

Python简介

温灿红

中国科学技术大学管理学院

大纲

- Jupyter notebook基本使用
- 基本数据类型
- 数据结构
- 控制流
- 函数
- 模块

Jupyter Notebook

- Jupyter Notebook 是一种Web应用，也是一个交互式笔记本，它能让用户将说明文本、数学方程、代码和可视化内容全部组合到一个易于共享的文档中，非常便于研究、展示和教学。
- 有以下优势：
 1. 可用于编写数据分析报告
 2. 支持多语言编程
 3. 用途广泛（包括数据分析的各项工作）
 4. 分享便捷
 5. 可远程运行
 6. 交互式展现

Jupyter Notebook 的打开

- 本地打开
 - 直接从Anaconda Navigator中单击进入
 - 从Anaconda Prompt 终端进入

```
In [ ]: jupyter notebook          # 直接打开Notebook主界面
        jupyter notebook notebook.ipynb  # 打开特定的Notebook 文件
```

- 在线打开
 - **Google colab**
 - **Kaggle Notebooks**

Jupyter Notebook 的界面

 jupyter

QuitLogout

FilesRunningClusters

Select items to perform actions on them.

UploadNew↺

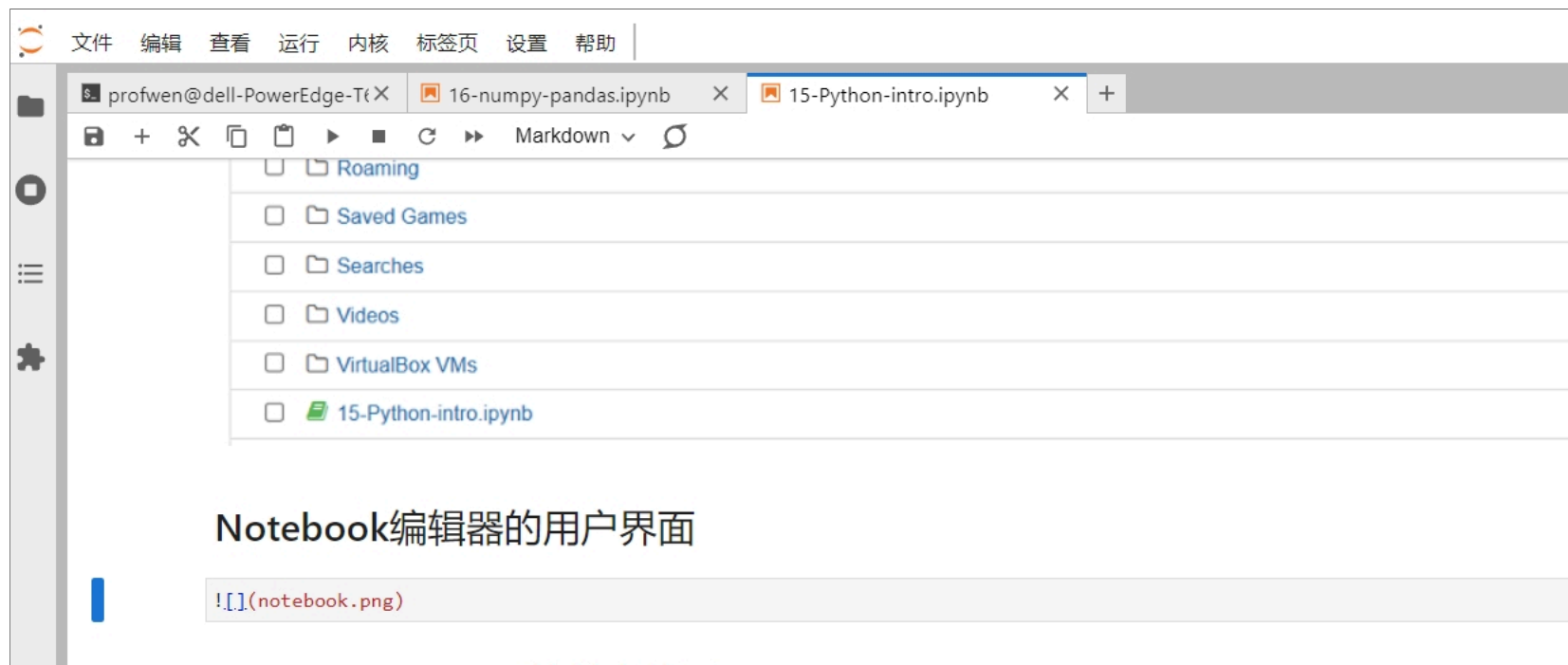
☐ 0 ▾

📁 /

Name ▾Last ModifiedFile size

<input type="checkbox"/>	📁 3D Objects	10 个月前	
<input type="checkbox"/>	📁 Contacts	10 个月前	
<input type="checkbox"/>	📁 Desktop	5 天前	
<input type="checkbox"/>	📁 Documents	2 个月前	
<input type="checkbox"/>	📁 Downloads	11 分钟前	
<input type="checkbox"/>	📁 Favorites	10 个月前	
<input type="checkbox"/>	📁 Links	10 个月前	
<input type="checkbox"/>	📁 Music	10 个月前	
<input type="checkbox"/>	📁 OneDrive	4 个月前	
<input type="checkbox"/>	📁 Pictures	10 个月前	
<input type="checkbox"/>	📁 Roaming	4 年前	
<input type="checkbox"/>	📁 Saved Games	10 个月前	
<input type="checkbox"/>	📁 Searches	10 个月前	
<input type="checkbox"/>	📁 Videos	10 个月前	
