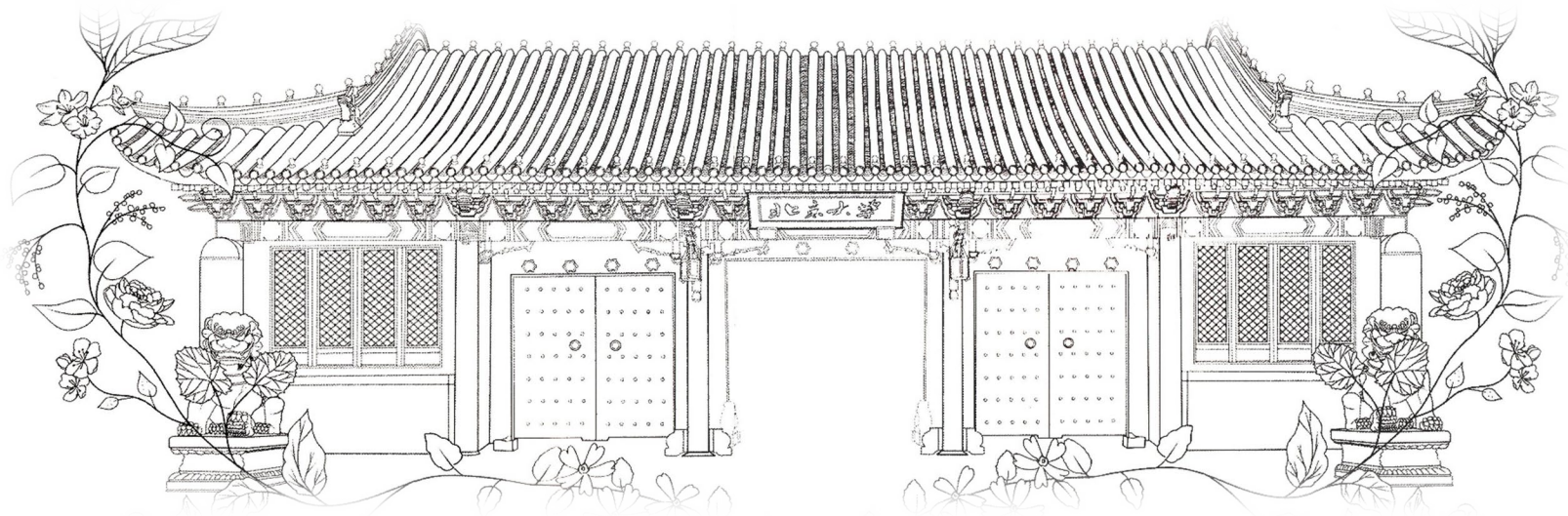




# 第 1 周内容回顾与提炼

北京大学信息管理系

2021/3/14





# 上周内容回顾

---

- 课程简介
- 数据分析能干什么？
- Python语法回顾
  - 四种数据结构（set, list, tuple, dict）
  - 两种表达形式（列表推导式、生成器表达式）
  - 两种函数（普通函数、匿名函数lambda）
  - 模块化



