Python.

Ловушки языка. 2to3.

Отступы.

- Отступы имеют значение =)

Присваивания.

- Создание объекта
- Связывание имени

- Python не копирует не явно

Присваивания.

- Не изменяемые объекты

```
a = b = 3
a = 4
print a, b  # 4, 3
```

- Изменяемые объекты

```
a = [1, 2, 3]
b = a
a.append(4)
print b
# b is now [1, 2, 3, 4] as well
```

Оператор +=

```
a = 1
a = a + 42
# a is 43
a = 1
a += 42
# a is 43
```

```
>>> z = [1, 2, 3]
>>> id(z)
24213240
>>> z += [4]
>>> id(z)
24213240
>>> z = z + [5]
>>> id(z)
24226184
```

Оператор +=

```
>>> t = ([],)
>>> t[0] += [2, 3]
Traceback (most recent call last):
    File "<input>", line 1, in ?
TypeError: object doesn't support item
assignment
>>> t
([2, 3],)
```

Оператор +=

- Меняет объект in-place

Атрибуты классов и объектов

```
>>> class Foo:
...     bar = []
...     def __init__(self, x):
...         self.bar.append(x)
...
>>> f = Foo(42)
>>> g = Foo(100)
>>> f.bar, g.bar
([42, 100], [42, 100])
```

Атрибуты классов и объектов

```
class Foo:
  a = 42
  def __init__(self):
    self.a = 43

f = Foo()
f.a
#43
```

```
class Foo:
  a = 42

f = Foo()
f.a
#42
```

Аргументы по-умолчанию

```
>>> def append_666(x=[]):
...     x.append(666)
...     print x
...
>>> append_666 ([1, 2, 3])
[1, 2, 3, 666]
>>> x = [1, 2]
>>> append_666 (x)
[1, 2, 666]
>>> x
[1, 2, 666]
```

Аргументы по-умолчанию

```
>>> append_666()
[666]
>>> append_666()
[666, 666]
>>> append_666()
[666, 666, 666]
```

Решение.

```
>>> def report(when=None):
...     if when is None:
...         when = time.time()
...     print when
...
>>> report()
1210294762.29
>>> time.sleep(5)
>>> report()
1210294772.23
```

UnboundLocalError

```
>>> def p():
...     x = x + 2
...
>>> p()
Traceback (most recent call last):
    File "<input>", line 1, in ?
    File "<input>", line 2, in p
UnboundLocalError: local variable 'x'
referenced before assignment
```

UnboundLocalError

```
>>> x = 2
>>> def q():
...     print x
...     x = 3
...     print x
...
>>> q()
Traceback (most recent call last):
    File "<input>", line 1, in ?
    File "<input>", line 2, in q
UnboundLocalError: local variable 'x' referenced
before assignment
```

"If a name is bound in a block, it is a local variable of that block. If a name is bound at the module level, it is a global variable. (The variables of the module code block are local and global.) If a variable is used in a code block but not defined there, it is a free variable.

When a name is not found at all, a NameError exception is raised. If the name refers to a local variable that has not been bound, a UnboundLocalError exception is raised."

- Переменные в функции могут быть либо глобальные, либо локальные.

  Но не одновременно.

Округление float

```
>>> c = 0.1
>>> c
0.10000000000000001
>>> repr(c)
'0.10000000000000001'
>>> str(c)
'0.1'
```

Округление float

```
>>> sum = 0.0
>>> for i in range(10):
...     sum += 0.1
...
>>> sum
0.9999999999999989
```

“There are no easy answers.” (c)

String concatenation

Pascal:

```
var S : String;
for I := 1 to 10000 do begin
    S := S + Something(I);
end;
```

Но в Python строки не изменяемы => на каждую операцию создается новая строка.

Пример.

```
>>> def f():
...     s = ""
...     for i in range(100000):
...         s = s + "abcdefg"[i % 7]
...
>>> timeit(f, number=10000)
23.7819999456
```

Пример.

```
>>> def g():
...     z = []
...     for i in range(100000):
...         z.append("abcdefg"[i % 7])
...     return ''.join(z)
...
>>> timeit(g, number=10000)
0.343000054359
```

- На самом деле это было немного вранье.
- Поведение пофикшено уже в Python 2.5

- Но! Это не является стандартом. В Jython и IronPython сохранено поведение.

Binary mode

```
f1 = open(filename, "r")  # text
f2 = open(filename, "rb") # binary
```

- Переводы строк.
- В Unix – файлы всегда в binary mode.

- В Windows text mode (\r\n) и binary mode (\n)

Исключения.

```
try:
    ...something that raises an error...
except IndexError, ValueError:
# expects to catch IndexError and ValueError
# wrong!

try:
  1/0
except ZeroDivisionError, e:
  print e

integer division or modulo by zero
```

Решение.

```
try:
    ...something that raises an error...
except (IndexError, ValueError):
# does catch IndexError and ValueError
```

Создание списков [[1] * 2] * 3

```
>>> x = [[1]*2]*3
>>> x
[[1, 1],
 [1, 1],
 [1, 1]]

>>> id(x[0]) == id(x[1]) == id(x[2])
True
```

```
>>> x[1][0] = 2
>>> x
[[2, 1],
 [2, 1],
 [2, 1]]
```

Решение.

```
>>> x = [[1]*2 for _ in range(3)]
>>> x
[1, 1], [1, 1], [1, 1]

>>> x[1][0] = 2
>>> x
[1, 1], [2, 1], [1, 1]
```

Разрешение имен.

