第二章

1. 字符串

（1）大小写

name.title() 字符串每个单词首字母都大写

name.upper() 所有字母都大写

name.lower() 所有字母都小写

1. 字符串合并 用+
2. 换行符 \n

制表符 \t

二者可以连续使用 \n\t

1. 删除空白

str.strip()两端

str.lstrip()左端

Str.rstrip()右端

1. 数字

Python支持直接运算（运算次序），空格不影响运算

用str()转换类型

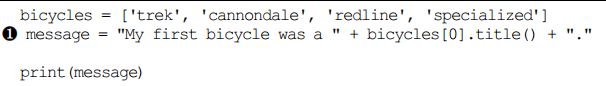
1. 用#注释

第三章

1. 列表简介

用方括号[] 来表示列表，并用逗号来分隔其中的元素

1. 访问列表元素同数组形式
2. 可以使用列表中的值参与输出



1. 修改列表元素：直接修改
2. 向列表中添加新元素

在列表末添加：a.append(‘b’)

在列表任何位置添加：a.insert(0.’b’) （在某个位置插入，后面的元素依次右移）

1. 删除元素

正常删除：del a[0]

删除并使用：a.pop() （默认为弹出最后一个）

a.pop(0) （按位置弹出）

根据值删除：a.remove(‘b’)

1. 组织列表
2. 按首字母永久性排序：a.sort()
3. 在输出的时候临时排序：sorted(a)
4. 倒序打印（永久）：a.reverse()
5. 返回列表长度：len(a)
6. 操作列表
7. 遍历

for a in as:

print(a)

注：for循环只循环缩进的print语句，没有缩进的则不在for循环之内

1. 创建数值列表
2. 函数range()：打印一系列数字

for a in range(1,5)

print(a)

只能打印出1-4

1. 利用range()创建数字列表

a=list(range(1,5))

list()用于生成列表

指定步长：range(2,11,2) 输出2,4,6,8,10

1. 数字列表的简单统计计算

最小值min(a)

最大值max(a)

和sum(a)

1. 列表解析

把for循环和创建新元素放在一行

1. 使用列表的一部分
2. 切片

打印出第二个到第四个元素（相当于a[1]-a[3]） a[1:4]

打印出第二个到元素结束 a[1:]

输出最后三个元素 a[-3:]

1. 遍历切片

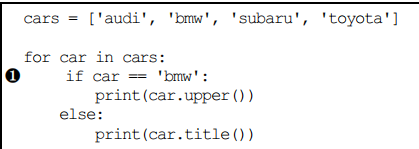
for a in as[3:]

print

1. 复制列表

b列表复制a的内容：b=a[:]

1. 元祖
2. 定义元祖（用圆括号） a=(200,50)
3. 遍历元祖：同列表使用for循环
4. 修改元祖变量：通过重新赋值形式
5. if语句



else后面要有：

1. 条件测试
2. 检查相等 ==

判断相等时不包括大小写

1. 检查不相等 ！=
2. 多个检查条件：用and和or
3. 检查特定值包含在列表中：‘b’ in a
4. 检查特定值不包含在列表中：‘b’ not in a
5. If语句
6. if else语句 （注意else后面有：）
7. If-elif-esle语句
8. if语句处理列表
9. 确认列表不为空 if a:
10. 字典
11. 使用字典

键值对应：alien\_0 = {'color': 'green', 'points': 5}

（大括号）

1. 访问字典中的值：alien\_0['color']
2. 添加键值对：alien\_0['x\_position'] = 0 （直接赋值）
3. 修改字典中的值：直接修改
4. 删除键值对：del alien\_0['points']
5. 遍历字典
6. 遍历所有键值对：for k, v in user\_0.items():
7. 遍历所有键：for name in favorite\_languages.keys():
8. 按顺序遍历所有键：for name in sorted(favorite\_languages.keys()):
9. 遍历所有值：for language in favorite\_languages.values()

去除重复项：for language in set(favorite\_languages.values()):