# 重点内容1：列表推导式和生成器表达式有何区别？

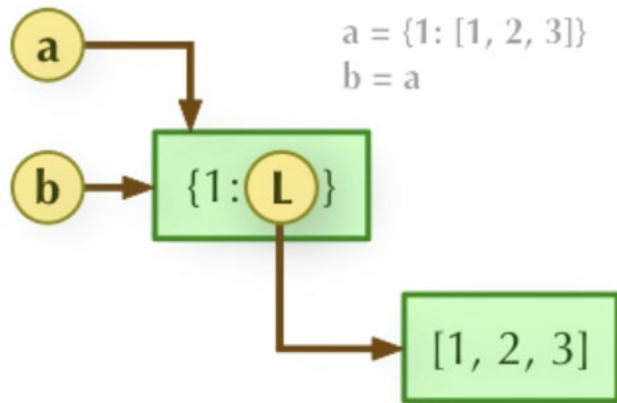
- 括号的形式
- 遍历的次数



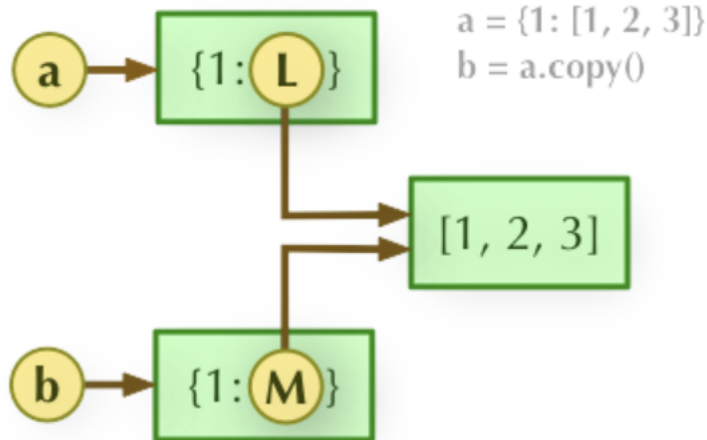
# 重点内容2：赋值与拷贝的概念

## •赋值、浅拷贝、深拷贝

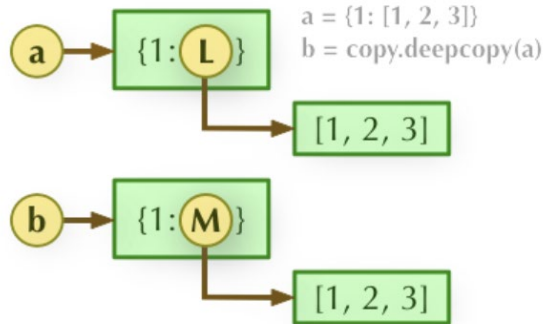
**b = a:** 赋值引用, a 和 b 都指向同一个对象。



**b = a.copy():** 浅拷贝, a 和 b 是一个独立的对象, 但他们的子对象还是指向统一对象 (是引用)。



**b = copy.deepcopy(a):** 深度拷贝, a 和 b 完全拷贝了父对象及其子对象, 两者是完全独立的。





## 重点内容3：匿名函数

---

```
sum = lambda arg1, arg2: arg1 + arg2  
print("运行结果: ", sum( 10, 20 ))  
print("运行结果: ", sum( 20, 20 ))
```



# 上周内容回顾 (续)

---

- Numpy 简介

- 创建与打印数组 (`np.array()`)
- 基本运算 (+、-、两种乘法、通用函数)
- 索引、切片和迭代
- 数组的形状操作、分割和组合
- 复制和视图 (注意和前面提到的赋值、拷贝的区别与联系！)



# array和np.ndarray

---

- array是一个方法，用于创建一个矩阵对象，而ndarray是该对象的类型

```
>>> a = np.array(1)
>>> a
array(1)
>>> type(a)
<class 'numpy.ndarray'>
>>> 
```



# 几个函数方法的比较

- `range([start:int], stop:int, [step:int])`: Python 自带函数, start、stop、step 均为 int
- `np.arange([start=None], stop=None, [step=None], dtype=None)`: 当 step 参数为非整数时 (如 `step=0.1`) , 结果往往不一致。对于这些情况, 最好使用 `linspace()`; start、stop、step 若任一个为浮点型, 那么都会生成一个浮点型序列
- `np.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None)`: 指定在 start 到 stop 的均分数值 (等差数列)
- `np.logspace(start, stop, num=50, endpoint=True, base=10.0, dtype=None)`: 返回等比数列







