

期末考查报告

《python 程序设计》

姓 名: _____

学 号: _____

系 别: 信息工程学院

专 业: 软件技术

年 级: 2024 级

班 级: _____

指导教师: 俞佳君

所 在 单 位： 2024 级 信息工程学院 系 软件技术专业 班

项目名称： 员工管理系统

项目目的：

设计一个员工信息管理系统，用于管理员工的基本信息，如姓名、工号、部门和联系方式。实现员工信息的增删改查功能。

项目内容及要求：

一、项目内容及要求

设计一个员工信息管理系统，用于管理员工的基本信息，如姓名、工号、部门和联系方式。具体要求如下：

- 1) 实现员工信息的增删改查功能。
- 2) 使用类和对象来表示员工信息。
- 3) 提供一个用户界面，允许 HR 人员进行员工信息管理。
- 4) 可以添加功能，如按部门或工号查询员工信息。

二、项目过程

1.显示所有操作

```
=====
员工信息管理系统
=====
1. 添加员工
2. 删除员工
3. 修改员工信息
4. 查询员工信息
5. 显示所有员工
0. 退出系统
=====
请选择操作 (0-5):
```

```
def __init__(self):
    """初始化系统，创建员工列表"""
    self.employees = []

def show_menu(self): 1个用法
    """显示主菜单"""
    print("\n" + "=" * 40)
    print("员工信息管理系统")
    print("=" * 40)
    print("1. 添加员工")
    print("2. 删除员工")
    print("3. 修改员工信息")
    print("4. 查询员工信息")
    print("5. 显示所有员工")
    print("0. 退出系统")
    print("=" * 40)

def add_employee(self): 1个用法
    """添加员工功能"""
    print("\n" + "=" * 40)
    print("添加员工信息")
    print("=" * 40)
```

2.增加员工

请选择操作 (0-5): 1

=====

添加员工信息

=====

请输入工号: |

获取员工信息

```
emp_id = input("请输入工号: ")
```

```
# 检查工号是否已存在
```

```
for employee in self.employees:
```

```
    if employee.emp_id == emp_id:
```

```
        print("错误: 该工号已存在! ")
```

```
        return False
```

```
name = input("请输入姓名: ")
```

```
department = input("请输入部门: ")
```

```
phone = input("请输入联系方式: ")
```

```
# 创建员工对象
```

```
new_employee = Employee(emp_id, name, department, phone)
```

```
# 添加到员工列表
```

```
self.employees.append(new_employee)
```

```
print(f"员工 {name} 添加成功! ")
```

```
return True
```

3.删除员工

请选择操作 (0-5): 2

=====

删除员工信息

=====

请输入要删除的员工工号: 1

找到员工: 1

确定要删除吗? (y/n): 1

删除操作已取消

```
def delete_employee(self): 1个用法
    """删除员工功能"""
    print("\n" + "=" * 40)
    print("删除员工信息")
    print("=" * 40)

    if len(self.employees) == 0:
        print("系统中暂无员工信息! ")
        return False

    emp_id = input("请输入要删除的员工工号: ")

    # 查找并删除员工
    found = False
    for i in range(len(self.employees)):
        if self.employees[i].emp_id == emp_id:
            found = True
            print(f"找到员工: {self.employees[i].name}")
            confirm = input("确定要删除吗? (y/n): ")

            if confirm.lower() == 'y':
                deleted_name = self.employees[i].name
                del self.employees[i]
```

4.显示所有员工信息

所有员工信息

=====

【技术部】部门（2人）：

工号：001，姓名：张三，联系方式：13800138000

工号：003，姓名：王五，联系方式：13700137000

【市场部】部门（1人）：

工号：002，姓名：李四，联系方式：13900139000

【人事部】部门（1人）：

工号：004，姓名：赵六，联系方式：13600136000

【1】部门（1人）：

工号：1，姓名：1，联系方式：1

```
# 显示每个部门的员工
for department, emp_list in departments.items():
    print(f"\n【{department}】部门（{len(emp_list)}人）:")
    print("-" * 40)
    for employee in emp_list:
        print(f"  工号：{employee.emp_id}, 姓名：{employee.name}, 联系方式：{employee.phone}")

    print(f"\n总计：{len(self.employees)} 名员工")

def run(self): 1个用法
    """运行员工信息管理系统"""
    # 添加一些示例数据
    self.employees.append(Employee(emp_id="001", name="张三", department="技术部", phone="13800138000"))
    self.employees.append(Employee(emp_id="002", name="李四", department="市场部", phone="13900139000"))
    self.employees.append(Employee(emp_id="003", name="王五", department="技术部", phone="13700137000"))
    self.employees.append(Employee(emp_id="004", name="赵六", department="人事部", phone="13600136000"))

    while True:
        self.show_menu()
```

个人总结:

通过对本学期 python 课程及个人项目实践的系统回顾,我完成了从离散知识接收者到完整项目构建者的关键转变。课程理论为我搭建了坚实的方法论框架:从需求分析中学会用用例图 and 用户故事精准捕捉业务场景,在 UML 建模过程中理解如何将现实世界的复杂关系抽象为清晰的类图与状态图,而敏捷开发与版本控制的实践则让我切身感受到迭代协作的真实节奏。这些理论在《基于 Python 的员工管理系统》个人项目中得到了完整贯通的验证:我以面向对象思想设计了可扩展的实体类结构,用文件操作模块实现数据持久化,并利用标准库与装饰器等特性提升代码质量;尤为重要的是,项目驱使我主动跨越技术舒适区——为改善命令行交互体验而探索 rich 库,为保障核心逻辑编写单元测试,这些在需求推动下自主拓展学习的过程,远比被动接受知识印象更为深刻。我认识到,一个系统的真正价值不仅在于功能实现,更在于其架构能否适应变化:通过模块化设计,我将数据层、业务层与表示层分离,使后续添加数据库支持或图形界面成为可能。这次从设计到部署的完整历程,让我深刻体会到软件工程是权衡的艺术——在功能完备性与开发效率间、在代码优雅性与交付时限间不断做出理性选择。未来,我将带着这种系统化思维与全链路实践能力,向更复杂的分布式架构或智能应用领域深入探索。

学生签名

指导教师签字 (签章):

俞佳君