**Python程序设计实践课程作业**

**刘卓明**

**2151406**



基于Python实现基本的数据库功能

2022 年 11 月 28 日

同济大学电子与信息工程学院

目录

**1** 设计要求 ............................................................................................................................ 1

**2** 代码解析

2.1 头文件 .................................................................................................................... [2](#_bookmark1)

2.2 第一部分 ................................................................................................................... 2

2.3 第二部分 ................................................................................................................... 5

2.4 第三部分 ................................................................................................................... 6

2.5 菜单函数 ................................................................................................................... 8

2.6 试验函数 ................................................................................................................... 9

**3** 测试效果 ............................................................................................................................. 9

**4** 源代码 ............................................................................................................................. 12

**5** 心得与体会............................................................................................................................. 18

**1.设计要求**

用Python编写一个类，用于储存一个或多个dict对象，并能够实现简易的“增、删、改、查”操作。然后写一段简易的主程序测试一下各项功能。

**第一部分：**

（1）这个类应该能够像下面这样使用（假设这个类叫DictDatabase）：

首先创建一个这个类的实例：ddb = DictDatabase()

新增元素：ddb.add({'id': '20190001', 'name': '张三', 'age': 19})

删除元素：ddb.delete('age', 19)

更新元素：ddb.update('id', '20190001', {'name': '李四'})

替换元素：ddb.replace('id, '20190001', {'id': '20190002', 'name': '李四', 'age': 18})

查找元素：result = ddb.find('age', 19)

元素个数：num = ddb.count()

（2）能够将储存的数据保存到文件里，也能从文件中读入之前保存的数据（读入时原有的数据清空），自行选用合适的文件格式和读写方式。例如，先创建了一个DictDatabase的实例：

ddb = DictDatabase()

然后我们可以用如下方式让它将已储存的数据写入文件，或从文件中读入先前保存的数据：

ddb.save('文件名')

ddb.load('文件名')

**第二部分：**

在上一次的作业“文件里的dict数据库”的基础上，增添数据统计功能，利用numpy来实现。假设你写的类是DictDatabase，假设你创建了一个该类的实例ddb，具体要求如下：

（1）给DictDatabase增加getValues函数，用于获取数据库中所有dict的某一个属性的值。以下是调用方式示例：

values = ddb.getValues('weight')

该函数的第一个参数是要获取的key，返回值是一个numpy的array，里面包含了数据库里所有记录中的该key的值。不含有该key的记录不要出现在结果里。

（2）给上述getValues函数增加一个可选的参数，名字叫orderBy，用于指明是否要根据另一个属性的值来排序。只需要对int和float类型排序即可。例如：

values = ddb.getValues('weight', orderBy='height')

意思是“获取所有weight属性的值，并按照height属性的值从小到大排序”，返回的结果是一个n\*2的numpy的array，其中第0列是要查询的属性的值们（此例中是weight），第1列

是排序所用的属性的值（此例中是height）。不含有所查询的属性的记录不要出现在结果里。不含有orderBy属性的记录一律放在最后面，不参与排序。

（3）给DictDatabase增加一个getStat函数，用于统计所有dict中某一个属性的最小值、最大值、平均值，只需要对int和float类型作此处理，int类型的平均值也需要是float。例如：

stat = ddb.getStat('weight')

该函数需要返回一个dict，其中包含min、max、avg三个属性，其值分别是被统计的属性

（此例中是weight）的最小值、最大值、平均值。不含有指定属性的记录不参与统计。

第三部分：

在上一次的作业“带统计功能的dict数据库”的基础上，增加如下四种画图功能。假设你编写的类叫做DictDatabase，创建的实例叫做ddb。

（1）为DictDatabase增加一个函数plotLine，指定两个属性（属性对应的值都是数值）分别作为横轴和纵轴，绘制曲线图。例如，数据库里存有一年内每天的气温，下面的代码将画出气温随天数的变化曲线。

ddb.plotLine('dayOfYear', 'temperature')

（2）为DictDatabase增加一个函数plotBars，指定两个属性（属性对应的值分别为字符串和数值），绘制柱状图。例如，数据库里存有每个国家的名字和人口数量，下面的代码将绘制柱状图，反映每个国家人口数量的多少。

ddb.plotBars('country', 'population')

1. 为DictDatabase增加一个函数plotPie，第一个参数是一个属性（属性对应的值是数

值），第二个参数是一个list，其中代表了统计区段的分隔点，绘制饼状图，反映该属性的值位于每个区段的百分比。例如，数据库里存着每位同学的成绩分数，下面的代码将绘制饼状图，里面的扇形分别代表分数位于(-∞,60)、[60,90)、[90,100)、[100,+∞)的人数。注意：画出来的饼状图的扇形块数应该是第二个参数list的长度+1。

ddb.plotPie('score', [60, 90, 100])