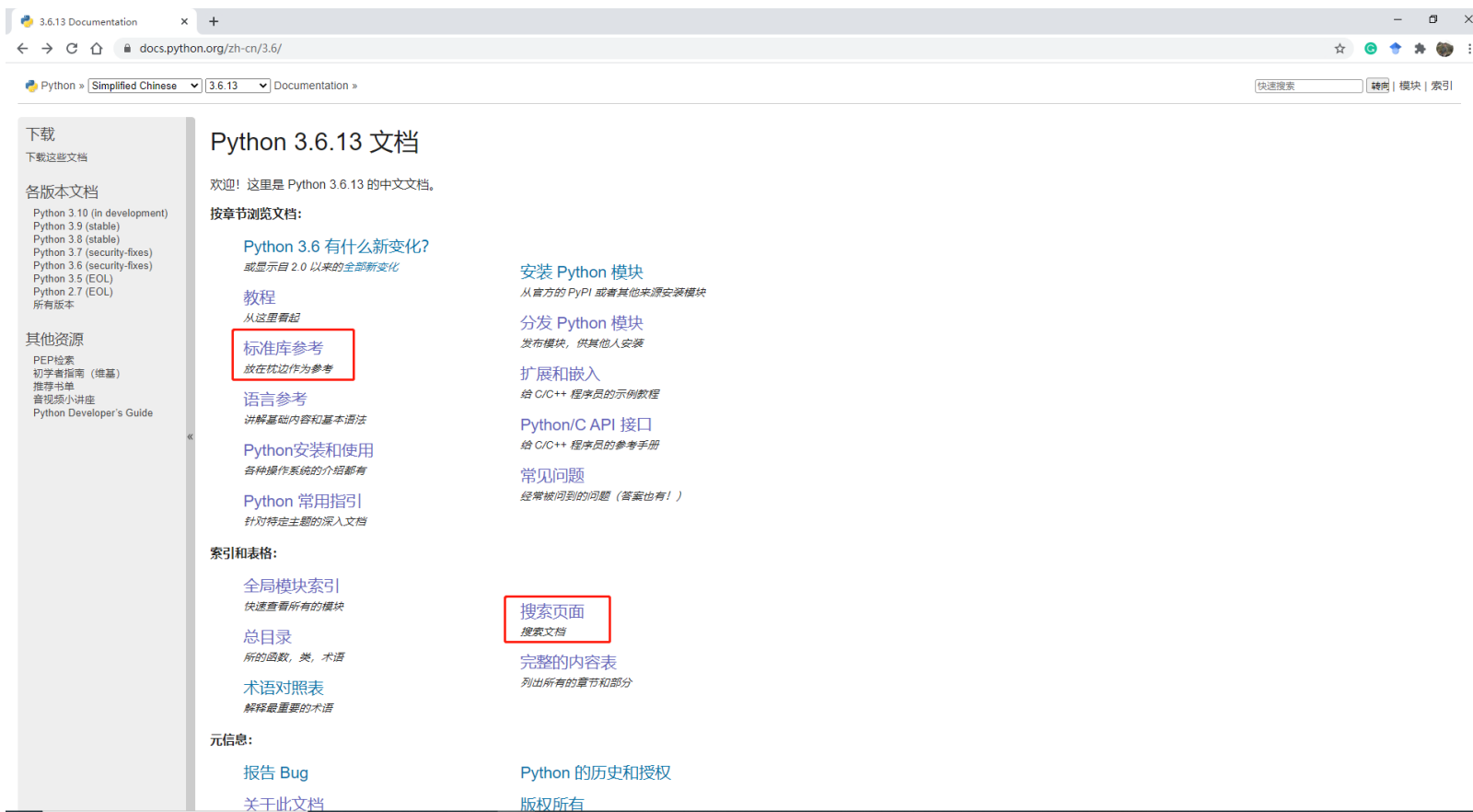
# 学会查阅帮助文档

---

- 程序或软件一般会有官方的介绍文档，主要会介绍程序或软件的基本情况、安装配置、基本功能和用法等。
- 官方文档具有权威性，并且非常细致，可以通过官方文档学习到更多内容，在使用过程中，也可能作为参考。
- 一般情况下官方文档可在官方网站直接打开。

