python in action

Lesson 3 - 容器数据类型，写第一个程序

凤凰山 github.com/gxtrobot/pyinaction

2019-10

Lesson3 - 容器数据类型，写第一个程序

课程内容

今天的课程将介绍 Python 的容器数据类型 (list, tuple, set, dict)，以及相关的操作. 了解 sys.argv 以及如何传递参数给程序，运用已学到的知识写第一个程序

课程目标

- 了解并使用 Python 的容器数据类型
- 了解每种容器数据类型的特点
- 知道每种容器数据类型的基本操作，如果添加数据，获取数据等
- 了解 sys.argv 的用法
- 编写一个小程序来接收并打印参数

容器（复合）数据类型

就是存放其他类型数据的类型，其元素可以为基本或其他容器类型，或者自定义的对象

list(列表))

- 可以存放连续任意个值，类型不限，数据按添加顺序排列
- 使用 [] 创建，如 `[1,2,3]`
- 也可调用 list()，如 `list([1,2,3])`
- 操作（以下用 nums 指定一个 list，值为 [1,2,3,4,5]），长度 length 为 5
  - 索引 ([i])，0<=i<=length-1，如 `nums[0]` 为 `1`
  - 切片 ([i:j])，`nums[1:3]` 为 `[2,3]`
  - 求长度 (len(nums))，`len(nums)` 为 `5`
  - 拼接两个 list(+)，`[1,2,3] + [2,3]` 为 `[1,2,3,2,3]`

容器（复合）数据类型

tuple(元组)

- 类似 list，可以存放任意各值，类型不限，但创建后不可修改，数据按添加顺序排练
- 使用 () 创建，如 (1,2,3)
- 使用 tuple() 创建
- 操作 (类似 list)
  - 索引
  - 切片
  - 求长度
  - 拼接

容器（复合）数据类型

set(集合)

- 可以存放任意个不同的数据（每个数据必须不同），没有顺序
- 使用 {} 创建
- 使用 set({})
- 操作 (s={1,2,3})
  - 获取一个随机数据 (pop)，`s.pop()`
  - 求长度 (len)
  - 测试数据是否存在，使用 in 操作符，`1 in s` 为 `True`
  - 求交集 (&)，`s & {1,2}` 为 `{1,2}`
  - 求并集 (|)，`s | {1,3,4}` 为 `{1,2,3,4}`

容器（复合）数据类型

dict(字典)

- 可以存放人一个键值对，每个键必须不同，值可以相同，没有顺序
- 使用 {key:value} 创建，`{'name':'jack', 'age':30}`
- 使用 dict() 创建
- 操作 `d={'name':'jack', 'age':30}`
  - 按键取值 [key]，`d['name']` 为 `'jack'`
  - 求键值对个数 (`len(d)` 为 `2`)
  - 添加新键值对，`d['job']='driver'`
  - 测试键是否存在，使用 in 操作符，`'age' in d` 为 `True`

IDLE 玩一玩

在 IDLE 试着用每种数据类型，以及相应操作符进行计算，并观察结果

```
nums = [1,2,3,4,5]
nums[0]
nums[1] = 10
nums + [1,2]
len(nums)
nums[2:4]

t = (1,2,3,4)
t[0]
t[1] = 10
len(t)
t + (5,6)
```

IDLE 玩一玩

```
s = {1,2,3,4}
s.pop()
s & {1,2,5}
s | {1,2,5}

d = {'name': 'jack', 'age': 30}
d['name']
d['job'] = 'driver'
'age' in d
```

引入一个模块 (import)

- 使用 import 可以引入一个模块，可以是系统模块或自定义模块
- 也可以使用 from ... import ...，从某个模块引入具体的对象

```
import sys
sys.argv
from sys import argv
argv
```

sys.argv 的使用

- sys.argv 记录了程序调用时候收到的所有参数，是一个 list
- 第一个值 (索引 0) 是程序本身，后面是所有参数
- 每个值都为字符串 (str)，需要自行转换将以下代码存为 argv.py

```
import sys
print('所有参数', sys.argv)
print('参数个数', len(sys.argv))
print('实际参数', sys.argv[1:])
```

数据类型转换

- int(), 可以将其他类型转为 int 型, 无法转换会报错
- float(), 可以将其他类型转为 float 型, 无法转换会报错
- str(), 可以将其他类型转为字符串型、
- bool(), 可以将其他类型转为布尔型

```
int('100')
float('20.5')
int(1.4)
str(100)
bool(1)
```

课后练习

- 创建 argv.py，并尝试传递不同参数，查看结果
- 修改 argv.py，尝试计算所有传递参数之和，假设所有参数都为整数

问题