<input type="checkbox"/>	📁 VirtualBox VMs	3 年前	
<input type="checkbox"/>	📄 15-Python-intro.ipynb	运行 2 分钟前	2.23 kB

Notebook编辑器的用户界面



Jupyter Notebook的基本使用（一）

1.运行代码： 点击“运行”或者使用快捷键【Shift+Enter】

```
In [2]: print("Hello World!")

print(1, "+", 2, "=", 1 + 2)

name = input("请输入你的名字: ")
print("你好, ", name)
```

Hello World!
1 + 2 = 3

你好, Canhong Wen

2.停止运行： 在命令模式下连续按下两次【I】 键

In [10]:

```
import time
time.sleep(10)
```

类型

```
KeyboardInterrupt                                Traceback (most recent call last)
<ipython-input-10-c20d360cf4b6> in <module>
      1 import time
----> 2 time.sleep(10)
```

KeyboardInterrupt:

Jupyter Notebook的基本使用（二）

3.结果输出：点击“运行”后输出到命令行

- 默认是只输出最后一行命令的结果，如果需要输出所有的结果，需要进行适当的修改。

```
In [4]: from IPython.core.interactiveshell import InteractiveShell  
  
InteractiveShell.ast_node_interactivity = "all"
```

4.文档存储和分享

- Notebook有自动存储的功能，默认以.ipynb格式存储。
- 其他格式可通过菜单栏中的【File】->【Download】命令，然后将文档存储为Python、HTML、Markdown、Latex和PDF等格式。

Jupyter Notebook的基本使用（三）

- Python内置 `open()` 函数可用于读写数据，基本流程是：打开文件 -> 读取文件内容 -> 关闭文件。
- 基本语法是 `f = open(<文件名>, <打开模式>)`
- 七种基本的打开模式

打开模式	说明
'r'	只读模式（默认模式），如果文件不存在则报错，如果文件存在则正常读取
'w'	覆盖写模式，如果文件不存在则新建文件并写入；如果文件存在则清空文件内容写入
'x'	创建写模式，如果文件存在则报错，如果文件不存在则新建文件并写入内容，比'w'模式更安全
'a'	追加写模式，如果文件不存在则新建文件并写入，如果文件存在则在文件的最后追加写入
'b'	二进制文件模式，如rb,wb,ab是指以bytes类型操作数据
't'	文本文件模式

打开模
式

说明


'+'

