第1次实验课作业（不用提交）

1.二分查找

给定一个 `n` 个元素有序的（升序）整型数组 `nums` 和一个目标值 `target` ，写一个函数 `BinarySearch` 搜索 `nums` 中的 `target` ，如果目标值存在返回下标，否则返回 `-1` 。

```
def BinarySearch(nums, target):
    """
    :param nums: list[int]
    :param target: int
    :return: int
    """

```

2.矩阵加法,乘法

给定两个$n \times n$的整型矩阵 `A` 和 `B` ，写两个函数 `MatrixAdd` 和 `MatrixMul` ，分别得出这两个矩阵加法和乘法的结果.

两个矩阵的数据类型为嵌套列表，即 `list[list]` ，且满足 `len(list)==n, len(list[0])==n` .

注意不要打乱原矩阵 `A` 和 `B` 中的数据.

```
def MatrixAdd(A, B):
    """
    :param A: list[list[int]]
    :param B: list[list[int]]
    :return: list[list[int]]
    """

def MatrixMul(A, B):
    """
    :param A: list[list[int]]
    :param B: list[list[int]]
    :return: list[list[int]]
    """
```

3.字典遍历

给定非空字典 `dict1` ，其键为姓名，值是学号. 写一个函数 `ReverseKeyValue` 返回另一个字典，其键是学号，值是姓名.

例如， `dict1={'Alice':'001', 'Bob':'002}` ，则 `ReverseKeyValue(dict1)` 返回的结果是 `{'001':'Alice', '002':'Bob'}` .

```
def ReverseKeyValue(dict1):
    """
    :param dict1: dict
    :return: dict
    """
```

4.管理Student数据

给定 `student_data.txt` 文本文件，每一行是一名学生的信息，从左到右分别是该学生的姓名,学号，性别和年龄，每个属性以空格间隔．数据类型如下：

```
name: str # 姓名
stu_num: str # 学号
gender: str # 性别, "M"为男性, "F"为女性
age: int # 年龄
```

编写 `StuData` 类，须有以下方法：

- 构造函数(即 `__init__`)，以文件名(`str` 类型，带 `.txt` 后缀)为输入，读取文件中的学生信息，存储到类成员 `data` 中． `data` 的数据类型为 `list` ，其中每一个学生的信息以列表方式存储．例如，读入一行学生信息"Aaron 243 M 18"，则 `data` 变为

  ```
  [["Aaron", "243", "M", 18]]
  ```

  再读入学生信息"Eric 249 M 19"，则 `data` 变为

  ```
  [["Aaron", "243", "M", 18], ["Eric", "249", "M", 19]]
  ```

- `AddData` 方法，以单个学生的信息作为输入，存储到 `data` 中．调用该方法的参数形式为学生属性的4个关键字实参．例如，执行 `self.AddData(name="Bob", stu_num="003", gender="M", age=20)` 后， `data` 变为

  ```
  [["Aaron", "243", "M", 18], ["Eric", "249", "M", 19], ["Bob", "003", "M", 20]]
  ```

- `SortData` 方法，以学生某个属性(`str` 类型，是 `'name', 'stu_num', 'gender', 'age'` 的其中之一)作为输入，将 `data` 按该属性从小到大排序．可以假定不会输入非学生属性的字符串．例如，执行 `self.Sort('stu_num')` 后， `data` 的学生信息按学号从小到大排序，变为

  ```
  [["Bob", "003", "M", 20],["Aaron", "243", "M", 18], ["Eric", "249", "M", 19]]
  ```

- `ExportFile` 方法，以导出的文件名(`str` 类型，带 `.txt` 后缀)为输入，新建一个 `txt` 文件，将 `data` 中的数据按当前列表顺序导出到该文件内，格式同原 `student_data.txt` 文本文件，即"姓名 学号 性别 年龄"，并存储在当前文件夹．例如，调用 `self.ExportFile('new_stu_data.txt')` ，则将 `data` 中数据导出 `new_stu_data.txt` 文件到当前文件夹．

提示

1. 学号信息数据类型为 `str` 而不是 `int` . 可以假定学号都由3个0-9数字组成.
2. 可以假设每个学生有且只有这4个属性, 且不会缺省.
3. 可以在类中编写其他辅助方法, 也可以在同一个代码文件中编写其他函数或类供自己调用.
4. 本次作业中, 类方法的输入参数名可自定义, 但参数数据类型需保证测试程序正常运行. 请不要更改类方法的名.
5. 调试代码时请将 `student_data.txt` 文件与代码文件放到同一文件夹中, 以避免不必要的bug. 提交代码时只提交一个 `.py` 代码文件, 请不要提交其他文件.