PYTHON 基础

金　林
中南财经政法大学统计系
jinlin82@qq.com

2020 年 1 月 6 日

Python Logo

Designer

Guido van Rossum, the creator of Python

Facts

1. Designed by Guido van Rossum,
2. First appeared 20 February 1991; 25 years ago
3. Stable release：
    1. 3.5.2 / 27 June 2016; 5 months ago
    2. 2.7.12 / 28 June 2016; 5 months ago
4. OS Cross-platform
5. License: free and open-source software
6. Website: www.python.org
7. ” I chose Python as a working title for the project, being in a slightly irreverent mood (and a big fan of Monty Python’s Flying Circus).”

Features and philosophy

1. Python is a multi-paradigm programming language:
    1. object-oriented programming
    2. structured programming
    3. and many language features support functional programming
    4. and many other paradigm
2. The core philosophy of the language is summarized by the document The Zen of Python (PEP 20), which includes aphorisms such as:
    1. Beautiful is better than ugly
    2. Explicit is better than implicit
    3. Simple is better than complex
    4. Complex is better than complicated
    5. Readability counts

- Console
- IDE

Python.exe–Interactive Mode

1. When commands are read from a tty, the interpreter is said to be in interactive mode.
2. In this mode it prompts for the next command with the primary prompt, usually three greater-than signs (`>>>`);
3. for continuation lines it prompts with the secondary prompt, by default three dots (...).
4. Continuation lines are needed when entering a multi-line construct.

IPython

- IPython provides a rich toolkit to help you make the most of using Python interactively. Its main components are:
    1. A powerful interactive Python shell
    2. A Jupyter kernel to work with Python code in Jupyter notebooks and other interactive frontends.

- Console
- IDE

IDE

1. IDLE
2. Spyder
3. Redeo
4. Emacs
5. PyCharm
6. Other

- Control Flow
- Functions

`if` Statement

1. The `if` statement is used for conditional execution:
2. There can be zero or more `elif` parts, and the `else` part is optional. The keyword `elif` is short for `else if`, and is useful to avoid excessive indentation.

if Statement example

```python
x = int(input("Please enter an integer: "))
if x < 0:
    x = 0
    print('Negative changed to zero')
elif x == 0:
    print('Zero')
elif x == 1:
    print('Single')
else:
    print('More'))
```

for Statements

- Python’ s for statement iterates over the items of any sequence (a list or a string), in the order that they appear in the sequence.

```python
words = ["cat", 'window', 'defenestrate']
for w in words:
    print(w, len(w))
"""""""
```

```
cat 3
window 6
defenestrate 12
```

`range` function

1. `range(stop)`
2. `range(start, stop[, step])`
3. This is a versatile function to create lists containing arithmetic progressions.
4. It is most often used in for loops.
5. The arguments must be plain integers.
6. If the step argument is omitted, it defaults to 1.
7. If the start argument is omitted, it defaults to 0.

`break` and `continue` Statements

1. The `break` statement, breaks out of the smallest enclosing for or while loop.
2. The `continue` statement, continues with the next iteration of the loop

- Control Flow
- Functions

Defining Functions

1. The keyword `def` introduces a function definition.
2. It must be followed by the function name and the parenthesized list of formal parameters.
3. The statements that form the body of the function start at the next line, and must be indented.
4. The first statement of the function body can optionally be a string literal; this string literal is the function’ s documentation string, or docstring.

Examples

```
def fib(n):    # write Fibonacci series up to n
    """Print a Fibonacci series up to n."""
    a, b = 0, 1
    while a < n:
^^Iprint a,
^^Ia, b = b, a+b
```

Coding Style: PEP8

1. Most languages can be written (or more concise, formatted) in different styles; some are more readable than others.
2. For Python, PEP 8 has emerged as the style guide that most projects adhere to; it promotes a very readable and eye-pleasing coding style.

- 数据类型
- 切片与索引

List

1. comma-separated values (items) between square brackets [ ] . Lists might contain items of different types, but usually the items all have the same type.
2. lists can be indexed and sliced.
3. lists are a mutable type, i.e. it is possible to change their content.
4. The list data type has some more methods, use `dir(L)` to show these mothods.
5. `.remove` : Remove the first item from the list whose value is x. It is an error if there is no such item.
6. the `del` statement: remove an item from a list given its index instead of its value.