与r/w/x/a一同使用，在原有功能的基础上增加同时读写功能

```
In [5]: f = open("test.txt", "w", encoding="utf-8")
        f.write("Hello World!\n实用统计软件课程\n")
        f.close()
```

Out[5]:

22

 jupyter test.txt ✓ 3 小时前

File	Edit	View	Language
1	2	3	
			Hello World!
			实用统计软件课程

Jupyter Notebook的基本使用（四）

- 一般的数据保存在.txt, .csv, .xlsx中，pandas库中提供了一些可以将表格型数据读取为DataFrame对象的函数。

函数	说明
read_csv	从文件、URL、文件型对象中加载带分隔符的数据。默认分隔符为逗号
read_table	从文件、URL、文件型对象中加载带分隔符的数据。默认分隔符为制表符("\t")
read_fwf	读取定宽列格式的数据（也就是说，没有分隔符）
read_clipboard	读取剪贴板中的数据，可以看作是read_table的剪贴板版本。在将网页转换成表格时很有用

- 如果是本地文件，首先要查看当前的工作目录并修改至适当的路径下

```
In [8]: import os

os.getcwd() # 查看当前的工作目录

#os.chdir("dir") # 修改工作目录
```

Out[8]:

' /home/profwen'

```
In [9]: import pandas as pd

coupon = pd.read_csv("coupon_nm.csv", encoding="gbk")
coupon.head()
```

Out[9]:

	团购名	店名	团购活动ID	团购介绍	购买人数	团购评价	评价人数	到期时间	团购价	市场价	地址	团购内容	备注	购买须知
0	壹分之贰豪华生日派对套餐	壹分之贰聚会吧桌游轰趴馆(旗	38744470	仅售880元, 价值7342元豪华生日派对聚会桌游轰趴夜场10小时(通宵包场)!	0	暂无评价	暂无评价	2018-08-10	NaN	NaN	【西安市雁塔区小寨东路11号壹又贰分之壹B楼1003】	【内容: 豪华主题场地, 规格: 10小时, 价格: 600元】 【内容: 轰趴	随便退	【有效期2017年06月07日至2018年08月10日】 \n【可用时间周

	团购名	店名	团购活动ID	团购介绍	购买人数	团购评价	评价人数	到期时间	团购价	市场价	地址	团购内容	备注	购买须知
		舰店)		免费WiFi, ...								场地布置, 规...		未法定节假日通用08...
1	壹分之贰豪华棋牌包间	壹分之贰聚会吧桌游轰趴馆(旗舰店)	38632123	仅售98元, 价值240元豪华棋牌包间4小时套餐 (4人) 1份! 免费WiFi!	0	暂无评价	暂无评价	2018-08-10	NaN	NaN	【西安市雁塔区小寨东路11号壹又贰分之壹B楼1003】	【内容: 豪华棋牌包间4小时套餐 (4人), 规格: 1份, 价格: 240元】	随便退	【有效期2017年05月26日至2018年08月10日】\n【可用时间周末法

团购名	店名	团购活动ID	团购介绍	购买人数	团购评价	评价人数	到期时间	团购价	市场价	地址	团购内容	备注	购买须知	
													定节假日通用08...	
2	壹分之贰豪华生日派对套餐	壹分之贰聚会吧桌游轰趴馆(旗舰店)	38744514	仅售480元, 价值3685元豪华生日派对聚会桌游轰趴5小时(包场)！ 免费WiFi, 需预约!	0	暂无评价	暂无评价	2018-08-10	NaN	NaN	【西安市雁塔区小寨东路11号壹又贰分之壹B楼1003】	【内容: 豪华主题场地, 规格: 5小时, 价格: 300元】 【内容: 轰趴场地布置,	随便退	【有效期2017年06月07日至2018年08月10日】 \n【可用时间周末法定节假日

