《Python 编程：从入门到实践》第 3 章 列表简介 · 语雀

登录 / 注册

3.1 列表是什么

列表（list）是 Python 中最常用的数据结构之一，它可以存储一组有序的元素，这些元素可以是不同类型的。

```
fruits = ['apple', 'banana', 'cherry']
numbers = [1, 2, 3, 4, 5]
mixed = ['hello', 42, True, 3.14]
```

要点： 列表 = 有序 + 可变 + 允许重复元素

3.2 访问列表元素

通过索引（index）访问列表中的单个元素，索引从 0 开始。

```
fruits = ['apple', 'banana', 'cherry']
print(fruits[0])   # apple
print(fruits[1])   # banana
print(fruits[-1])  # cherry（倒数第一个）
print(fruits[-2])  # banana（倒数第二个）
```

常见错误： 索引超出范围会报 `IndexError`

```
print(fruits[3])  # IndexError: list index out of range
```

3.3 修改、添加和删除元素

修改元素

```
fruits = ['apple', 'banana', 'cherry']
fruits[1] = 'blueberry'
print(fruits)  # ['apple', 'blueberry', 'cherry']
```

添加元素

append()：在末尾添加

```
fruits = ['apple', 'banana']
fruits.append('cherry')
print(fruits)  # ['apple', 'banana', 'cherry']
```

insert()：在指定位置插入

```
fruits = ['apple', 'cherry']
fruits.insert(1, 'banana')
print(fruits)  # ['apple', 'banana', 'cherry']
```

删除元素

del：按索引删除

```
fruits = ['apple', 'banana', 'cherry']
del fruits[1]
print(fruits)  # ['apple', 'cherry']
```

pop()：删除并返回末尾元素

```
fruits = ['apple', 'banana', 'cherry']
last = fruits.pop()
print(last)    # cherry
print(fruits)  # ['apple', 'banana']
```

pop(i)：删除并返回指定位置元素

```
fruits = ['apple', 'banana', 'cherry']
second = fruits.pop(1)
print(second)  # banana
print(fruits)  # ['apple', 'cherry']
```

remove()：按值删除

```
fruits = ['apple', 'banana', 'cherry']
fruits.remove('banana')
print(fruits)  # ['apple', 'cherry']
```

注意： remove() 只删除第一个匹配的值，如果值不存在会报 `ValueError`

3.4 组织列表

sort()：永久排序

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6]
numbers.sort()
print(numbers)  # [1, 1, 2, 3, 4, 5, 6, 9]

numbers.sort(reverse=True)
print(numbers)  # [9, 6, 5, 4, 3, 2, 1, 1]
```

sorted()：临时排序

```
numbers = [3, 1, 4, 1, 5, 9, 2, 6]
print(sorted(numbers))  # [1, 1, 2, 3, 4, 5, 6, 9]
print(numbers)          # [3, 1, 4, 1, 5, 9, 2, 6]（原列表不变）
```

reverse()：反转顺序

```
fruits = ['apple', 'banana', 'cherry']
fruits.reverse()
print(fruits)  # ['cherry', 'banana', 'apple']
```

len()：获取长度

```
fruits = ['apple', 'banana', 'cherry']
print(len(fruits))  # 3
```

3.5 遍历列表

使用 for 循环遍历列表中的每个元素：

```
fruits = ['apple', 'banana', 'cherry']
for fruit in fruits:
    print(fruit)
```

输出：

```
apple
banana
cherry
```

缩进很重要！ for 循环后面的代码块必须缩进（通常 4 个空格）。

3.6 创建数值列表

range()：生成数字序列

```
for i in range(1, 6):
    print(i)  # 1 2 3 4 5
```

注意： range(1, 6) 不包含 6

列表解析（List Comprehension）

```
squares = [x**2 for x in range(1, 11)]
print(squares)  # [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

统计计算

```
numbers = [1, 2, 3, 4, 5]
print(min(numbers))  # 1
print(max(numbers))  # 5
print(sum(numbers))  # 15
```

3.7 切片

切片用于获取列表的一部分，语法：`list[start:end]`

```
fruits = ['apple', 'banana', 'cherry', 'date', 'elderberry']
print(fruits[1:3])   # ['banana', 'cherry']
print(fruits[:3])    # ['apple', 'banana', 'cherry']
print(fruits[2:])    # ['cherry', 'date', 'elderberry']
print(fruits[-3:])   # ['cherry', 'date', 'elderberry']
```

复制列表

```
fruits = ['apple', 'banana', 'cherry']
copy = fruits[:]
copy.append('date')
print(fruits)  # ['apple', 'banana', 'cherry']（原列表不变）
print(copy)    # ['apple', 'banana', 'cherry', 'date']
```

3.8 元组

元组（tuple）与列表类似，但**不可修改**，使用圆括号 `()`。

```
dimensions = (200, 50)
print(dimensions[0])  # 200
print(dimensions[1])  # 50
```

尝试修改元组会报错：

```
dimensions = (200, 50)
dimensions[0] = 250  # TypeError: 'tuple' object does not support item assignment
```

遍历元组

```
dimensions = (200, 50)
for dimension in dimensions:
    print(dimension)
```

修改元组变量

虽然不能修改元组的元素，但可以给变量重新赋值：

```
dimensions = (200, 50)
print("Original dimensions:")
for dimension in dimensions:
    print(dimension)

dimensions = (400, 100)
print("\nModified dimensions:")
for dimension in dimensions:
    print(dimension)
```

3.9 本章小结

- 列表是有序、可变、允许重复的数据结构
- 通过索引访问元素，索引从 0 开始，支持负索引
- 使用 append()、insert()、del、pop()、remove() 修改列表
- sort() 永久排序，sorted() 临时排序，reverse() 反转
- 使用 for 循环遍历列表
- range() 生成数字序列，列表解析可以简洁地创建列表
- 切片 `list[start:end]` 获取子列表，`list[:]` 复制列表
- 元组不可修改，适合存储不应改变的数据

练习题

1. 创建一个包含你最喜欢的 5 部电影的列表，然后按字母顺序打印。
2. 创建一个列表，包含 1 到 20 的所有奇数，然后用 for 循环打印。
3. 创建一个列表，包含 10 个随机数（提示：使用 random 模块），然后计算它们的平均值。
4. 使用列表解析创建一个包含 1 到 100 中所有能被 3 整除的数的列表。
5. 创建一个元组，包含一周七天的名称，然后打印出来。

参考资料

- Python 官方文档：https://docs.python.org/3/tutorial/datastructures.html
- 《Python 编程：从入门到实践》第 3 章