Tuple

1. A tuple consists of a number of values separated by commas.on output tuples are always enclosed in parentheses () .
2. Tuples are immutable, and usually contain a heterogeneous sequence of elements.
3. the construction of tuples containing 0 or 1 items:
    1. Empty tuples are constructed by an empty pair of parentheses;
    2. a tuple with one item is constructed by following a value with a comma (it is not sufficient to enclose a single value in parentheses).

sequence packing and unpacking

1. The statement t = 12345, 54321, 'hello!' is an example of tuple packing: the values 12345, 54321 and 'hello!' are packed together in a tuple.
2. x, y, z = t, This is called sequence unpacking and works for any sequence on the right-hand side.
3. Sequence unpacking requires the list of variables on the left to have the same number of elements as the length of the sequence.

Set

1. Python also includes a data type for sets.
2. A set is an unordered collection with no duplicate elements.
3. Basic uses include membership testing and eliminating duplicate entries.
4. Set objects also support mathematical operations like union, intersection, difference, and symmetric difference.
5. the `set()` function can be used to create sets.
6. Note: to create an empty set you have to use `set()`, not `{}`; the latter creates an empty dictionary

Examples

```python
basket = ['apple', 'orange', 'apple', 'pear', 'orange', 'banana']
fruit = set(basket)                 # create a set without duplicates
fruit
set(['orange', 'pear', 'apple', 'banana'])
'orange' in fruit                   # fast membership testing
'crabgrass' in fruit

# Demonstrate set operations on unique letters from two words
a = set('abracadabra')
b = set('alacazam')
a                                  # unique letters in a
a - b                              # letters in a but not in b
a | b                              # letters in either a or b
a & b                              # letters in both a and b
a ^ b                              # letters in a or b but not both

""""""
```

Dictionary

1. Unlike sequences, which are indexed by a range of numbers, dictionaries are indexed by keys, which can be any immutable type; strings and numbers can always be keys.
2. A pair of braces creates an empty dictionary: `{}`. Placing a comma-separated list of key:value pairs within the braces adds initial key:value pairs to the dictionary;

Examples

```
tel = {'jack': 4098, 'sape': 4139}
tel['guido'] = 4127
tel
{'sape': 4139, 'guido': 4127, 'jack': 4098}
tel['jack']
4098
del tel['sape']
tel['irv'] = 4127
tel
{'guido': 4127, 'irv': 4127, 'jack': 4098}
tel.keys()
['guido', 'irv', 'jack']
'guido' in tel
```

- 数据类型
- 切片与索引

Index(索引) and Slice(切片)

- 在 Python 中，list, tuple 以及字符串等可以遍历访问的类型都可以应用 slice 访问，Python 使用 [] 来对有序序列进行索引。
- slice 本身的意思是指切片，在这些可以遍历访问的类型中截取其中的某些部分。
- 负整数索引是从尾部开始取

```python
x = list(range(10))
x[2]
x[1:5]
x[-1]
x[2:-2]
""""""
```

- 注意：
  1. 所取的 slice 是一个 半开半闭 的区间：`l[a:b]==> l[a, b)`.
  2. 下标是从 0 开始，不是从 1 开始.

如何包含列表中最后一个元素

1. 使用超过列表长度的索引
2. 可以列出的访问下标值超出数组长度范围，不会造成越界错误，只不过仅仅返回能遍历到的元素而已
3. 另外一种办法，使用省略

```
x[0:11]
x[0:12]
x[0:]
x[:]
"""""""
```

倒序取元素

1. 用 `x[a:b]` 的方式来访问元素来看，我们这里 a, b 取的值要么满足 0<= a <= b 或者 a >= 0, b < 0。实际上，a 所对应元素的位置总是在 b 所对应位置的前面。
2. 如果 a 所对应元素的位置总是在 b 所对应位置的后面，并不是所期望的返回一个倒过来的数组，而是返回一个空的数组。

```
x[3:1]
x[-1:1]
"""""""
```

extended slice

1. `x[a:b:step]` slice of x from i to j with step k
2. 如果 a 在 b 前面，step 要取正数，否则取的切片为空
3. 如果 a 在 b 后面，step 要取负数，否则取的切片为空

```
x[1:7:2]
x[1:7:-2]
x[7:1:2]
x[7:1:-2]
x[::-1]
""""""
```

总结

1. 在 `x[a:b]` 的情况下，必须保证 a 所在的索引位置在前，b 所在的索引位置在后，否则返回结果为空。
2. 在 `x[a:b:step]` 的情况下，我们首先要根据 a, b 的位置来判断方向，a 在前，b 在后的话，step 应该为正，否则应该为负。
3. 不符合这些情况的话，则返回空的数组。也就是说，看 a, b 的位置来确定方向，不要犯方向性的错误

Module

1. A module is a file containing Python definitions and statements.
2. The file name is the module name with the suffix .py appended.