团购名	店名	团购活动ID	团购介绍	购买人数	团购评价	评价人数	到期时间	团购价	市场价	地址	团购内容	备注	购买须知	
											规格...		假日通用08...	
3	396元1-15人服务	壹分之贰聚会吧桌游轰趴馆(旗舰店)	13377118	仅售396元, 价值2575元台球优享套餐! 免费WiFi, 需预约!	8	5	3	2018-09-04	396.0	2575.0	【西安市雁塔区小寨东路11号壹又贰分之壹B楼1003】	【内容: 复式结构场地, 规格: 1次, 价格: 200元】 【内容: 场地布置, 规格: 1...	随便退	【有效期2016年05月06日至2018年09月04日】 \n【可用时间周末法定节假日

团购名	店名	团购活动ID	团购介绍	购买人数	团购评价	评价人数	到期时间	团购价	市场价	地址	团购内容	备注	购买须知	
													通用24...	
4	六婆串串香火锅100代金券	六婆串串香火锅	38767590	仅售85元, 价值100元代金券! 免费停车, 免费WiFi!	6	暂无评价	暂无评价	2018-08-30	85.0	100.0	【西安市雁塔区长丰园小区12号楼7号商铺】 【西安市周至县中心街隆发云塔广场C016号】	【内容: 代金券, 规格: 1张, 价格: 100元】	随便退	【有效期2017年06月09日至2018年08月30日】 \n【可用时间周末法定节假日

团购名	店名	团购活动ID	团购介绍	购买人数	团购评价	评价人数	到期时间	团购价	市场价	地址	团购内容	备注	购买须知
-----	----	--------	------	------	------	------	------	-----	-----	----	------	----	------

通用
09...

基本数据类型

基本数据类型

1. 整型 (int)
2. 浮点型 (float)
3. 布尔型 (bool)
4. 复数 (complex)
5. 字符串(str)
6. 时间类型(pandas模块)

查看和转换数据类型

- 查看数据类型：
 - `type` 查看单个类型的数据
 - `dtypes` 查看多个类型的所有数据类型
- 转换数据类型
 - `int()`
 - `float()`
 - `bool()`
 - `str()`

例子（一）

```
In [10]: type(200)
          type(1.0)
type(True)
print(3 * 0.15 == 0.45)
print("复数", 0 + 1j, "的乘法: ", (0 + 1j) * (0 + 1j), ". 实部是", (0 + 1j).real, " 虚部是", (0 + 1j).imag)
s = "实用" + "统计软件"
s
```

Out[10]:

int

Out[10]:

float

Out[10]:

bool

False

复数 1j 的乘法: (-1+0j) . 实部是 0.0 虚部是 1.0

Out[10]:

'实用统计软件'

例子 (二)

```
In [11]: int(-2.8)
          float(2)
int("123")
bool(-2)
bool(0)
str(234)
int("a")
```

Out[11]:

-2

Out[11]:

2.0

Out[11]:

123

Out[11]:

True

Out[11]:

False

Out[11]:

'234'

ValueError Traceback (most recent call last)

Cell In[11], line 7

5 bool(0)

6 str(234)

----> 7 int("a")

ValueError: invalid literal for int() with base 10: 'a'

算子和表达式

- 算子包括

- 四则运算: `+`, `-`, `*`, `/`, `**`, `//`, `%`
- 逻辑运算: `==`, `>`, `<`, `<=`, `>=`, `!=`, `and`, `or`, `not`
- 字符串运算: `+`, `*`, `in`, `len`, 单个索引或切片索引, `isX` (如 `isalpha()`, `isalnum()`, `isdecimal()`, `isspace()`, `istitle()`)
- 取下标等等

四则运算和逻辑运算

```
In [12]: 1 + 2
```

```
Out[12]:
```

```
3
```

```
In [13]: 2**3
```

```
Out[13]:
```

```
8
```

```
In [14]: 1 > 2 or 1 != 2
```

```
Out[14]:
```

```
True
```

```
In [15]: not False
```

```
Out[15]:
```

```
True
```

```
In [16]: (0 + 1j) > 1
```

TypeError Traceback (most recent call last)

Cell In[16], line 1

----> 1 (0 + 1j) > 1

TypeError: '>' not supported between instances of 'complex' and 'int'

字符串的基本运算(一)

```
In [17]: "统计软件" + "Python"
```

Out[17]:

