Python Basics v2.0

модуль I

Числа и строки:
Простейшие программы

Сергей Колесник

Минск, WGU, 2016

Кто Я?

- Сергей Колесник
- Web-разработчик, python
- МАИ, кафедра вычислительной математики и программирования
- WarGaming, Яндекс
- sergey.s.kolesnik@wargaming.net
- skype: sergey.s.kolesnik

Кто Вы?

- Навыки программирования
- Чем занимаетесь на работе?
- Проекты, языки, интересы

Принцип обучения

- Вы, не я

Принцип обучения

- Вы, не я
- От меня:
  - Структурность информации
  - Ответы на вопросы
  - Помощь в непонятных ситуациях
  - Личный опыт

Что мешает обучению?

- Скромность
- Отсутствие цели
- Отсутствие своевременной практики

Принцип обучения

- Вы, не я
- От Вас:
  - Освоение материала
  - При любых затруднениях, непонятках: заострить мое внимание на этом
  - Практика, не откладывая в долгий ящик
  - Мотивация

Цель?

Цель

- Обучение мышлению
  - Систематическому
  - Абстрактному
  - Аналитическому
- Прикладное направление
  - Использование для работы

Формат занятий

- ???

Формат

- Теория, практика
- По окончанию блока:
  - Самостоятельная проработка
  - Кейсы (тестирование)

Формат кейсов

- Набор неочевидных ситуаций
- Цель: познакомиться с тонкостями языка и проверить свои знания

Формат самостоятельной работы

- Набор задач разной сложности
- Дедлайнов нет
- Можно присылать решения, если есть вопросы

Материалы тренинга

- Слайды, задачи с занятия, задачи для самостоятельной работы, кейсы, примеры кода
- Google-диск

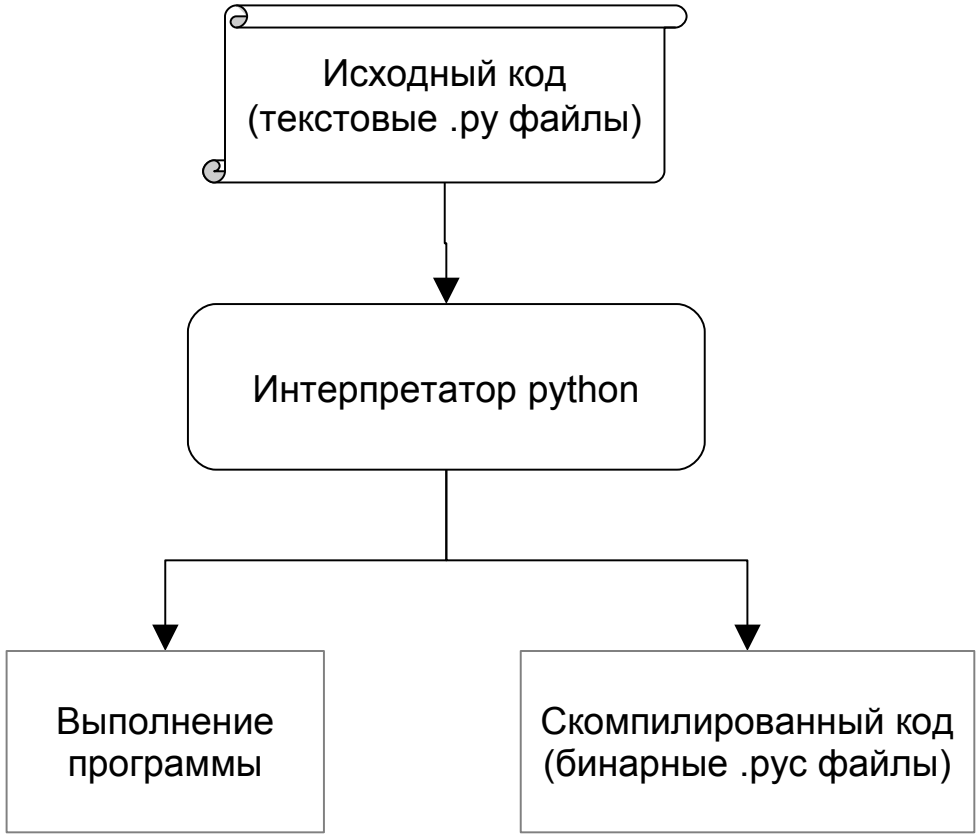
Почему Python?

Почему Python?

- Легкий для изучения
- Современный (большое сообщество)
- Поддержка нескольких концепций
- Язык общего назначения (не предметно-ориентированный)
- Практичный

Что из себя представляет python программа?