# Python文档

- Python官方文档的网址为：<https://docs.python.org/>
- 内容非常丰富，浏览或搜索均可



The screenshot shows the Python 3.6.13 documentation website in Chinese. The browser address bar displays `docs.python.org/zh-cn/3.6/`. The page title is "Python 3.6.13 文档". The left sidebar contains navigation links for "下载" (Download), "各版本文档" (Documentation for various versions), and "其他资源" (Other resources). The main content area is titled "Python 3.6.13 文档" and includes a welcome message, a list of links for "按章节浏览文档" (Browse by chapter), and a section for "索引和表格" (Index and tables). The "按章节浏览文档" section includes links for "Python 3.6 有什么新变化?" (What's new in Python 3.6?), "教程" (Tutorial), "标准库参考" (Standard library reference), "语言参考" (Language reference), "Python安装和使用" (Installing and using Python), and "Python 常用指引" (Python frequently asked questions). The "索引和表格" section includes links for "全局模块索引" (Global module index), "总目录" (Table of contents), "术语对照表" (Glossary), and "搜索页面" (Search page). The "搜索页面" link is highlighted with a red box. The "标准库参考" link is also highlighted with a red box. The "Python 3.6 有什么新变化?" link is highlighted with a red box. The "教程" link is highlighted with a red box. The "语言参考" link is highlighted with a red box. The "Python安装和使用" link is highlighted with a red box. The "Python 常用指引" link is highlighted with a red box. The "全局模块索引" link is highlighted with a red box. The "总目录" link is highlighted with a red box. The "术语对照表" link is highlighted with a red box. The "搜索页面" link is highlighted with a red box. The "完整的内容表" link is highlighted with a red box. The "列出所有的章节和部分" link is highlighted with a red box. The "报告 Bug" link is highlighted with a red box. The "Python 的历史和授权" link is highlighted with a red box. The "关于此文档" link is highlighted with a red box. The "版权所有" link is highlighted with a red box.

3.6.13 Documentation

docs.python.org/zh-cn/3.6/

Python » Simplified Chinese » 3.6.13 » Documentation »

快速搜索 转码 模块 索引

## Python 3.6.13 文档

欢迎！这里是 Python 3.6.13 的中文文档。

**按章节浏览文档：**

- [Python 3.6 有什么新变化?](#)  
或显示自 2.0 以来的全部新变化
- [教程](#)  
从这里开始
- [标准库参考](#)  
放在枕边作为参考
- [语言参考](#)  
讲解基础内容和基本语法
- [Python安装和使用](#)  
各种操作系统的介绍都有
- [Python 常用指引](#)  
针对特定主题的深入文档

**索引和表格：**

- [全局模块索引](#)  
快速查看所有的模块
- [总目录](#)  
所有的函数，类，术语
- [术语对照表](#)  
解释最重要的术语
- [搜索页面](#)  
搜索文档
- [完整的内容表](#)  
列出所有的章节和部分

**元信息：**

- [报告 Bug](#)
- [Python 的历史和授权](#)
- [关于此文档](#)
- [版权所有](#)

**其他资源：**

- PEP检索
- 初学者指南 (维基)
- 推荐书单
- 音视频小讲座
- Python Developer's Guide

**各版本文档：**

- Python 3.10 (in development)
- Python 3.9 (stable)
- Python 3.8 (stable)
- Python 3.7 (security-fixes)
- Python 3.6 (security-fixes)
- Python 3.5 (EOL)
- Python 2.7 (EOL)
- 所有版本

**下载：**

下载这些文档



# Python文档

- 标准库部分介绍了与 Python 一同发行的标准库，首先可以在“内置函数”中进一步了解各函数的用法。

Python 标准库 — Python 3.6.13

docs.python.org/zh-cn/3.6/library/index.html

Python » Simplified Chinese » 3.6.13 » Documentation »

## Python 标准库

Python语言参考 描述了 Python 语言的具体语法和语义。这份库参考则介绍了与 Python 一同发行的标准库。它还描述了通常包含在 Python 发行版中的一些可选组件。