'统计软件Python'

```
In [18]: "统计软件" + "Python" + 2021
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[18], line 1  
----> 1 "统计软件" + "Python" + 2021
```

TypeError: can only concatenate str (not "int") to str

```
In [19]: "统计软件" + "Python" + str(2021)
```

Out[19]:

'统计软件Python2021'

```
In [20]: len("统计软件")
```

Out[20]:

字符串的基本运算(二)

```
In [21]: "统计软件" * 5
```

Out[21]:

'统计软件统计软件统计软件统计软件统计软件'

```
In [22]: "统计软件" * 5.0
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[22], line 1  
----> 1 "统计软件" * 5.0
```

TypeError: can't multiply sequence by non-int of type 'float'

```
In [23]: "统计软件"[2:4]
```

Out[23]:

'软件'

```
In [24]: "Applied Statistical Software".split()
```

Out[24]:

['Applied', 'Statistical', 'Software']

数据结构

数据结构

- 列表 (list)
- 元组 (tuple)
- 字典 (dictionary)
- 集合 (set)

列表 (list)

- 列表是由一系列按照特定顺序排列的元素组成的。
- 列表中的不同元素通过逗号分隔，整体放在一个方括号"[]"中。
- 列表中的数据类型可以是相同的，也可以是不同的，甚至可以是列表。
- 可通过方括号"[]"或者 `list()` 创建一个列表，也可通过 `range()` 构建一个规则序列的列表（类似于R中的 `seq()` 函数）。

创建列表

```
In [25]: list0 = []  
list0
```

Out[25]:

```
[]
```

```
In [26]: list1 = ["团购名", "id", 123456, 0 + 3j]  
list1
```

Out[26]:

```
['团购名', 'id', 123456, 3j]
```

```
In [27]: list("abcdefg")
```

Out[27]:

```
['a', 'b', 'c', 'd', 'e', 'f', 'g']
```

```
In [28]: list(range(10))
```

Out[28]:

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

```
In [29]: list(range(20, 10, -2))
```

Out[29]:

[20, 18, 16, 14, 12]

列表的基本操作

- 查看类型: `type(list)`
- 添加一个元素:
 - `mylist.append(new_element)`
 - `mylist.insert(position, new_element)`
- 列表的连接: `+`
- 列表长度: `len()`
- 删除列表: `del`
- 列表索引和切片: `list[i]`, `list[:length]`, `list[start:]`, `list[start:end:step]`, `list[-i]`
- 列表中是否存在某个元素: `in`, `not in`
- 寻找某个元素: `index()`
- 排序: `sort` 和 `sorted()`

例子（一）

```
In [30]: list1.append("ABC")  
list1
```

Out[30]:

```
['团购名', 'id', 123456, 3j, 'ABC']
```

```
In [31]: list1.insert(0, "店名")  
list1
```

Out[31]:

```
['店名', '团购名', 'id', 123456, 3j, 'ABC']
```

```
In [32]: len(list1)
```

Out[32]:

```
6
```

```
In [33]: list1 + list("abcdefg")
```

Out[33]:


```
['店名', '团购名', 'id', 123456, 3j, 'ABC', 'a', 'b', 'c', 'd', 'e', 'f', 'g']
```

例子 (二)

```
In [34]: list5 = list(range(10))
```

```
In [35]: list5[:]
```

Out[35]:

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

```
In [36]: list5[0]
```

Out[36]:

```
0
```

```
In [37]: list5[:3]
```

Out[37]:

```
[0, 1, 2]
```

```
In [38]: list5[2:7:2]
```

Out[38]:

```
[2, 4, 6]
```

```
In [39]: list5[-1]
```

```
Out[39]:
```

```
9
```

例子 (三)

```
In [40]: L = [5, 3, 7, 1]
         L.sort() # 永久改变列表
L
```

Out[40]:

```
[1, 3, 5, 7]
```

```
In [41]: L = [5, 3, 7, 1]
         sorted(L) # 临时改变列表
```