1. 嵌套
2. 字典列表

alien\_0 = {'color': 'green', 'points': 5}

alien\_1 = {'color': 'yellow', 'points': 10}

alien\_2 = {'color': 'red', 'points': 15}

aliens = [alien\_0, alien\_1, alien\_2]

1. 在字典中存储列表
2. 在字典中储存字典

第七章 用户输入和while循环

函数input()

input("Tell me something, and I will repeat it back to you: ") 提示输入的内容

在输入数字的时候，如果需要比较大小，则需要手动转换int类型

求模运算符：%

1. while循环
2. 使用标志

用True和False简化while循环

1. 用break结束循环
2. 用continue继续循环
3. 用while循环处理列表和字典
4. 在列表中移动元素：用pop
5. 删除包含特定值的所有列表元素：while特定值，remote
6. 使用用户输入来填充字典

第八章 函数

1.定义函数

def greet\_user():后面加上缩进的内容就构成函数

1. 向函数传递信息：（）内加入参数
2. 传递实参
3. 位置实参：输入必须按顺序
4. 关键字实参：指定位置输入，不需要顺序
5. 默认值：可以给参数设定默认自
6. 函数等效调用：
7. 返回值
8. 让实参变成可选的：用if
9. 返回字典
10. 结合使用函数和while循环
11. 传递列表
12. 在函数中修改列表
13. 禁止函数修改列表
14. 传递任意数量的实参

def make\_pizza(\*toppings):

1. 结合使用位置实参和任何数量实参：def make\_pizza(size, \*toppings):
2. 使用任意数量的关键字实参：def build\_profile(first, last, \*\*user\_info):

两个\*创建字典

1. 导入整个模块

Import文件名

1. 导入特定的函数：from module\_name import function\_name
2. 使用as给函数指定别名：可以省略一部分函数名
3. 导入模块中的所有函数；from pizza import \*

第九章 类

1.创建和使用类

❶ class Dog():

❷ """一次模拟小狗的简单尝试"""

❸ def \_\_init\_\_(self, name, age):

"""初始化属性name和age"""

❹ self.name = name

self.age = age

❺ def sit(self):

"""模拟小狗被命令时蹲下"""

print(self.name.title() + " is now sitting.")

def roll\_over(self):

"""模拟小狗被命令时打滚"""

print(self.name.title() + " rolled over!")

1. 访问属性：.name的方式
2. 调用方式：.的方式
3. 使用类的实例

（1）修改属性值的方法：直接修改和通过方法修改

1. 继承
2. 继承方式：class ElectricCar(Car):

父类在括号内，并且在文件中父类要在子类前出现

1. 子类的方法\_init\_()继承父类
2. 重写父类的方法：def加方法名称
3. 导入类：from import

如果从同一模块导入多个类，则用逗号隔开

导入整个模块：直接import

导入模块中的所有类：from module\_name import \*

1. 文件与异常
2. 从文件中读取数据

with open('pi\_digits.txt') as file\_object:

contents = file\_object.read()

print(contents)

（1）文件相对路径：with open('text\_files/filename.txt') as file\_object:

（2）文件绝对路径：

file\_path = 'C:\Users\ehmatthes\other\_files\text\_files\filename.txt'

with open(file\_path) as file\_object:

（2）逐行读取：

filename = 'pi\_digits.txt'

with open(filename) as file\_object:

for line in file\_object:

print(line)

1. 创建一个包含文件各行内容的列表

filename = 'pi\_digits.txt'

with open(filename) as file\_object:

lines = file\_object.readlines()

for line in lines:

print(line.rstrip())

1. 使用文件的内容
2. 包含一万位的大型文件：
3. 写入文件
4. 写入空文件

filename = 'programming.txt'

❶ with open(filename, 'w') as file\_object:

❷ file\_object.write("I love programming.")

读取模式：r 写入模式：w 附加模式：a

附加到文件用a

1. 异常
2. try-except
3. else代码块
4. 分析文本：split()
5. 使用多个文件
6. 存储数据
7. 使用json.dump() 和json.load()

函数json.dump() 接受两个实参：要存储的数据以及可用于存储数据的文件对象