Python 标准库非常庞大，所提供的组件涉及范围十分广泛。正如以下内容目录所显示的，这个库包含了多个内置模块（以 C 编写），Python 程序员必须依靠它们来实现系统级功能，例如文件 I/O。此外还有大量以 Python 编写的模块，提供了日常编程中许多问题的标准解决方案。其中有些模块经过专门设计，通过将特定平台功能抽象化为平台中立的 API 来鼓励加强 Python 程序的可移植性。

Windows 版本的 Python 安装程序通常包含整个标准库，往往还包含许多额外组件。对于类 Unix 操作系统，Python 通常会分成一系列的软件包，因此可能需要使用操作系统所提供的包管理工具来获取部分或全部可选组件。

在这个标准库以外还存在成千上万并且不断增加的其他组件（从单独的程序、模块、软件包直到完整的应用开发框架）。访问 [Python 包索引](#) 即可获得这些第三方包。

- 1. 概述
- 2. 内置函数
- 3. 内置常量
  - 3.1. 由 site 模块添加的常量
- 4. 内置类型
  - 4.1. 逻辑值检测
  - 4.2. Boolean Operations — and, or, not
  - 4.3. 比较运算
  - 4.4. 数字类型 — int, float, complex
  - 4.5. 迭代器类型
  - 4.6. 序列类型 — list, tuple, range
  - 4.7. 文本序列类型 — str
  - 4.8. 二进制序列类型 — bytes, bytearray, memoryview
  - 4.9. 集合类型 — set, frozenset
  - 4.10. 映射类型 — dict
  - 4.11. 上下文管理器类型
  - 4.12. 其他内置类型
  - 4.13. 特殊属性
- 5. 内置异常
  - 5.1. 异常

2. 内置函数

Python 解释器内置了很多函数和类型，您可以在任何时候使用它们。以下按字母表顺序列出它们。

内置函数				
abs()	dict()	help()	min()	setattr()
all()	dir()	hex()	next()	slice()
any()	divmod()	id()	object()	sorted()
ascii()	enumerate()	input()	oct()	staticmethod()
bin()	eval()	int()	open()	str()
bool()	exec()	isinstance()	ord()	sum()
bytearray()	filter()	issubclass()	pow()	super()
bytes()	float()	iter()	print()	tuple()
callable()	format()	len()	property()	type()
chr()	frozenset()	list()	range()	vars()
classmethod()	getattr()	locals()	repr()	zip()
compile()	globals()	map()	reversed()	__import__()
complex()	hasattr()	max()	round()	
delattr()	hash()	memoryview()	set()	

abs(x)

返回一个数的绝对值。实参可以是整数或浮点数。如果实参是一个复数，返回它的模。

all(iterable)

如果 iterable 的所有元素为真（或迭代器为空），返回 True。等价于：





# Python文档

•这里介绍了print函数共有5个参数，及基本用法和逻辑：

1. *\*objects*：要打印的对象
2. *sep* = ' '：对象间的分隔符
3. *end* = '\n'：对象打印完后最后的结束符
4. *file* = `sys.stdout`：要打印到的文件流
5. *flush* = `False`：是否要缓存输出

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

将 *objects* 打印到 *file* 指定的文本流，以 *sep* 分隔并在末尾加上 *end*。 *sep*, *end*, *file* 和 *flush* 如果存在，它们必须以关键字参数的形式给出。

所有非关键字参数都会被转换为字符串，就像是执行了 `str()` 一样，并会被写入到流，以 *sep* 且在末尾加上 *end*。 *sep* 和 *end* 都必须为字符串；它们也可以为 `None`，这意味着使用默认值。如果没有给出 *objects*，则 `print()` 将只写入 *end*。

*file* 参数必须是一个具有 `write(string)` 方法的对象；如果参数不存在或为 `None`，则将使用 `sys.stdout`。由于要打印的参数会被转换为文本字符串，因此 `print()` 不能用于二进制模式的文件对象。对于这些对象，应改用 `file.write(...)`。

