффициенты a , b , c

Output: Два вещественных корня квадратного уравнения

Элементарные типы данных

Числа

Строки

Логический тип

Пустой тип

Списки

Кортежи

Словари

Файлы

Множества

Строки

- Строка — набор символов (коллекция)
- Символ — строка единичной длины

Создание строки

`'hello, world'`

Создание строки

'hello, world'

"hello, world"

Создание строки

`'hello, world'`

`"hello, world"`

`"""hello,
world"""`

Создание строки

`'hello, world'`

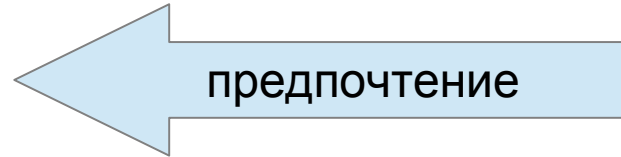
`"hello, world"`

`"""hello,
world"""`

`"""hello,
world"""`

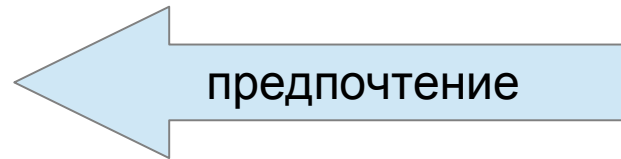
Создание строки

`'hello, world'`



`"hello, world"`

`"""hello,
world"""`



`"""hello,
world"""`

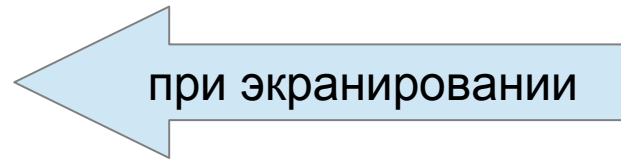
Создание строки

`'hello, world'`

`"hello, world"`

`"""hello,
world"""`

`"""hello,
world"""`



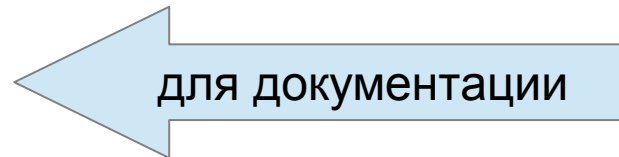
Создание строки

```
'hello, world'
```

```
"hello, world"
```

```
"""hello,  
    world"""
```

```
""" "hello,  
    world" " " "
```



Вывод специальных символов

- Экранирование символов:
"\' \" ' → строка из одинарной и двойной кавычки (длины 2)

Вывод специальных символов

- Экранирование символов:
 '\' \' ' → строка из одинарной и двойной кавычки (длины 2)
- Специальные символы:
 'hello,\nworld' → строка из двух линий (\n – символ перевода строки)

Выражения со строками

- Строка – объект в python
 - `type('hello') → str`
- `''` → пустая строка
- Строка может участвовать в выражениях
 - `'hello,' + ' world' → 'hello, world'`
 - `'x' * 3 → 'xxx'`

Операции со строками

- Получение символа по индексу

`'hello'[1] → 'e'`

Символы строк нумеруются с нуля

Операции со строками

- Получение символа по индексу

`'hello'[1] → 'e'`

- Получение среза (подстроки)

`'hello'[1:3] → 'el'`

`'hello'[:4:2] → 'hl'`

`'hello'[:-1] → 'hell'`

Операции со строками

- Получение символа по индексу

`'hello'[1] → 'e'`

- Получение среза (подстроки)

`'hello'[1:3] → 'el'`

`'hello'[:4:2] → 'hl'`

`'hello'[:-1] → 'hell'`

- Реверс строки

`'hello'[::-1] → 'olleh'`

Практика

Вывод цифр трехзначного числа через пробел

Input: Трехзначное целое число

Output: Три цифры, записанные через пробел

Restriction: Использовать строковые операции

Методы объектов

- Метод объекта – функция, соотнесенная с этим объектом
- Вызов метода объекта подобен вызову функции, но предваряется самим объектом, с последующей точкой