Out[41]:

```
[1, 3, 5, 7]
```

```
In [42]: L
```

Out[42]:

```
[5, 3, 7, 1]
```

```
In [43]: sorted(L, reverse=True)
```

Out[43]:

[7, 5, 3, 1]

元组 (tuple)

- 元组是一种有序的数据类型，类似于列表，但一旦建立后不能对其元素进行修改，因此更安全和运行速度更快。
- 元组使用小括号 () 创建，元素之间用逗号隔开。

例子 (一)

```
In [44]: tuple1 = (2, 4, 5)
         tuple2 = ("a", "b", "c")
```

```
In [45]: tuple2[1]
```

Out[45]:

'b'

```
In [46]: tuple2[2:3]
```

Out[46]:

('c',)

```
In [47]: tuple2[2] = 4
```

TypeError

Traceback (most recent call last)

Cell In[47], line 1

----> 1 tuple2[2] = 4

TypeError: 'tuple' object does not support item assignment

```
In [48]: tuple2 + tuple1
```

Out[48]:

```
('a', 'b', 'c', 2, 4, 5)
```


字典 (dictionary)

- 字典是一种保存“键值对”的数据结构
- 使用花括号 { } 可以创建字典，字典用键-值(key-value)存储，其中key和value用 : 对应。
- 常用的函数操作有
 - `keys()` : 返回所有键的信息
 - `values()` : 返回所有值的信息
 - `items()` : 返回所有键值的信息
 - `get()` : 如果键存在，则返回相应的值；如果键不存在，则结果为 `get()` 方法设置的默认值

例子（一）

```
In [49]: dict0 = {"海底捞": 10, "海银海记": 8}
```

```
In [50]: dict0["海底捞"] # 等价于 dict0.get("海底捞")
```

Out[50]:

10

```
In [51]: dict0.keys()
```

Out[51]:

dict_keys(['海底捞', '海银海记'])

```
In [52]: dict0.values()
```

Out[52]:

dict_values([10, 8])

```
In [53]: dict0["老北京涮肉"] = 0 # 新增内容
          dict0
```

Out[53]:

{'海底捞': 10, '海银海记': 8, '老北京涮肉': 0}

```
In [54]: del dict0["老北京涮肉"]  
dict0
```

Out[54]:

```
{'海底捞': 10, '海银海记': 8}
```

集合 (set)

- 集合和字典一样，是无序的，但集合中的元素不可重复。
- 集合可以理解为只有键没有值的字典。
- Python中使用集合主要有两个重要的功能：
 1. 集合操作
 2. 消除重复元素
- 创建集合：使用花括号 { } 或者 `set()` 创建。
- 运算： $-$, $\&$, $|$

例子 (一)

```
In [55]: set1 = {1, 2, "s", 1, 1}
          set1
```

Out[55]:

```
{1, 2, 's'}
```

```
In [56]: set2 = set([1, 3, 4, 5])
          set2
```

Out[56]:

```
{1, 3, 4, 5}
```

```
In [57]: set1 - set2
```

Out[57]:

```
{2, 's'}
```

```
In [58]: set1 | set2
```

Out[58]:

```
{1, 2, 3, 4, 5, 's'}
```

```
In [59]: set3 = {"北京", "上海", "广州", "深圳"}  
         set3.add("合肥")  
         set3
```

Out[59]:

```
{'上海', '北京', '合肥', '广州', '深圳'}
```

```
In [60]: set3.remove("合肥")  
         set3
```

Out[60]:

```
{'上海', '北京', '广州', '深圳'}
```

控制流

分支结构

- if-else 分支结构:

```
In [61]: x = int(input("请输入一个整数: "))
        if x >= 0:
            print(x)
        else:
            print(-x)
```

1

- if-elif-else 分支结构:

```
In [62]: x = int(input("请输入一个整数: "))
        if x > 0:
            print(x)
        elif x < 0:
            print(-x)
        else:
            print(0)
```

1

循环结构

- for

```
In [63]: s = 0
         for i in range(10):
             s = s + i
         print(s)
```

45

```
In [64]: s = 0
         for i in range(10):
             s = s + i
         print(s)
```

