



Python 是一个高层次的结了解释性、编译性、互动性和面向对象的脚本语言。

它具有很强的可读性，丰富的工具库支撑，是在数据科学和机器学习 AI 中最广泛应用的编程语言之一。

作者 | 韩信子 @ShowMeAI

设计 | 南 乔 @ShowMeAI

参考 | datcamp cheatsheet

1. 变量与数据类型

变量赋值

```
> x=5
> x
5
```

类型与类型转换

```
> str() # 转为字符串
'5', '3.1415', 'True'
```

```
> int() # 转为整数
5, 3, 1
```

```
> float() # 转为浮点数
5.0, 1.0
```

```
> bool() # 转为布尔值
True, True, True
```

变量计算

```
> x+2 # 加
7
```

```
> x-2 # 减
3
```

```
> x*2 # 乘
10
```

```
> x**2 # 幂
25
```

```
> x%2 # 取余
1
```

```
> x/float(2) # 除
2.5
```

2. 字符串

初始化字符串

```
> my_string = 'ShowMeAI-Is-Awesome' # 单引号 / 双引号 / 三引号都可以
> my_string
'ShowMeAI-Is-Awesome'
```

字符串运算

```
> my_string * 2
'ShowMeAI-Is-AwesomeShowMeAI-Is-Awesome'
```

```
> my_string + 'Innit'
'ShowMeAI-Is-AwesomeInnit'
```

```
> 'm' in my_string
True
```

字符串操作

注意字符串的索引 index 从 0 开始

```
> my_string[3] # 根据索引取字符
> my_string[4:9] # 根据索引切片取子串
```

字符串方法

```
> my_string.upper() # 字符串字母全部大写
> my_string.lower() # 字符串字母全部小写
```

```
> my_string.count('w') # 统计某字符出现的次数
```

```
> my_string.replace('e', 'i') # 替换字符
> my_string.strip() # 清除左右空格
```

调用帮助

```
> help(str)
```

3. 列表

```
> a = 'is'
> b = 'nice'
> my_list = ['my', 'list', a, b]
> my_list2 = [[4, 5, 6, 7], [3, 4, 5, 6]]
```

选择列表元素

取元素

```
> my_list[1] # 选择索引 1 对应的值
> my_list[-3] # 选择倒数第 3 个索引对应的值
```

切片

```
> my_list[1:3] # 选取索引 1 和 2 对应的值
> my_list[1:] # 选取索引 1 及之后对应的值
> my_list[:3] # 选取索引 3 之前对应的值
> my_list[:] # 复制列表
```

子集列表的列表

```
> my_list2[1][0] # my_list[list][itemOfList]
> my_list2[1][:2]
```

列表操作

```
> my_list + my_list
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
```

```
> my_list * 2
['my', 'list', 'is', 'nice', 'my', 'list', 'is', 'nice']
```

列表方法

```
> my_list.index(a) # 获取某值的索引
> my_list.count(a) # 统计某值出现的次数
```

```
> my_list.append('!!') # 追加某值
> my_list.remove('!!') # 移除某值
```

```
> del(my_list[0:1]) # 移除某个数据切片
```

```
> my_list.reverse() # 反转列表
> my_list.extend('!!') # 添加某值
> my_list.pop(-1) # 移除并返回某值
> my_list.insert(0, '!!') # 插入某值
> my_list.sort() # 列表排序
```

4. 初始化

导入库

```
> import numpy
> import numpy as np
```

导入指定功能

```
> from math import pi
```

5. Numpy 数组

Numpy 数组创建与操作

```
> my_list = [1, 2, 3, 4]
> my_array = np.array(my_list)
> my_2darray = np.array([[1, 2, 3], [4, 5, 6]])
```

选取 Numpy 数组的值

```
> my_array[1]          # 选择索引 1 对应的值
2
> my_array[0:2]        # 选择索引 0 和 1 对应的值
array([1, 2])
> my_2darray[:, 0]     # my_2darray[rows, columns]
array([1, 4])
```

Numpy 数组运算

```
> my_array > 3
array([False, False, False,  True], dtype=bool)

> my_array * 2
array([2, 4, 6, 8])

> my_array + np.array([5, 6, 7, 8])
array([6, 8, 10, 12])
```

Numpy 数组函数

```
> my_array.shape        # 获取数组形状

> np.append(other_array) # 追加数据
> np.insert(my_array, 1, 5) # 插入数据
> np.delete(my_array, [1]) # 删除数据

> np.mean(my_array)      # 平均值
> np.median(my_array)    # 中位数
> np.corrcoef(my_array, other_array) # 相关系数
> np.std(my_array)       # 标准差
```

安装 Python

**ANACONDA**

Python 首选开源数据科学平台

**SPYDER**

Anaconda 内置的免费 IDE

**Jupyter**

创建包含代码、可视图与文本的文档

常用 Python 库 [本系列速查表包含的所有库]



Numpy 是 Python 数据科学计算的核心库，提供了高性能多维数组对象及处理数组的工具。

使用以下语句导入 Numpy 库：

```
> import numpy as np
```



SciPy 是基于 NumPy 创建的 Python 科学计算核心库，提供了众多数学算法与函数。



Pandas 是基于 Numpy 创建的 Python 库，为 Python 提供了易于使用的数据结构和数据分析工具。

使用以下语句导入 Pandas 库：

```
> import pandas as pd
```



PySpark 是 Spark 的 Python API，允许 Python 调用 Spark 编程模型。

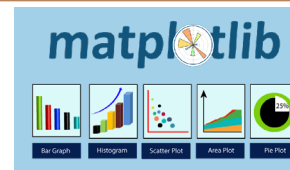
Spark SQL 是 Apache Spark 处理结构化数据的模块。



Scikit-learn 是开源的 Python 库，通过统一的界面实现机器学习、预处理、交叉验证及可视化算法。



Keras 是强大、易用的深度学习库，基于 Theano 和 TensorFlow 提供了高阶神经网络 API，用于开发和评估深度学习模型。



Matplotlib 是 Python 的二维绘图库，用于生成符合出版质量或跨平台交互环境的各类图形。

```
> import matplotlib.pyplot as plt
```



Seaborn 是基于 matplotlib 开发的高阶 Python 数据可视图库，用于绘制优雅、美观的统计图形。

使用下列别名导入该库：

```
> import matplotlib.pyplot as plt
```

```
> import seaborn as sns
```



Bokeh 是 Python 的交互式可视图库，用于生成在浏览器里显示的大规模数据集高性能可视图。Bokeh 的中间层通用 bokeh.plotting 界面主要为两个组件：数据与图符号。

```
> from bokeh.plotting import figure
```

```
> from bokeh.io import output_file, show
```



扫码回复“速查表”

下载最新全套资料