

G4-19 课后练习 —— 综合项目三

姓名： ____ 日期： ____

面向7-10岁少儿 | G4 编程高手 | GESP四级综合复习

一、填空题（回顾概念）

题目1： 列表推导式 `[x**2 for x in range(1, 6) if x % 2 == 0]` 的结果是 ____ 。这行代码等价于用 `for` 循环筛选 ____ 数再平方。

题目2： `sorted([(1, "b"), (3, "a"), (2, "c")], key=lambda x: x[1])` 的结果是 ____ 。这里 `key=lambda x: x[1]` 表示按每个元组的第 ____ 个元素排序。

二、代码补全（补写代码）

题目3： 编写一个"通讯录"程序，数据持久化到文件。补全代码。

```
import json
```

```
contacts = []
```

```
def save_contacts():
    """保存联系人到文件"""
    with open("contacts.json", "___", encoding="utf-8") as f:
        json.__(contacts, f, ensure_ascii=False, indent=2)
    print("通讯录已保存!")
```

```
def load_contacts():
    """从文件加载联系人"""
    global contacts
    try:
        with open("contacts.json", "___", encoding="utf-8") as f:
            contacts = json.__(f)
        print(f"已加载{len(contacts)}个联系人")
    except FileNotFoundError:
        print("没有找到通讯录文件，将创建新的")
```

```
contacts = []
```

```
def add_contact(name, phone):
    contacts.append({"name": name, "phone": phone})
    save_contacts()
```

```
# 加载已有数据
load_contacts()
add_contact("小明", "13800001111")
print(f"当前联系人: {contacts}")
```

题目4： 用递归函数实现"二分查找"算法。补全代码。

```
def binary_search(arr, target, left, right):
    """
    在有序列表 arr 的 [left, right] 范围内查找 target
    返回目标索引，未找到返回 -1
    """
    if left >= right:
        return -1
```

```
    mid = (left + right) // 2
```

```
    if arr[mid] == target:
        return mid
    elif arr[mid] > target:
        return binary_search(arr, target, left, mid - 1)
    else:
        return binary_search(arr, target, mid + 1, right)
```

```
# 测试
nums = [1, 3, 5, 7, 9, 11, 13, 15]
index = binary_search(nums, 7, 0, len(nums) - 1)
print(f"数字7在索引: {index}") # 期待输出: 3
```

三、创作题（动手编程）

题目5： 综合项目：编写"迷你成绩分析系统"。

数据文件「scores.csv」 内容（手动创建或用代码生成）：

```
姓名,语文,数学,英语
小明,90,85,88
小红,95,92,90
小刚,78,88,82
小丽,92,96,94
小华,65,72,70
```

要求实现以下功能（用函数封装）：

- 「read_scores(filename)」：读取 CSV 文件，返回列表套字典的数据结构
- 「calc_stats(students)」：计算每位学生的总分、平均分、等级（90以上优秀，80以上良好，60以上及格，否则不及格），添加进字典
- 「find_top(students, subject)」：找出指定科目分数最高的同学
- 「class_report(students)」：生成班级报告，含各科平均分、最高分、最低分
- 「save_report(students, filename)」：将分析结果保存到 JSON 文件

请写出完整代码。

<details>

<summary>参考答案</summary>

题目1： [4, 16]，偶

题目2： [(3, 'a'), (1, 'b'), (2, 'c')], 1（第2个元素，从0开始计数）

题目3：

```
import json
```

```
contacts = []
```

```
def save_contacts():  
    with open("contacts.json", "w", encoding="utf-8") as f:  
        json.dump(contacts, f, ensure_ascii=False, indent=2)  
    print("通讯录已保存！")
```

```
def load_contacts():  
    global contacts  
    try:  
        with open("contacts.json", "r", encoding="utf-8") as f:  
            contacts = json.load(f)  
        print(f"已加载{len(contacts)}个联系人")  
    except FileNotFoundError:  
        print("没有找到通讯录文件，将创建新的")  
        contacts = []
```

```
def add_contact(name, phone):  
    contacts.append({"name": name, "phone": phone})  
    save_contacts()
```

```
load_contacts()  
add_contact("小明", "13800001111")  
print(f"当前联系人: {contacts}")
```

题目4：

```
def binary_search(arr, target, left, right):
    if left > right:
        return -1
```

```
mid = (left + right) // 2
```

```
if arr[mid] == target:
    return mid
elif arr[mid] > target:
    return binary_search(arr, target, left, mid - 1)
else:
    return binary_search(arr, target, mid + 1, right)
```

```
nums = [1, 3, 5, 7, 9, 11, 13, 15]
index = binary_search(nums, 7, 0, len(nums) - 1)
print(f"数字7在索引: {index}")
```

题目5:

```
import json
import csv
```

```
# 先生成数据文件
def create_sample_data():
    data = """姓名,语文,数学,英语
小明,90,85,88
小红,95,92,90
小刚,78,88,82
小丽,92,96,94
小华,65,72,70"""
    with open("scores.csv", "w", encoding="utf-8") as f:
        f.write(data)
```

```
# 1. 读取CSV
def read_scores(filename):
    students = []
    with open(filename, "r", encoding="utf-8") as f:
        reader = csv.DictReader(f)
        for row in reader:
            row["语文"] = int(row["语文"])
            row["数学"] = int(row["数学"])
            row["英语"] = int(row["英语"])
            students.append(row)
    return students
```

```
# 2. 计算统计
def calc_stats(students):
    for stu in students:
        stu["总分"] = stu["语文"] + stu["数学"] + stu["英语"]
        stu["平均分"] = round(stu["总分"] / 3, 1)
        avg = stu["平均分"]
        if avg >= 90:
            stu["等级"] = "优秀"
```

```
elif avg >= 80:
    stu["等级"] = "良好"
elif avg >= 60:
    stu["等级"] = "及格"
else:
    stu["等级"] = "不及格"
return students
```

3. 找单科最高

```
def find_top(students, subject):
    top = max(students, key=lambda s: s[subject])
    return top
```

4. 班级报告

```
def class_report(students):
    subjects = ["语文", "数学", "英语"]
    report = {"各科统计": {}}
    for sub in subjects:
        scores = [s[sub] for s in students]
        report["各科统计"][sub] = {
            "平均分": round(sum(scores) / len(scores), 1),
            "最高分": max(scores),
            "最低分": min(scores)
        }
    report["总人数"] = len(students)
    report["优秀人数"] = sum(1 for s in students if s["等级"] == "优秀")
    report["不及格人数"] = sum(1 for s in students if s["等级"] == "不及格")
    return report
```

5. 保存报告

```
def save_report(report, filename):
    with open(filename, "w", encoding="utf-8") as f:
        json.dump(report, f, ensure_ascii=False, indent=2)
    print(f"报告已保存至 {filename}")
```

---- 主程序 ----

```
create_sample_data()
students = read_scores("scores.csv")
students = calc_stats(students)
```

```
print("==== 学生成绩 =====")
for s in students:
    print(f"{s['姓名']}: 总分{s['总分']} 平均{s['平均分']} 等级{s['等级']}")
```

```
top_math = find_top(students, "数学")
print(f"\n数学最高: {top_math['姓名']} ({top_math['数学']}分) ")
```

```
report = class_report(students)
print(f"\n班级人数: {report['总人数']}")
for sub, stats in report["各科统计"].items():
    print(f"{sub}: 平均{stats['平均分']} 最高{stats['最高分']} 最低{stats['最低分']}")
```

```
save_report(report, "class_report.json")
```

</details>



自我评价： ☐ 全都会了 ☐ 大部分会了 ☐ 还要再练练