`'hello'.upper() → 'HELLO'`

Проверка строки

- `'lo' in 'hello' → True`

Проверка строки

- `'lo' in 'hello' → True`

- **Метод `.find`**

- `'hello'.find('lo') → 3`

- `'hello'.find('python') → -1`

Проверка строки

- `'lo' in 'hello' → True`
- **Метод `.find`**
 - `'hello'.find('lo') → 3`
 - `'hello'.find('python') → -1`
- **Методы `.startswith` и `.endswith`**
 - `'hey'.startswith('he') → True`
 - `'hey'.endswith('he') → False`

Практика

Программы, отвечающие на вопросы:

- С какого символа во введенной строке начинается сочетание букв 'wg'? [.find]
- Начинается и кончается ли строка на сочетание букв 'wg'? [.startswith, .endswith]

Проверка строки

- Методы `.islower`, `.isupper`, `.istitle`

`'hello'.islower()` → `True`

`'HELLO'.isupper()` → `True`

`'Hello'.istitle()` → `True`

Проверка строки

- Методы `.islower`, `.isupper`, `.istitle`

`'hello'.islower()` → `True`

`'HELLO'.isupper()` → `True`

`'Hello'.istitle()` → `True`

- Методы `.isalnum`, `.isalpha`, `.isspace`, `.isdigit`

`'hey'.isalpha()` → `True`

`'123'.isdigit()` → `True`

`'hey123'.isalnum()` → `True`

`' \n\t'.isspace()` → `True`

Преобразование строки

- Методы `.lower`, `.upper`, `.title`

`'HELlo'.lower()` → `'hello'`

`'HELlo'.upper()` → `'HELLO'`

`'HELlo'.title()` → `'Hello'`

- Методы `.replace`, `.strip`

`'hello'.replace('hell', 'heaven')` →
`'heaveno'`

`' \n\thello\t\n '`.strip() → `'hello'`

Практика

Программы, отвечающие на вопросы:

- Как бы выглядела строка со всеми большими буквами? `[.upper]`
- Написано ли слово с большой буквы? `[.istitle]`
- Если заменить 'wg' на звездочки, то как бы выглядело предложение? `[.replace]`

Оставшиеся методы строки

- Метод `.split`

`'x,y,z'.split(',') → ['x', 'y', 'z']`

`'A B'.split() → ['A', 'B']`

`'A B'.split(' ') → ['A', ' ', 'B']`

- Метод `.join`: обратная операция

`';'.join(['x', 'y', 'z']) → 'x;y;z'`

Практика

Запрос имени и фамилии пользователя.
Затем вывод приветствия, с упоминанием
только имени, но не фамилии.

Input: Два слова, записанных через пробел
на одной строке

Output: Приветствие с именем

Hint: метод строки `.split()`

Практика

Состоит ли первое слово ввода только из цифр, а второе только из букв?

Input: Два слова, записанных через пробел на одной строке

Output: Два раза True или False

Hint: методы строки `.split`, `.isdigit`, `.isalpha`

Функции для работы со строками

- `len` — возвращает длину строки
 - `len('hello') → 5`
- У каждого символа есть код (номер) в кодовой таблице символов (в кодировке)
`ord` — возвращает код символа в текущей кодировке
 - `ord('A') → 65`
- `chr` — возвращает символ по коду из текущей кодировки
 - `chr(65) → 'A'`

Практика

Преобразование трехзначной строки в число без использования преобразования типа

Input: Целое трехзначное число

Output: Целое число

Форматированный вывод

- Метод строки `.format`
- `{ }` - позиционный place holder
- `{ name }` — именованный place holder

Форматированный вывод

- Можно не беспокоиться про типы передаваемых значений

```
'first {} second {}'.format('str', 13)
```

- Можно использовать одно и тоже значение несколько раз

```
'first {value} first again {value}'.format(value='value')
```

- Позволяет сосредоточиться на формате вывода

Устаревший вариант

- **Позиционная передача**

```
'first %s second %s' % ('str', 13)
```

- **Именованная передача**

```
'first %(value)s first %(again)s' % dict(value='value')
```

Спасибо за внимание!