输出是否被缓存通常决定于 *file*，但如果 *flush* 关键字参数为真值，流会被强制刷新。

在 3.3 版更改：增加了 *flush* 关键字参数。





# Python文档

## • 类似的，可以查个各内置函数的文档

`open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)`

打开 `file` 并返回对应的 `file object`。如果该文件不能打开，则触发 `OSError`。

`file` 是一个 `path-like object`，表示将要打开的文件的路径（绝对路径或者当前工作目录的相对路径），也可以是要被封装的整数类型文件描述符。（如果是文件描述符，它会随着返回的 I/O 对象关闭而关闭，除非 `closefd` 被设为 `False`。）

`mode` 是一个可选字符串，用于指定打开文件的模式。默认值是 `'r'`，这意味着它以文本模式打开并读取。其他常见模式有：写入 `'w'`（截断已经存在的文件）；排它性创建 `'x'`；追加写 `'a'`（在一些 Unix 系统上，无论当前的文件指针在什么位置，所有写入都会追加到文件末尾）。在文本模式，如果 `encoding` 没有指定，则根据平台来决定使用的编码：使用 `locale.getpreferredencoding(False)` 来获取本地编码。（要读取和写入原始字节，请使用二进制模式并不要指定 `encoding`。）可用的模式有：

字符	含义
<code>'r'</code>	读取（默认）
<code>'w'</code>	写入，并先截断文件
<code>'x'</code>	排它性创建，如果文件已存在则失败
<code>'a'</code>	写入，如果文件存在则在末尾追加
<code>'b'</code>	二进制模式
<code>'t'</code>	文本模式（默认）
<code>'+'</code>	更新磁盘文件（读取并写入）
<code>'U'</code>	<code>universal newlines mode (deprecated)</code>

默认的模式是 `'r'`（打开并读取文本，同 `'rt'`）。对于二进制写入，`'w+b'` 模式打开并把文件截断成 0 字节；`'r+b'` 则不会截断。

正如在 [概述](#) 中提到的，Python 区分二进制和文本 I/O。以二进制模式打开的文件（包括 `mode` 参数中的 `'b'`）返回的内容为 `bytes` 对象，不进行任何解码。在文本模式下（默认情况下，或者在 `*mode*` 参数中包含 `'t'`）时，文件内容返回为 `str`，首先使用指定的 `encoding`（如果给定）或者使用平台默认的字节编码解码。





# Python文档

---

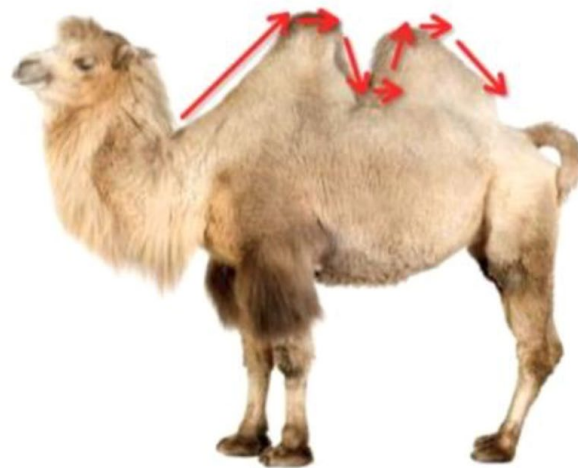
- NumPy和Pandas 同样有文档可以参考，链接地址分别为：
- NumPy中文参考手册：
  - <https://www.numpy.org.cn/reference/>
- NumPy英文文档：
  - <https://numpy.org/doc/stable/>
- Pandas中文文档：
  - <https://www.pypandas.cn/docs/>
- Pandas英文文档：
  - <https://pandas.pydata.org/docs/>
- 文档（Documentation）一般具有用户指南（介绍、安装、使用）、参考手册和开发等部分。
- 参考手册（Reference）主要用来介绍函数、模块和对象，从该技术层面指导使用。



# 标识符

为了规范命名标识符，关于标识符的命名提以下建议：

1. 见名知意
2. 类名建议使用驼峰式，如  
userName, userLoginFlag
3. 函数名建议用下划线分割，  
如get\_user\_name,  
get\_user\_login\_flag



如：  userName       userLoginFlag



# 上周练习

---

- Python基础练习
- Numpy练习