Общение с машиной



Составляющие программы

Программа
- Пакеты
  - Модули
    - Функции
      - Инструкции и выражения
        - Объекты и операторы

Составляющие программы

Программа
- Пакеты
  - Модули
    - Функции
      - Инструкции и выражения
        - **Объекты и операторы**

Объекты

- Строительная единица представления данных
- Делятся на группы по общим свойствам
- Будем называть группу типом данных (или просто типом)

Элементарные типы данных

Числа
Строки
Логический тип
Пустой тип
Списки
Кортежи
Словари
Файлы
Множества

Задание

- Разворачивание окружения
- Cygwin, консоль
- python, pip, ipython

Элементарные типы данных

Числа
Строки
Логический тип
Пустой тип
Списки
Кортежи
Словари
Файлы
Множества

Числа

- `int`
- `long`
- `float`
- `complex`

Числа

- `int: 1  -5  0  2147483647`
- `long`
- `float`
- `complex`

Числа

- `int: 1  -5  0  2147483647`
- `long:` все, что не влезло в `int`
- `float`
- `complex`

Числа

- `int: 1  -5  0  2147483647`
- `long:` все, что не влезло в `int`
- `float: 1.0  -4.7  0.3333  9.5e-07`
- `complex`

Числа

- `int: 1  -5  0  2147483647`
- `long:` все, что не влезло в `int`
- `float: 1.0  -4.7  0.3333  9.5e-07`
- `complex: 1+2j  1j`

32bit vs 64bit

- 32bit
  - размер типа int 4 байта
  - границы: `-2 ** 31 <= int <= 2 ** 31 – 1`

    ```
    -2 ** 31     = -2147483648
     2 ** 31 – 1 =  2147483647
    ```
  - long: `< -2 ** 31, >= 2 ** 31`

32bit vs 64bit

- 64bit
  - размер типа int 8 байтов
  - границы: `-2 ** 63 <= int <= 2 ** 63 – 1`

    ```
    -2 ** 63     = -9223372036854775808
     2 ** 63 – 1 =  9223372036854775807
    ```
  - long: `< -2 ** 63, >= 2 ** 63`

Альтернативные представления целых чисел

- двоичное: `0b111`
- восьмеричное: `042`
- шестнадцатеричное: `0x1A`

Вопрос

Чем отличаются int и long?

Вопрос

Чем отличаются `int` и `long`?

- Разная скорость операций: int поддерживается на уровне процессора (т.е. быстрее)
- Разная размерность: long потенциально бесконечный

Операции с числами

- `1 + 5`
- `78 - 455`
- `32 * 32`
- `2 ** 10000`
- `10 / 3,` целочисленное деление
- `10.0 / 3,` вещественное деление (обычное)
- `10 % 3,` остаток от деления

Сравнение чисел

- `1 < 3` → `True`
- `1 > 3` → `False`
- `1 >= 1` → `True`
- `1 <= 3` → `True`
- `1 == 5` → `False`
- `1 != 5` → `True`

Сравнение вещественных чисел

```
10.0 / 3 - 3 == 1 / 3.0
```

Сравнение вещественных чисел

Аккуратно!

Неправильно:

```
10.0 / 3 - 3 == 1 / 3.0
```

Правильно:

```
abs((10.0 / 3 - 3) - (1 / 3.0)) < 1e-6
```

Порядок операций

- `1 + 2 * 3`

Порядок операций

- `1 + 2 * 3`
- `1 + 2 * 3`

Порядок операций

- `1 + 2 * 3`
- `1 + 2 * 3`
- `(1 + 2) * 3`

Порядок операций

- `1 + 2 * 3`
- `1 + 2 * 3`
- `(1 + 2) * 3`

- `2 ** 3 ** 2`

Порядок операций

- `1 + 2 * 3`
- `1 + 2 * 3`
- `(1 + 2) * 3`

- `(2 ** 3) ** 2`

Если нет уверенности в порядке – используй скобки

Переменные

Переменные

- Переменная — это имя, ссылающееся на некоторое значение (на объект)
- Не существует, пока не начнет ссылаться на какое-либо значение
- Чтобы создать переменную, надо «назначить» ей значение

Создание переменной

- `x = 0`

Создаем переменную с именем «икс», ссылающуюся на значение 0 (объект типа int)

- Замечание: это не знак «равно», а инструкция присваивания.

- Можно читать «икс присвоить ноль»

Использование значения переменной

Переменная может участвовать в любых выражениях, в которых ее значение уместно

```
print x
```

Выведет 0

Использование значения переменной

Переменная может участвовать в любых выражениях, в которых ее значение уместно

```
print x
```

Выведет 0

```
13 / x
```

Нельзя делить на ноль

Изменение значения переменной

- Переменная может ссылаться только на одно значение в момент времени
- При повторном использовании инструкции присваивания, значение переменной будет «перетерто» на новое

```
x = 17
print x
```

Выведет 17

Количество переменных

- Одновременно может существовать несколько переменных
- Теоретическое количество не ограниченно
- Практическое количество ограниченно оперативной памятью

```
y = 1
x = y + 1
y = x ** 2
```

x ссылается на 2, y ссылается на 4

Специальные инструкции присваивания

- `x = 1`
- `x = x + 1`

Корректная инструкция. Увеличивает значение переменной икс на единицу

- `x += 1`

Аналог инструкции `x = x + 1`. Работает чуть быстрее.