Package

1. Packages are a way of structuring Python’ s module namespace by using “dotted module names” .
2. For example, the module name A.B designates a submodule named B in a package named A.

library

- Python has a large standard library, commonly cited as one of Python's greatest strengths,providing tools suited to many tasks.
- This is deliberate and has been described as a "batteries included" Python philosophy.
- For Internet-facing applications, many standard formats and protocols (such as MIME and HTTP) are supported.
- Modules for creating graphical user interfaces, connecting to relational databases, pseudorandom number generators, arithmetic with arbitrary precision decimals, manipulating regular expressions, and doing unit testing are also included.

Module import

1. `import module1, module2`
   - Using the module name we can access the function using dot (.) operation.

```python
import math
math.sin(3)
""""""
```

2. `import modname as mm`
3. `from modname import name1, name2`
4. `from modname import *`

PyPI - the Python Package Index

- As of November, 2016, the Python Package Index, the official repository containing third-party software for Python, contains over 92,000 packages offering a wide range of functionality, including:
  1. graphical user interfaces, web frameworks, multimedia, databases, networking and communications
  2. test frameworks, automation and web scraping, documentation tools, system administration
  3. scientific computing, text processing, image processing

The dir() Function

1. The built-in function dir() is used to find out which names a module defines.
2. Without arguments, dir() lists the names you have defined currently:

查看 modules

- 查看内置函数 (builtin 模块中的函数)：`dir(__builtin__)`
- 查看内置模块 (written in C and built in to the Python interpreter)：`sys.builtin_module_names`
- 查看标准模块：
  1. CMD 中 `pip install stdlib-list`
  2. `>>> from stdlib_list import stdlib_list`
  3. `>>> libraries = stdlib_list("2.7")`
- 查看所有安装的模块：`help("modules")`，CMD 中 `pip list`
- 查看当前加载的模块：`dir()`; `sys.modules.keys()`

查看某一 package 中所有子包和子模块

- 使用标准库中的 `pkgutil` 库，如列出 `numpy` 中的所有子包 (sub packges) 和子模块 (sub modules)

```python
import pkgutil
import numpy

for importer, modname, ispkg in pkgutil.iter_modules(numpy.__
    path__, prefix="numpy."):
    print(modname)

"""""""
```

简介

1. use functions to organize code and built-in types to organize data.
2. ”object-oriented programming”：uses programmer-defined types to organize both code and data.
3. Python’s built-in types:
    1. list
    2. tuple
    3. dict
    4. set

programmer-defined type: class

1. A programmer-defined type is also called a class. A class definition looks like this:

```python
class Point:
    """Represents a point in 2-D space."""
Point()
""""""
```

2. The header indicates that the new class is called Point .
3. The body is a docstring that explains what the class is for.
4. You can define variables and methods inside a class definition.

object

1. Defining a class named `Point` creates a class object:

```
>>> Point
<class '__main__.Point'>
```

2. The class object is like a factory for creating objects.
3. To create a Point, you call Point as if it were a function

```
>>> blank = Point()
>>> blank
<__main__.Point object at 0xb7e9d3ac>
```

instance

1. The return value is a reference to a Point object, which we assign to blank.
2. Creating a new object is called instantiation, and the object is an instance of the class.
3. When you print an instance, Python tells you what class it belongs to and where it is stored in memory
4. (the prefix 0x means that the following number is in hexadecimal).
5. Every object is an instance of some class, so “object” and “instance” are interchangeable.

