编写程序

【内容描述】使用 Python 语言，设计一个小型的学生宿舍管理程序，系统用户为宿舍管理员。
【功能要求】
（1）学生信息：学号、姓名、性别(男/女)、宿舍房间号、联系电话。
（2）系统功能：
1. 可按学号查找某一位学生的具体信息；
2. 可以录入新的学生信息；
3. 可以显示现有的所有学生信息。
【程序要求】
（1）使用函数 列表 字典 字符串 条件循环等解决问题；
（2）程序规模在 80-200 行左右。

任务分析

数据结构

学生宿舍管理程序主要实现三个功能：查找学生信息、录入学生信息、显示所有学生信息。学生信息包括学号、姓名、性别、房间号和电话，其中学号学号和姓名最适合作为数据索引。但是学号作为纯数字相比字符串（姓名）在查找数据时不仅效率更高而且实现简单，更适合作为索引。同时，由于学生信息中包括姓名、性别、房间号等多种不同数据，相比列表等、元组等只能存放单一值的数据结构，采用 python 字典以多组键值对的形式存放数据更契合任务需求。综上所述，我认为可以基于 python 字典的数据结构实现上述三种系统功能。

```
self.main_dict[id]={"name":name ,"gender":gender,"dorm_num":dorm_num,"phone_num":phone_num}
```

功能分析

1. 查找学生信息功能可以要求用户输入学生学号，并通过学号直接从字典种索引到学生信息。其内部逻辑 Python 已经实现。
2. 录入学生信息功能的实现需要分两种情况完成。用户输入要添加的学生学号后，若字典中无此索引项，则直接在字典中添加学生信息，若已存在此索引项则直接覆盖旧信息。
3. 显示所有信息功能可以通过遍历字典中所有数据的方式实现。

程序代码

```
# shc 2024.3.19
class Database :
    # 构造函数
    def __init__(self) :
        self.all_data = self.Stduents_data()

    # 启动数据库
    def start(self) :
        Database.print_dividing(context='*')
        print("欢迎使用【宿舍管理系统】")
        while(True):   # 主循环
```

```
        # 打印操作提示
        print("1-查找学生")
        print("2-新增学生")
        print("3-显示全部")
        print("0-退出系统")
        # 用户输入操作代码，调用对应方法
        command_num = (input("请选择希望执行的操作："))
        if not command_num.isdigit(): # 判断输入信息合法性
            # print("invalid")
            continue
        command_num = int(command_num)
        if (command_num == 0): # 退出程序
            Database.print_dividing("_")
            break
        elif (command_num == 1): # 搜索学生
            self.search()
        elif (command_num == 2): # 添加学生
            self.adds()
        elif (command_num == 3): # 打印所有学生信息
            self.printall()
        else :
            print("非法输入！")
            Database.print_dividing()

# 学生搜索功能
def search(self):
    # 打印交互信息
    Database.print_dividing()
    print("搜索学生")
    print(" ")
    this_id = input("请输入要搜索的学号：")
    # 判断输入合法性
    if not this_id.isdigit():
        print("invalild")
        Database.print_dividing()
        return
    this_id = int(this_id)
    # 搜索学生
    this_students_dict = self.all_data.search_by_id(this_id)
    # 输出
```

```
    if this_students_dict == None:
        print("查无此人")
    else:
        Database.print_student_info_head()
        Database.print_student_info(this_id,this_students_dict)
    Database.print_dividing()

# 学生添加功能
def adds(self):
    # 打印交互信息
    Database.print_dividing()
    print("添加学生")
    print(" ")
    this_id = input("请输入学生的学号：")
    # 判断输入合法性
    if not this_id.isdigit():
        print("invalild")
        Database.print_dividing()
        return
    this_id = int(this_id)
        # 提示用户将覆盖原有信息，用户做出选择
    if self.all_data.search_by_id(this_id)!= None :
        print("注意：系统已存储该学生资料，将覆盖原有数据！")
        com =str(input("是否覆盖该条学生数据(y/[n]):"))
        if com != 'y':
            Database.print_dividing()
            return
    # 录入信息
    this_name = input("请输入学生的姓名：")
    this_gender = input("请输入学生的性别：")
    this_dorm_num = input("请输入学生的房间号：")
    this_phone_num = input("请输入学生的手机号：")
    self.all_data.add_student(this_id,this_name,this_gender,this_dorm_num,this_phone_num)
    print("添加成功!")
    Database.print_dividing()

    # 输出全部学生数据功能
def printall(self):
    Database.print_dividing()
    main_dict = self.all_data.get_dict()
    Database.print_student_info_head()
```

```
    for id , student_dict in main_dict.items():
        Database.print_student_info(id,student_dict)
    Database.print_dividing()

# 静态方法区
# 打印分割条
@staticmethod
def print_dividing(context="-"):
    string = ''
    for i in range(6):
        context += context
    print(context)
    # 打印学生信息表头
@staticmethod
def print_student_info_head():
        print("学号      姓名      性别      房间号      电话")
        Database.print_dividing(context='=')
    # 打印学生信息表体
@staticmethod
def print_student_info(this_id,this_students_dict):
    Database.print_one_info(this_id,is_integer=True)
    Database.print_one_info(this_students_dict["name"])
    Database.print_one_info(this_students_dict["gender"])
    Database.print_one_info(this_students_dict["dorm_num"])
    Database.print_one_info(this_students_dict["phone_num"])
    print(end='\n')
    # 实现表头与标体对齐输出
@staticmethod
def print_one_info(string,is_integer=False,column_len = 12):
    if is_integer: string = str(string)
    print(string, end= '')
    n = (column_len - len((string)))
    if n >0 :
        for i in range (n):
            print (end=' ')

# 学生数据存储类
class Stduents_data :
    def __init__(self):
        self.main_dict = {}
    #添加数据
    def add_student (self,id , name , gender , dorm_num , phone_num) :
        # if self.search_by_id(id) == None :
                                                                    self.main_dict[id]              =
```

```
{"name":name ,"gender":gender,"dorm_num":dorm_num,"phone_num":phone_num}
        #查找数据
        def search_by_id ( self, id ) :
            return   self.main_dict.get(id)
        # Getter 仅用于访问数据
        def get_dict(self):
            return self.main_dict

# main
db = Database()
db.start()
```