

G4-06 课后练习 —— 复合嵌套下

姓名： ____ 日期： ____

面向7-10岁少儿 | G4 编程高手

一、填空题（回顾概念）

题目1：字典套字典的结构「data = {"a": {"x": 1, "y": 2}}」中，访问值 1 的写法是「data[____][____]」。

题目2：下面代码中，「[row[0] for row in matrix]」的作用是取出矩阵的 ____ 列。

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

二、代码补全（补写代码）

题目3：

学校有多个班级，每个班有多个学生，每个学生有多科成绩（三层嵌套）。请补全代码，计算每个班的平均分。

```
school = {  
    "一班": {  
        "小明": {"语文": 90, "数学": 85},  
        "小红": {"语文": 95, "数学": 92}  
    },  
    "二班": {  
        "小刚": {"语文": 78, "数学": 88},  
        "小丽": {"语文": 88, "数学": 90}  
    }  
}
```

```
for class_name, students in school.___():  
    total_score = 0  
    count = 0  
    for student, scores in students.___():  
        for subject, score in scores.___():  
            total_score = total_score + ____
```

```
count = count + ____
avg = total_score / count
print(f"{class_name}平均分: {avg:.1f}")
```

题目4： 给定一个嵌套字典（省份城市区）， 用双层循环打印所有区名称。补全代码。

```
areas = {
    "北京市": {"海淀区": ["中关村", "五道口"], "朝阳区": ["国贸", "望京"]},
    "上海市": {"浦东新区": ["陆家嘴", "张江"], "静安区": ["南京西路"]}
}
```

```
# 打印所有区
for city, districts in areas.___():
    print(f"\n{city}的区: ")
    for district, places in districts.___():
        print(f" - {district}: {' '.___(places)}")
```

三、创作题（动手编程）

题目5： 编写"图书管理系统"（字典套列表套字典的多层嵌套）。

- 数据结构：

```
```python
library = {
 "文学类": [
 {"title": "西游记", "author": "吴承恩", "year": 1592},
 {"title": "小王子", "author": "圣埃克苏佩里", "year": 1943}
],
 "科普类": [
 {"title": "十万个为什么", "author": "韩启德", "year": 1961},
 {"title": "昆虫记", "author": "法布尔", "year": 1879}
]
}
```
```

- 实现以下功能：

1. 遍历所有分类，打印每本书的信息
2. 统计图书馆共有多少本书
3. 找到出版年份最早的书
4. 添加一本新书到"文学类"：《三国演义》，作者罗贯中，出版年份1522

请写出完整代码。

<details>

<summary>参考答案</summary>

题目1： [data["a"]["x"]]

题目2： 第一列（索引0的列）

题目3：

```
school = {
    "一班": {
        "小明": {"语文": 90, "数学": 85},
        "小红": {"语文": 95, "数学": 92}
    },
    "二班": {
        "小刚": {"语文": 78, "数学": 88},
        "小丽": {"语文": 88, "数学": 90}
    }
}
```

```
for class_name, students in school.items():
    total_score = 0
    count = 0
    for student, scores in students.items():
        for subject, score in scores.items():
            total_score = total_score + score
            count = count + 1
    avg = total_score / count
    print(f"{class_name}平均分: {avg:.1f}")
```

题目4：

```
areas = {
    "北京市": {"海淀区": ["中关村", "五道口"], "朝阳区": ["国贸", "望京"]},
    "上海市": {"浦东新区": ["陆家嘴", "张江"], "静安区": ["南京西路"]}
}
```

```
for city, districts in areas.items():
    print(f"\n{city}的区: ")
    for district, places in districts.items():
        print(f" - {district}: {' '.join(places)}")
```

题目5：

```
library = {
    "文学类": [
        {"title": "西游记", "author": "吴承恩", "year": 1592},
        {"title": "小王子", "author": "圣埃克苏佩里", "year": 1943}
    ],
    "科普类": [
```

```
{ "title": "十万个为什么", "author": "韩启德", "year": 1961},  
  { "title": "昆虫记", "author": "法布尔", "year": 1879}  
]  
}
```

1. 遍历所有分类

```
print("==== 图书馆藏书 ====")  
for category, books in library.items():  
    print(f"\n【{category}】")  
    for book in books:  
        print(f"  《{book['title']}》 - {book['author']} ({book['year']}年)")
```

2. 统计总数

```
total = sum(len(books) for books in library.values())  
print(f"\n共计: {total}本书")
```

3. 最早出版的书

```
all_books = [book for books in library.values() for book in books]  
earliest = min(all_books, key=lambda b: b["year"])  
print(f"最早出版: 《{earliest['title']}》 ({earliest['year']}年) ")
```

4. 添加新书

```
library["文学类"].append({ "title": "三国演义", "author": "罗贯中", "year": 1522})  
print(f"\n已添加《三国演义》到文学类")  
print(f"文学类现有{len(library['文学类'])}本书")
```

</details>

自我评价: ☐ 全都会了 ☐ 大部分会了 ☐ 还要再练练