Аналогично: `-=, *=, /=, %=`

Правила именования переменной

- Английские буквы верхнего и нижнего регистра
- Цифры
- Подчеркнутый пробел: «_»
- Имя не может начинаться с цифры

- Имя, передающее смысл переменной — залог читаемого кода

Практика

Вычисление площади треугольника по формуле:

$$S_{triangle} = \frac{1}{2} ha$$

Python

Интерактивный режим

Запуск написанного скрипта

Python

Интерактивный режим

Запуск написанного скрипта

В чем разница?

Python

Интерактивный режим

Запуск написанного скрипта

В чем разница?

- Необходимость `print`'а
- Специальная переменная _

Навигация в командной строке Unix

- Просмотр содержимого папки:
  - `ls`
- Перемещение по файловой системе:
  - `cd path/to/folder`
- В каком я каталоге?
  - `pwd`

Навигация в командной строке Unix

- Создать папку:
  - `mkdir new-folder`
- Удалить пустую папку:
  - `rmdir new-folder`

Навигация в командной строке Unix

- Создание пустого файла:
  - `touch new-file`
- Удаление файла (безвозвратно!):
  - `rm new-file`
- Удаление не пустой папки:
  - `rm -r dir-with-files`

Навигация в командной строке Unix

- Создание файла с содержимым:
  - `cat > new-file`
  - по окончанию ввода: `ctrl+d`
- Добавление содержимого в файл:
  - `cat >> existed-file`
- Показать содержимое файла:
  - `cat file`

Навигация в командной строке Unix

- Переименование файла == перемещение:
  - `mv dir1/existed-file dir2/new-name`
- Копирование файла:
  - `cp dir1/file dir2/`
- Копирование каталога:
  - `cp -r dir1 dir2/`

Преобразование типов

- `256 * 1.0` → `256.0`
- `float(256)` → `256.0`
- `long(256)` → `256L`
- `complex(256)` → `256+0j`

Встроенные функции

- `dir`

```
>>> dir(256+1j)
['conjugate','imag','real']
```

Встроенные функции

- dir

```
>>> dir(256+1j)
['conjugate','imag','real']
```

- help

```
>>> help(int)
```

Встроенные функции для чисел

- `abs(-5)` → `5`
- `max(1, 4)` → `4`
- `min(1, 4)` → `1`
- `sum([1, 4])` → `5`

Встроенные функции для чисел

- `abs(-5)` → `5`
- `max(1, 4)` → `4`
- `min(1, 4)` → `1`
- `sum([1, 4])` → `5`

- `bin(17)` → `'0b1111'`
- `oct(17)` → `'017'`
- `hex(17)` → `'0xF'`

Проверка типов объектов

- `type(1)` → `int`
- `type(1L)` → `long`
- `type(1.0)` → `float`
- `type(1+1j)` → `complex`

Проверка типов объектов

- `type(1)` → `int`
- `type(1L)` → `long`
- `type(1.0)` → `float`
- `type(1+1j)` → `complex`

- `type(int)` → `type`
- `type(type)` → `type`

Инструкция вывода

- До этого были только выражения
- Пример инструкции ***вывода информации***

```
print 1 + 2
```

- Выводит на экран результат вычисления выражения 1 + 2

Инструкции vs Выражения

Выражения

- Состоят из объектов и операторов, примененных к этим объектам
- Вычисляются интерпретатором
- Имеют результат вычисления

Инструкции

- Выполняются интерпретатором, а не вычисляются
- Т.о. не имеют результата вычисления
- Имеют результат выполнения (что-то происходит)
- Для записи могут использоваться выражения

Ввод

- Простейший ввод данных в программу – это ввод с клавиатуры
- Ввод осуществляется с помощью функции `raw_input`
- При этом возвращается строка, введенная пользователем

```
x = raw_input()
```

- В иксе окажется строка, которую ввели с клавиатуры
- Т.о. ввод всегда является строкой, даже если ввели число

Структура простой программы

Ввод данных

Обработка

Вывод

Практика

Программа, которая просит ввести имя пользователя, а затем выводит приветствие, с упоминанием этого имени

Input: Строка: имя пользователя
Output: Строка в виде: Hello, <name>!

Практика

Добавление ввода и вывода программе с треугольником: ввод высоты и основания, вывод конечного результата

Input: Два вещественных числа: высота и основание треугольника
Output: Строка в виде: S = <число>

Практика

Напишем программу, вычисляющую выражение

$$-7{,}1x^3 - 5{,}23x^2 + 0{,}4$$

Input: Вещественное число x
Output: Вещественное число: результат вычисления

Практика

Окргуление числа в большую сторону

Input: Вещественное число
Output: Целое число, полученное после округления

Практика

Преобразование дней, часов, минут и секунд в секунды

Input: Четыре целых числа: кол-во дней, часов, минут и секунд
Output: Целое число: общее кол-во секунд

Практика

Вывод цифр трехзначного числа через пробел

Input: Трехзначное целое число
Output: Три цифры, записанные через пробел

Практика

Преобразование секунд в дни, часы, минуты и секунды

Input: Целое число: кол-во секунд
Output: Четыре целых числа: кол-во дней, часов, минут и секунд

Резюме

- Работа с интерпретатором
- Базовые числовые типы данных
- Средства ввода-вывода и структура простой программы

Спасибо за внимание