0
1
3
6
10
15
21
28

36

45

循环结构(续)

- while

```
In [65]: i = 1
         while i**2 < 100:
             print(i, "的平方是", i**2)
             i = i + 1
```

```
1 的平方是 1
2 的平方是 4
3 的平方是 9
4 的平方是 16
5 的平方是 25
6 的平方是 36
7 的平方是 49
8 的平方是 64
9 的平方是 81
```

循环结构-跳出循环

- break：跳出整个循环
- continuous：跳出当前的迭代，继续往下迭代。

```
In [66]: l = [1, 3, 65, 3, -1, 56, -10]
         for x in l:
             if x < 0:
                 break
         print("列表l中出现的第一个负数为", x)
```

列表l中出现的第一个负数为 -1

```
In [67]: from math import sqrt
         l = [1, 3, 65, 3, -1, 56, -10]
         for x in l:
             if x < 0:
                 continue
             print(f"{x} 的平方根为 {sqrt(x): .3f}")
```

1 的平方根为 1.000
3 的平方根为 1.732
65 的平方根为 8.062
3 的平方根为 1.732
56 的平方根为 7.483

函数

函数

- Python函数的定义如下：

```
In [ ]: def 函数名 (<参数列表>
        函数主体
        return <返回值列表>
```

- 函数名：符合Python命名规则的任意有效表示符。
- 参数列表：调用该函数时传递给它的值，多个参数之间用逗号隔开。
- return：产生函数返回值，其中多条返回语句可被接受，如果没有遇到return语句，则自动返回None。

例子

```
In [69]: def double(x):  
         "This function multiplies its argument by two."  
         return x * 2
```

```
In [70]: help(double)
```

Help on function double in module __main__:

```
double(x)  
    This function multiplies its argument by two.
```

```
In [71]: double(4)
```

Out[71]:

8

```
In [72]: double("abs")
```

Out[72]:

'absabs'

函数参数的默认值

- 输入参数带默认值，只需要在参数定义的时候赋值
- 在定义函数的时候，指定默认的参数必须在所有的参数的最后，否则将产生语法错误。
- 默认参数必须指向不可变的对象。

例子

```
In [73]: def double(x=2):  
         "This function multiplies its argument by two."  
         return x * 2  
double()
```

Out[73]:

4

```
In [74]: def mul(x=2, y):  
         "This function multiplies its argument by two."  
         return x * 2  
double()
```

Cell In[74], line 1

```
def mul(x=2, y):  
    ^
```

SyntaxError: non-default argument follows default argument

```
In [75]: def mul(x, y=2):  
         "This function multiplies its argument by two."  
         return x * y  
mul(2)
```

Out[75]:

函数的可变参数

- 可变参数可以使得传入函数的参数个数不限，可以是0个，1个，任意个。
- 可变参数的形式有 `*parameter` 和 `** parameter` 两种。
 - `*parameter`可以接收任意多个参数，并将这些参数放在一个元组中。参数传递时，按照位置传递。
 - `** parameter` 接收任意多个参数并将参数放在一个字典中。在参数传递时，按照参数名传递。

例子（一）

```
In [76]: def tv(*name):  
         print("我喜欢看的电视剧有：")  
         for item in name:  
             print(item)
```

```
In [77]: tv("琅琊榜", "庆余年")
```

我喜欢看的电视剧有：
琅琊榜
庆余年

```
In [78]: name = ["琅琊榜", "庆余年"]  
         tv(*name)
```

我喜欢看的电视剧有：
琅琊榜
庆余年

例子 (二)

```
In [79]: def tv2(**name):  
         print("我喜欢看的电视剧有：")  
         for key, value in name.items():  
             print(key + "的播放量为：" + value)
```

```
In [80]: tv2(庆余年="100", 琅琊榜="100")
```

我喜欢看的电视剧有：
庆余年的播放量为：100
琅琊榜的播放量为：100

```
In [81]: dict0 = {"庆余年": "100", "琅琊榜": "100"}  
         tv2(**dict0)
```