Пример

Целочисленное деление

```
>>> 5/2
2

>>> 5*1.0/2
2.5
```

List slicing

```
>>> x = [10, 20, 30, 40, 50]
>>> x[2]
30
>>> x[2:]
[30, 40, 50]

>>> x[7]
Traceback (most recent call last):
    File "<stdin>", line 1, in <module>
IndexError: list index out of range

>>> x[7:]
[]
```

Code like pythonista.

http://python.net/~goodger/projects/pycon/2007/idiomatic/handout.html

Отличия Python 2.x от Python 3.x

Print

- В 2.7 print это не функция, а оператор.

```
# 2
print a, b
print >> file_name, a, b

# 3
print (a, b)
print (a, b, file=file_name)
```

Int vs. long

- В python 2.7 есть 2 типа int и long.

- Long в python 2.7 эквивалентен int в 3.3

Целочисленное деление.

```
# 2
>>> 5/2
2

# 3
>>> 5/2
2.5
```

Переименованные функции.

- Input(prompt) в python 3.x эквивалентен raw_input в python 2.x
- Range в python 3.x эквивалентен xrange в python 2.x

Сравнение с учетом типов

```
1 < ''
0 > None
```

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unorderable types: int() < str()
```

Строки

- Все строки являются Unicode строками

- Упразднен модификатор u””

```
# 3
>>>"I love {0}, {1}, and {2}".format("eggs",
"bacon", "sausage")
'I love eggs, bacon, and sausage'
```

Словари

- удалены методы словаря dict.iterkeys(), dict.itervalues() и dict.iteritems()

- переработаны методы .keys(), .values() и .items(), которые вместо списка ключей или значений возвращают легковесные объекты-контейнеры

- Многое другое

http://docs.python.org/3.0/whatsnew/3.0.html

- from __future__ import
- Python 2to3

- Почему еще не все используют Python 3.x?

Интерпретаторы

- Cpython – написан на Си под руководством Гвидо.
- Jython – написан на Java. Программы, выполняющиеся в среде Jython могут одновременно использовать классы языков Java и Python (версия 2.5)
- IronPython - написан на C#. В IronPython можно использовать типы .NET (версия 2.7)
- PyPy — это интерпретатор языка программирования Python, который написан на Python и может компилировать сам себя. (версия 1.9)

Проектики

- Пары слов

Вход: текст на русском языке и файл, содержащий список "не слов".

Программа должна построить список наиболее частых пар слов, употребляемых в одной фразе (предложении), при этом слова могут быть расположены в любом порядке в предложении. Из рассматриваемых слов необходимо исключить все слова, указанные в списке "не слов".

- Взлом шифра

Вход: текст, зашифрованный произвольным подстановочным шифром.

Выход: оригинальное сообщение

- Лесенка

Вход: исходное слово (например МУХА), целевое слово (например СЛОН) и словарь.
Выход: цепочка однобуквенных преобразований, позволяющая получить из исходного слова целевое, при этом каждый промежуточный шаг должен также являться словом

- Многочлены

Вход: 2 многочлена от нескольких переменных, заданных в математической форме (т.е. знаки некоторых операций могут отсутствовать).

Выход: Указание ошибки в записи, если таковая присутствовала, иначе сообщение о совпадении или несовпадении введенных многочленов.

- Кроссворд

Вход: список слов и геометрия (описание внешнего вида кроссворда).

Выход: заполненный кроссворд (сообщение об ошибке, если заполнить невозможно).

- Игра

Написать программу, позволяющую играть с компьютером в одну из следующих игр:

- Шахматы
- Покер

- Лабиринт

Вход: файл с описанием лабиринта, начальная позиция, позиция выхода из лабиринта и число бомб.

Выход: кратчайший путь от входа до выхода.
Примечание: бомба способна уничтожить одну стенку

- Жизнь

Написать программу, позволяющую задать начальную позицию, прокрутить время вперед, проверить на отсутствие предыдущей позиции

- Раскраска карты

Вход: описание стран на карте множеством отрезков (страна - многоугольник).

Выход: минимальное число цветов, необходимое для раскраски данной карты (т.е. 2 соседних страны не могут быть одного цвета), и соответствующая раскраска.

- Детектор частей речи в предложении

Разработать библиотеку, однозначно определяющую части речи и формы русских слов в предложении на основе вариантов, предложенных морфологическим анализатором (например, pymorphy) и вероятностной информации, извлеченной из уже размеченного текста (например, http://opencorpora.org/)

- Разбор графических файлов

Вход: файл(ы) в одном из следующих форматов:

- GIF (в т.ч. с анимацией)

- JPEG

- PNG

Выход: файл(ы) в формате BMP

- Чат

Децентрализованная программа-чат для произвольного числа пользователей.

- Сетевые сервисы

Написать одну из следующих программ:

- клиент для получения почты (по протоколу POP3)
- клиент для отправки почты (по протоколу SMTP)
- FTP-клиент
- BitTorrent-клиент

- Twitter-анализатор

Вход: слово (или хэштег), для которого определяется эмоциональная окраска твитов