Attributes

1. You can assign values to an instance using dot notation:

```
>>> blank.x = 3.0
>>> blank.y = 4.0
```

2. These elements are called attributes.
3. You can read the value of an attribute using the same syntax:

```
>>> blank.y
4.0
>>> x = blank.x
>>> x
3.0
```

Functions

1. Pure Function
    - a pure function does not modify any of the objects passed to it as arguments and it has no effect, like displaying a value or getting user input, other than returning a value.
2. Modifiers
    - a function to modify the objects it gets as parameters.

Methods

- a method is a function that is associated with a particular class.

```python
a=[1,2,3]
import numpy as np
a=np.array(a)
a+1
""""""
```

- a method is called right after it is bound:

```python
x.f()
```

Methods and Functions

- Methods are semantically the same as functions, but there are two syntactic differences:
  1. Methods are defined inside a class definition in order to make the relationship between the class and the method explicit.
  2. The syntax for invoking a method is different from the syntax for calling a function.
- Methods and functions can be changed from one form to another, you can choose the best form for whatever you are doing.

Magic Methods

1. They are special methods with fixed names. They are the methods with this clumsy syntax, i.e. the double underscores at the beginning and the end.
2. don't have to invoke magic methods directly. The invocation is realized behind the scenes.
3. By convention, the first parameter of a method is called self and the second parameter other.
4. The init method (short for "initialization") is a special method that gets invoked when an object is instantiated.
5. The str is a special method, like init, that is supposed to return a string representation of an object(with `print` function).

Operating System Interface

1. The os module provides dozens of functions for interacting with the operating system.
2. For daily file and directory management tasks, the shutil module provides a higher level interface that is easier to use
3. use dir() and help() 函数

String Pattern Matching

1. The re module provides regular expression tools for advanced string processing. For complex matching and manipulation, regular expressions offer succinct, optimized solutions:
2. When only simple capabilities are needed, string methods are preferred because they are easier to read and debug

Mathematics

1. The math module gives access to the underlying C library functions for floating point math:
2. The random module provides tools for making random selections:
3. The decimal module offers a Decimal datatype for decimal floating point arithmetic.

Tools for Working with Lists

1. The array module provides an array() object that is like a list that stores only homogeneous data and stores it more compactly.
2. The collections module provides a deque() object that is like a list with faster appends and pops from the left side but slower lookups in the middle. These objects are well suited for implementing queues and breadth first tree searches:

CSV File Reading and Writing

1. The csv module implements classes to read and write tabular data in CSV format.
2. It allows programmers to say, “write this data in the format preferred by Excel,” or “read data from this file which was generated by Excel,” without knowing the precise details of the CSV format used by Excel. Programmers can also describe the CSV formats understood by other applications or define their own special-purpose CSV formats.
3. The csv module’s reader and writer objects read and write sequences.

Structured Markup Processing and Web Tools

1. HTMLParser defines a class HTMLParser which serves as the basis for parsing text files formatted in HTML (HyperText Mark-up Language) and XHTML.
2. xml pakcages.
3. The webbrowser module provides a high-level interface to allow displaying Web-based documents to users. Under most circumstances, simply calling the open() function from this module will do the right thing.
4. urllib module provides a high-level interface for fetching data across the World Wide Web. In particular, the urlopen() function is similar to the built-in function open(), but accepts Universal Resource Locators (URLs) instead of filenames.

Batteries Included

1. Python has a “batteries included” philosophy.
2. This is best seen through the sophisticated and robust capabilities of its larger packages.
3. See The Python Standard Library

运算

1. 3/2 3.0/2
2. The return type of a division (/) operation depends on its operands. If both operands are of type int, floor division is performed and an int is returned. If either operand is a float, classic division is performed and a float is returned.
3. it is possible to use the ** operator to calculate powers
4. In interactive mode, the last printed expression is assigned to the variable _.

multiple assignment

1. a, b = 0, 1: the variables a and b simultaneously get the new values 0 and 1.
2. a, b = b, a+b: expressions on the right-hand side are all evaluated first before any of the assignments take place. The right-hand side expressions are evaluated from the left to the right.

indentation

1. indentation is Python’ s way of grouping statements.
2. At the interactive prompt, you have to type a tab or space(s) for each indented line.
3. In practice you will prepare more complicated input for Python with a text editor; all decent text editors have an auto-indent facility.
4. When a compound statement is entered interactively, it must be followed by a blank line to indicate completion (since the parser cannot guess when you have typed the last line).
5. Note that each line within a basic block must be indented by the same amount.