Python实验教学案例：手机号码验证程序

实验目的

1. 掌握Python字符串基本操
2. 理解程序流程控制（特别是if选择结构）
3. 熟悉IDLE调试环境的使用
4. 培养严谨的逻辑思维能力

实验环境

- Python 3.x
- IDLE集成开发环境
- Windows/macOS操作系统

实验步骤

阶段一：问题分析

知识点：中国手机号规则

- 11位数字
- 第1位固定为1
- 第2位：3/4/5/7/8/9
- 第3-11位：任意数字

阶段二：手工验证

任务：编写手工验证流程

```
phone = "13812345678"
if len(phone) == 11 \         # 注意转义字符\
    and phone.isdigit() \     # isdigit()方法，判断是否全为数字
    and phone[0] == "1" \     # 1为何加引号
    and phone[1] in "345789": # 为何in
    print("有效号码")
else:
    print("无效号码")
```

阶段三：测试和调试（20分钟）

IDLE调试技巧：

1. 按F5运行程序
2. 在代码行号处右键设置断点
3. 使用调试控制按钮：
    - Go：执行到断点
    - Step：单步执行
    - Over：跳过函数
    - Out：跳出函数

基础实验任务

实现基本验证功能：

1. 长度验证
2. 数字验证
3. 号段验证

完整代码

```python
def validate_phone(phone):
    """
    手机号码验证函数
    参数：phone (str) - 待验证的号码
    返回：验证结果 (bool)
    """
    # 预处理：去除前后空格
    phone = phone.strip()

    # 基础验证（使用if选择结构）
    if len(phone) != 11:          # 长度检查
        return False
    if not phone.isdigit():       # 纯数字检查
        return False
    if phone[0] != '1':          # 首位检查
        return False
    if phone[1] not in '3456789':  # 第二位检查
        return False

    # 全部通过
    return True

# 主程序
if __name__ == "__main__":
    # 获取用户输入
    input_phone = input("请输入手机号码：")

    # 调用验证函数
    if validate_phone(input_phone):
        print("✅ 有效手机号码")
    else:
        print("❌ 无效手机号码")
```

代码逐行解析

1. `def validate_phone(phone):`：定义验证函数
2. `phone = phone.strip()`：去除输入字符串首尾空格
3. `len(phone) != 11`：长度验证条件判断
4. `phone.isdigit()`：检查是否全为数字
5. `phone[0] != '1'`：验证首位是否为1
6. `phone[1] not in '3456789'`：验证第二位是否合法

7. `return True`：通过验证
8. `if __name__ == "__main__"`：主程序入口
9. `input_phone = input("请输入手机号码：")`：获取用户输入
10. `if validate_phone(input_phone):`：调用验证函数
11. `print("✅ 有效手机号码")`：输出验证结果

调试技巧

1. 使用IDLE的调试功能
2. 逐行执行代码
3. 观察变量变化
4. 检查条件是否正确

重点知识说明

if选择结构：

- 使用多个条件判断逐步过滤无效号码
- 注意条件顺序：先进行快速失败的检查
- 使用逻辑运算符组合条件

程序运行逻辑

```
开始
↓
输入号码 → 预处理
↓
长度检查 → 失败 → 返回False
↓
数字检查 → 失败 → 返回False
↓
首位检查 → 失败 → 返回False
↓
第二位检查 → 失败 → 返回False
↓
正则表达式验证 → 返回最终结果
结束
```

阶段四：扩展任务

扩展任务1：永久循环

任务：

实现一个永久循环，让用户可以不断输入号码进行验证，直到输入"exit"退出。
提示：使用while循环和break语句

代码示例

```
# 主程序
if __name__ == "__main__":
    while True:
        input_phone = input("请输入手机号码（输入'exit'退出）：")
        if input_phone.lower() == "exit":  # 转换为小写并比较
            break  # 退出循环
        if validate_phone(input_phone):
            print("✅ 有效手机号码")
        else:
            print("❌ 无效手机号码")
    print("程序已退出。")
```

代码逐行解析

1. `while True:`：创建一个无限循环
2. `input_phone = input("请输入手机号码（输入'exit'退出）：")`：获取用户输入
3. `if input_phone.lower() == "exit":`：转换输入为小写并比较
4. `break`：退出循环
5. `if validate_phone(input_phone):`：调用验证函数
6. `print("✅ 有效手机号码")`：输出验证结果
7. `else:`：输入无效
8. `print("❌ 无效手机号码")`：输出无效提示
9. `print("程序已退出。")`：退出提示
10. `if __name__ == "__main__":`：主程序入口

扩展任务2：正则表达式实现

知识点：

- `re`模块的`match()`方法

- 正则表达式语法：
  `r'^1[3-9]\d{9}$'`

功能：

- `^1`:匹配1开头的11位数字
- `[3-9]`:第2位为3/4/5/7/8/9
- `\d{9}`:第3-11位为任意数字
- 返回匹配对象或None

代码示例

```
import  re  # 导入正则表达式模块
def validate_phone_regex(phone):
    """
    使用正则表达式验证手机号码
    参数：phone (str) - 待验证的号码
    返回：验证结果 (bool)
    """
    pattern = r'^1[3-9]\d{9}$'  # 正则表达式模式
    return bool(re.match(pattern, phone.strip()))   # 匹配并返回布尔值

# 主程序
if __name__ == "__main__":
    input_phone = input("请输入手机号码：")
    if validate_phone_regex(input_phone):    # 调用正则表达式验证函数
        print("✅ 有效手机号码")
    else:
        print("❌ 无效手机号码")
```

总结

1. 理解了手机号码验证的基本规则
2. 掌握了字符串基本操作
3. 学习了if选择结构的使用
4. 熟悉了IDLE调试环境
5. 提高了逻辑思维能力
6. 理解了程序的运行逻辑