我喜欢看的电视剧有：
庆余年的播放量为：100
琅琊榜的播放量为：100

函数的返回值

- 可返回多个值

```
In [82]: def dou_tri(x=2):  
         "This function multiplies its argument by two."  
         return x * 2, x * 3
```

```
dou_tri(1)
```

Out[82]:

(2, 3)

```
In [83]: def dou_tri(x=2):  
         "This function multiplies its argument by two."  
         return {"double": x * 2, "triple": x * 3}
```

```
dou_tri(1)
```

Out[83]:

{'double': 2, 'triple': 3}

Map函数

- `map` 函数通过输入列表，然后对列表里的元素一一进行运算后输出一个计算后的列表。
其实就是实现向量化运算。

```
In [84]: s = "12 43 64 6"  
        L = s.split()  
L
```

```
Out[84]:  
['12', '43', '64', '6']
```

```
In [85]: int(L)
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[85], line 1  
----> 1 int(L)
```

```
TypeError: int() argument must be a string, a bytes-like object or a number, not 'list'
```

```
In [86]: list(map(int, L))
```

```
Out[86]:
```

[12, 43, 64, 6]

匿名函数

- Python中的匿名函数称为lambda表达式。
- 语法为： `lambda param1, param2,...: expression`

```
In [87]: L = [2, 3, 5]
         list(map(lambda x: 2 * x + x**2, L))
```

Out[87]:

[8, 15, 35]

filter 函数

- Python中的 `filter` 函数通过输入一个判断函数和列表，输出使得判断函数为 `True` 的子列表。
- 语法为： `filter(func, list)`

```
In [88]: def is_odd(x):  
         return x % 2 == 1
```

```
L = [1, 4, 5, 9, 10]  
list(filter(is_odd, L))
```

Out[88]:

```
[1, 5, 9]
```

模块

模块

- 当一个函数比较简单时，写一个文件即可。
- 但如果函数越来越复杂，同时涉及到多个函数，为了便于管理与维护，需要分成不同的文件存放，这样子按照不同类别存放文件就形成了不同的模块。
- Python的模块实际就是包含函数和其他语句的Python脚本文件，扩展名为.py。
- Python的标准库中包含了成千上万的模块，常用的基本模块有：
 - math
 - random
 - os
 - sys

模块(续)

- Python中重要的模块有：
 - numpy
 - pandas
 - matplotlib
 - statsmodels
 - scikit-learn

例子

```
In [89]: import numpy as np
```

```
my_arr = np.arange(1000000)  
my_list = range(1000000)
```

```
In [90]: %time a = my_arr * 2
```

CPU times: user 5.23 ms, sys: 1.84 ms, total: 7.07 ms
Wall time: 4.58 ms

```
In [91]: b = my_list * 2
```

```
-----  
TypeError                                Traceback (most recent call last)  
Cell In[91], line 1  
----> 1 b = my_list * 2
```

TypeError: unsupported operand type(s) for *: 'range' and 'int'

```
In [92]: %time b = [x*2 for x in my_list]
```

CPU times: user 96.4 ms, sys: 24.2 ms, total: 121 ms
Wall time: 120 ms

模块的导入

- Python中可以使用以下三种方式导入模块或模块中的函数：
 1. import 模块名
 2. import 模块名 as 新名称
 3. from 模块名 import 函数名

例子

```
In [93]: math.cos(1)
```

```
-----  
NameError                                Traceback (most recent call last)  
Cell In[93], line 1  
----> 1 math.cos(1)
```

```
NameError: name 'math' is not defined
```

```
In [94]: import math  
  
math.cos(1)
```

Out[94]:

0.5403023058681398

```
In [95]: import math as shuxue  
  
shuxue.cos(1)
```

Out[95]:

0.5403023058681398

```
In [96]: cos(1)
```

NameError

Traceback (most recent call last)

Cell In[96], line 1

----> 1 cos(1)

NameError: name 'cos' is not defined

```
In [97]: from math import cos
```

```
cos(1)
```

Out[97]:

0.5403023058681398

谢谢