# 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()