

1. 变量与数据类型

1.1 变量赋值

```
x=5
x
5
```

1.3 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
```

1.2 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
```

2. 字符串

```
my_string = 'ShowMeAI-Is-Awesome'
my_string ## 单引号 / 双引号 / 三引号都可以
'ShowMeAI-Is-Awesome'
my_string * 2
'ShowMeAI-Is-AwesomeShowMeAI-Is-Awesome'
my_string + 'Innit'
'ShowMeAI-Is-AwesomeInnit'
'm' in my_string
True
```

初始化

字符串运算

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
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') # 统计某字符出现次数
my_string.replace('e', 'i') # 替换字符
my_string.strip() # 清除左右空格
```

操作

字符串方法

3. 列表

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

3.1 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]
```

3.2 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']
```

3.3 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. Numpy 数组

4.1 Numpy 数组创建与操作

```
import numpy
import numpy as np
from math import pi
```

```
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])
```

4.2 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])
```

4.3 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
```



5. 容器

初始化

5.1 字典 / Dictionary

键值对用冒号分割, 包括在花括号中

```
dict = {'a': 1, 'c': 2, 'b': '3'}
```

取值

```
dict.items() # 取全部键值对
```

```
dict_items([('a', 1), ('c', 2), ('b', '3')])
```

```
dict.keys() # 取全部的 key
```

```
dict_keys(['a', 'c', 'b'])
```

```
dict.values() # 取全部的 values 值
```

```
dict_values([1, 2, '3'])
```

```
dict['b'] # 取 key 为 b 对应的值
```

```
'3'
```

```
dict['d'] # 取 key 为 d 对应的值, 将报错
```

```
KeyError: 'd'
```

赋值与修改

```
dict['d'] = "ShowMeAI" # 赋值, 添加
```

```
dict['c'] = 8 # 更新
```

删除

```
del dict['c'] # 删除键是 'c' 的条目
```

初始化

5.2 元组 / Tuple

```
tup1 = () # 空元组
```

```
tup1 = (50,) # 只有一个元素的元组
```

```
tup3 = ('physics', 'chemistry', 1997, 2000)
```

```
tup4 = (1, 2, 3, 4, 5, 6, 7)
```

取值

```
tup3[0] # 取 index 为 0 的值
```

```
tup4[1:5] # 取 index 从 1 到 4 的值
```

修改

```
tup1[0] = 100 # 错误! 元组的内容不允许更改
```

```
tup_new = (100,) + tup1[1:] # 重新定义一个元组
```

删除

```
del tup # 只能删除整个元组, 不可以删除某个元素
```

5.3 集合 / Set

初始化

集合 set 是无序不重复元素序列, 通过序列和 {} 定义

```
company = {"Google", "Twitter", "Taobao"}
```

添加元素

```
company.add("Facebook") # 添加
```

追加集合

```
company.update({"Tencent", "Bytedance", "ShowMeAI"})
```

删除

```
company.remove("ShowMeAI") # 如果没有这个值, 会报错
```

```
company.discard("ShowMeAI") # 有此值就删除, 没有就不操作
```

集合运算

```
a = set('abracadabra')
```

```
b = set('alacazam')
```

a

```
{'a', 'r', 'b', 'c', 'd'}
```

a - b # 集合 a 中包含而集合 b 中不包含的元素

```
{'r', 'd', 'b'}
```

a | b # 集合 a 或 b 中包含的所有元素

```
{'a', 'c', 'r', 'd', 'b', 'm', 'z', 'l'}
```

a & b # 集合 a 和 b 中都包含了的元素

```
{'a', 'c'}
```

a ^ b # 不同时包含于 a 和 b 的元素

```
{'r', 'd', 'b', 'm', 'z', 'l'}
```

Counter 用于容器计数

5.4 计数器 / Counter

```
from collections import Counter
```

```
colors = ['blue', 'blue', 'blue', 'red', 'red']
```

```
counter = Counter(colors) # Counter 计数
```

```
counter['yellow'] += 1 # 频次 +1
```

```
Counter({'blue': 3, 'red': 2, 'yellow': 1})
```

```
counter.most_common()[0] # 最高频
```

```
('blue', 3)
```

6. 流程控制

6.1 选择结构 - if 语句

```
x = int(input(" 输入一个整数: "))
```

```
if x > 0:
```

```
    print(x, ' 是一个正整数 ')
```

```
elif x < 0:
```

```
    print(x, ' 是一个负整数 ')
```

```
else:
```

```
    print('0')
```

6.2 循环结构 - for/while

```
> words = ['show', 'me', 'ai']
```

```
for w in words: #for 循环
```

```
    print(w, end=" ")
```

```
show me ai
```

```
count = 0
```

```
while (count < 9) #while 循环
```

```
    print('The count is:', count)
```

```
count = count + 1
```

7. 字典 / 列表推导式

对 0-9 的列表中每个值进行平方变换

```
> [x**2 for x in range(10)]
```

```
[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]
```

对 0-29 的列表中满足 3 的倍数的值进行平方变换

```
> [i**2 for i in range(30) if i % 3 is 0]
```

```
[0, 9, 36, 81, 144, 225, 324, 441, 576, 729]
```

```
listdemo = ['ShowMeAI', 'showmeai.tech']
```

对 listdemo 里每个值生成一个值: 长度的键值对, 构建字典

```
> {key:len(key) for key in listdemo}
```

```
{'ShowMeAI': 8, 'showmeai.tech': 14}
```

8. 遍历

0 S

1 h

```
range(10)
```

```
range(0, 10)
```

```
list(range(10))
```

```
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
for i, el in enumerate("ShowMeAI"):
```

