

Day7-Python 元组与集合

一、学习目标

理解元组 (Tuple) 与集合 (Set) 的核心概念与区别，掌握元组的创建、索引访问和切片，掌握集合的创建、元素增删与集合运算（并/交/差），能用集合去重，完成基础数据清洗。

二、知识点讲解

1. 元组 (Tuple)

- 有序
- 不可变（创建后不能修改）
- 使用 () 创建
- 可索引访问和切片

```
# 创建一个包含 5 个整数的元组；元组是有序且不可变的数据结构
my_tuple = (10, 20, 30, 40, 50)
# 打印整个元组，直观查看其所有元素
print("元组:", my_tuple) # 元组: (10, 20, 30, 40, 50)
# 通过正向索引访问第一个元素（索引从 0 开始）
print("第一个元素:", my_tuple[0]) # 第一个元素: 10
# 通过负向索引访问最后一个元素（-1 表示倒数第一个）
print("最后一个元素:", my_tuple[-1]) # 最后一个元素: 50
# 使用切片获取从索引 1 到 3 的元素（左闭右开，不包含索引 4）
print("索引 1 到 3 的元素:", my_tuple[1:4]) # 索引 1 到 3 的元素: (20, 30, 40)
```

2. 集合 (Set)

- 无序
- 可变

- 元素唯一（自动去重）
- 用 {} 或 set() 创建
- 支持增删、集合运算

```
# 使用花括号创建一个包含三种水果的集合；集合无序、元素唯一
my_set = {"苹果", "香蕉", "橙子"}
# 打印原集合；注意输出顺序可能与插入顺序不同（无序）
print("原集合:", my_set) # 原集合：{'苹果', '香蕉', '橙子'}
# 使用 add 方法添加一个新元素；若已存在则不会重复添加
my_set.add("葡萄")
print("添加葡萄:", my_set) # 添加葡萄：{'苹果', '香蕉', '橙子', '葡萄'}
# 使用 remove 精确删除一个已存在的元素；如果不存在会抛出 KeyError
my_set.remove("香蕉")
print("删除香蕉:", my_set) # 删除香蕉：{'苹果', '橙子', '葡萄'}
# 定义两个整数集合以演示集合运算
set_a = {1, 2, 3}
set_b = {3, 4, 5}
# 并集：属于 set_a 或 set_b 的所有元素（去重合并）
print("并集:", set_a | set_b) # 并集：{1, 2, 3, 4, 5}
# 交集：同时属于 set_a 与 set_b 的元素
print("交集:", set_a & set_b) # 交集：{3}
# 差集：属于 set_a 但不属于 set_b 的元素（方向性：a - b 与 b - a 不同）
print("差集:", set_a - set_b) # 差集：{1, 2}
```

三、 作业安排

任务描述：

1. 创建一个元组并尝试访问其中的元素。
2. 用集合去掉列表 ['红', '蓝', '红', '黄'] 中重复的元素，并转换为列表输出。

```
# 任务 1：定义一个包含三种编程语言的元组
```

```
task_tuple = ("Python", "Java", "C++")
# 打印整个元组，确认元素
print("元组:", task_tuple)
# 访问第一个元素（索引 0）
print("第一个元素:", task_tuple[0])
# 访问最后一个元素（索引 -1）
print("最后一个元素:", task_tuple[-1])

# 任务 2：准备一个包含重复颜色的列表
colors = ["红", "蓝", "红", "黄"]
# 将列表转换为集合，自动去重（集合无序）
unique_colors = set(colors)
# 打印原列表与去重后的集合，便于对比
print("原列表:", colors)
print("去重后的集合:", unique_colors)
# 将集合再转换回列表（如果需要稳定顺序，可使用 sorted(unique_colors)）
unique_colors_list = list(unique_colors)
# 打印最终的去重列表
print("去重后的列表:", unique_colors_list)
```

四、总结

元组：有序、不可变，适合表示固定结构的数据（如经纬度、RGB）。集合：无序、元素唯一，支持并/交/差等运算，常用于去重与集合逻辑计算。集合去重后若需顺序，可再转回列表并按需要排序。