```
    print(i, el)
```

7 I

9. 函数定义 / Function

模板

```
def function_name( parameters ):

    " 函数_文档字符串 "

    function_suite

    return [expression]
```

式例

```
def my_abs(x):
    if x >= 0:
        return x
    else:
        return -x
```



10. 类 / Class [示例]

```
class ShowMeAI:
    ' 所有内容的基类 '
    articleCount = 0
    def __init__(self, title, length):
        self.title = title
        self.length = length
        ShowMeAI.articleCount += 1

    def displayCount(self):
        print("Total Article", \
              ShowMeAI.articleCount)

    def displayArticle(self):
        print("Title: " + str(self.title), \
              "Length: " + str(self.length))
```

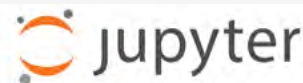
Python 是一个高层次的结了解释性、编译性、互动性和面向对象的脚本语言。它具有很强的可读性，丰富的工具库支撑，是在数据科学和机器学习 AI 中最广泛应用的编程语言之一。



Python 首选开源数据科学平台



Anaconda 内置的免费 IDE



创建包含代码、可视图与文本的文档



扫码回复“编程语言”
下载最新全套速查表

Python3 速查表

获取最新版 | <http://www.showmeai.tech/>

作者 | 韩信子 @ShowMeAI

设计 | 南乔 @ShowMeAI

参考 | DataCamp Cheatsheet

11. 正则表达式 / Regex

11.1 re.match 函数

```
import re # 正则模块使用
```

re.match 尝试从字符串的起始位置匹配一个模式。

```
import re
# 在起始位置匹配
print(re.match('www', 'www.showmeai.tech').span())
# 不在起始位置匹配
print(re.match('tech', 'www.showmeai.tech'))
```

(0, 3) None

11.2 re.search 方法

re.search 扫描整个字符串并返回第一个成功的匹配。

```
import re
line = "ShowMeAI is awesome website";
searchObj = re.search(r'(.*) is (.*) .*', line, re.M|re.I)
if searchObj:
    print("searchObj.group() : ", searchObj.group())
    print("searchObj.group(1) : ", searchObj.group(1))
    print("searchObj.group(2) : ", searchObj.group(2))
else:
    print("Nothing found!!")
```

searchObj.group() : ShowMeAI is awesome website

searchObj.group(1) : ShowMeAI

searchObj.group(2) : awesome

11.3 检索和替换

Python 的 re 模块提供了 re.sub 用于替换字符串中的匹配项。

```
> import re
phone = "2021-969-559 # 这是一个国外电话号码"

# 删除字符串中的 Python 注释
num = re.sub(r'#[.*$]', "", phone)
print(" 电话号码是 : ", num)

# 删除非数字 (-) 的字符串
num = re.sub(r'\D', "", phone)
print(" 电话号码是 : ", num)
```

电话号码是 : 2021-969-559

电话号码是 : 2021969559

数据科学工具库速查表



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：

```
import pandas as pd
```



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：数据与图示例。

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

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

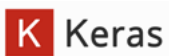


PySpark 是 Spark 的 Python API，允许 Python 调用 Spark 编程模型。Spark SQL 是 Apache Spark 处理结构化数据模块。

AI 垂直领域工具库速查表



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。



“TensorFlow™ is an open source software library for numerical computation using data flow graphs.” **TensorFlow** 是 Google 公司开发的机器学习架构，兼顾灵活性和扩展性，既适合用于工业生产也适合用于科学研究。



PyTorch 是 Facebook 团队 2017 年初发布的深度学习框架，有利于研究人员、爱好者、小规模项目等快速搞出原型。**PyTorch** 也是 Python 程序员最容易上手的深度学习框架。



Hugging Face 以开源的 NLP 预训练模型库 **Transformers** 而广为人知，目前 GitHub Star 已超过 54000+。**Transformers** 提供 100+ 种语言的 32 种预训练语言模型，简单，强大，高性能，是新手入门的不二选择。



OpenCV 是一个跨平台计算机视觉库，由 C 函数 /C++ 类构成，提供了 Python、MATLAB 等语言的接口。**OpenCV** 实现了图像处理和计算机视觉领域的很多通用算法。

编程语言速查表



SQL 是管理关系数据库的结构化查询语言，包括数据的增删查改等。作为数据分析的必备技能、岗位 JD 的重要关键词，SQL 是技术及相关岗位同学一定要掌握的语言。



Python 编程语言简洁快速、入门简单且功能强大，拥有丰富的第三方库，已经成为大数据和人工智能领域的主流编程语言。

More...

AI 知识技能速查表



Jupyter Notebook 交互式计算环境，支持运行 40+ 种编程语言，可以用来编写漂亮的交互式文档。这个教程把常用的基础功能讲解得很清楚，对新手非常友好。



正则表达式 非常强大，能匹配很多规则的文本，常用于文本提取和爬虫处理。这也是一门令人难以捉摸的语言，字母、数字和符号堆在一起，像极了“火星文”。

More...



ShowMeAI 速查表 (©2021)

获取最新版 | <http://www.showmeai.tech/>

作者 | 韩信子 @ShowMeAI

设计 | 南乔 @ShowMeAI

数据科学工具库速查表

扫码回复“数据科学”

获取最新全套速查表

AI 垂直领域工具库速查表

扫码回复“工具库”

获取最新全套速查表

编程语言速查表

扫码回复“编程语言”

获取最新全套速查表

AI 知识技能速查表

扫码回复“知识技能”

获取最新全套速查表