（4）为DictDatabase增加一个函数plotPoints3D，该函数接3个或4个参数。3个参数时，是三个属性名，用这三个属性依次作为x、y、z坐标，绘制三维散点图。4个参数时，前3个属性代表坐标，用第4个属性对应的值来涂颜色。例如，数据库里存着各种型号手机的CPU频率、屏幕大小、电池寿命和价格等数据，下面的第一行代码分别以CPU频率、屏幕大小、待机时间作为坐标，把所有条目对应的点绘制在三维坐标系里。下面的第二行代码则在第一行的基础上，将所有点按照价格涂色。

ddb.plotPoints3D('cpuFreq', 'screenSize', 'batteryLife')

ddb.plotPoints3D('cpuFreq', 'screenSize', 'batteryLife', 'price')

1. **代码解析**

**2.1头文件**



matploblib.plot库是用来可视化绘图的库，numpy是一个以数组为核心的库，第三行的库是用来三维绘图的库，pandas是主要用于处理二维数据的库，其核心的数据结构是Dataframe，json是一个用来将字符串形式的字典转换为实际的字典的库。

至于colors\_all是一个有十个颜色的列表，是为了后面绘图时候取不同颜色方便用。

补：后来又补充导入了matploblib，为了画出3D图中的颜色轴

**2.2第一部分**



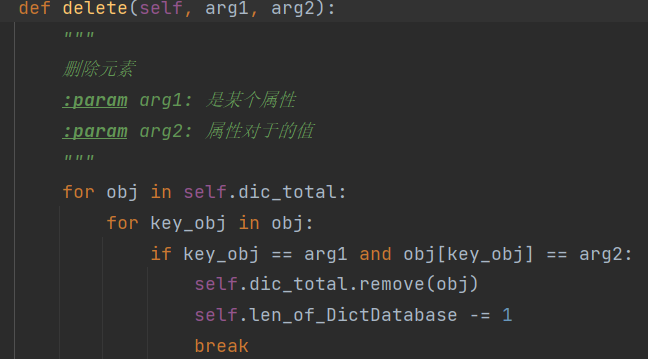
开始：整体数据库的实现是通过创建一个类实现的，图片中类的前两行是代表了数据库中元素的个数，以及创建了一个空列表，后续添加的元素都是字典，作为元素放在这个列表中。

****\_\_init\_\_函数****（方法）：

这个函数是用来初始化这个类的，初始时候将长度设定为0，并且通过for循环的方式实现了将每个人（或者是其他）保存为字典，并且将字典作为了总列表的每一个元素。

add函数：

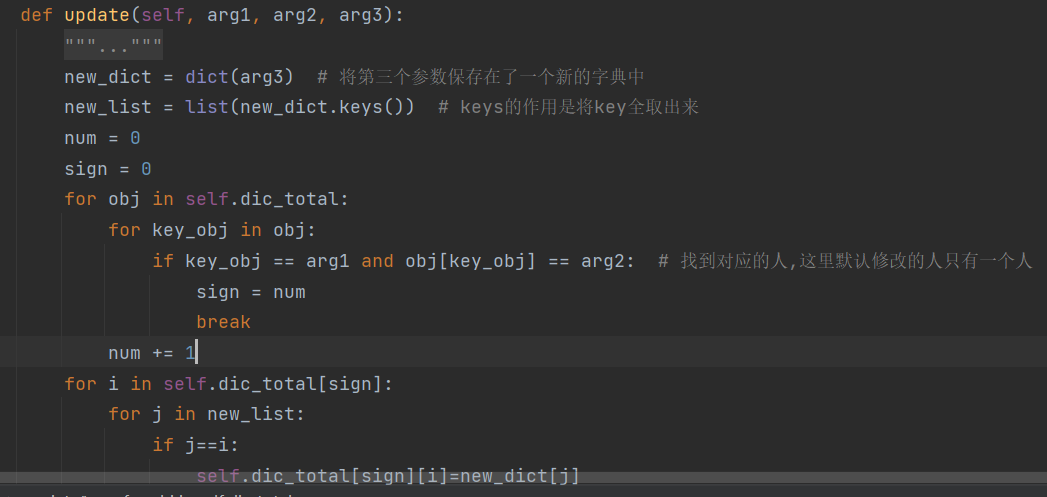
作用是增加新的元素，增加时，需要让总长度加一，然后通过列表的append方法去增加元素。



delete函数：

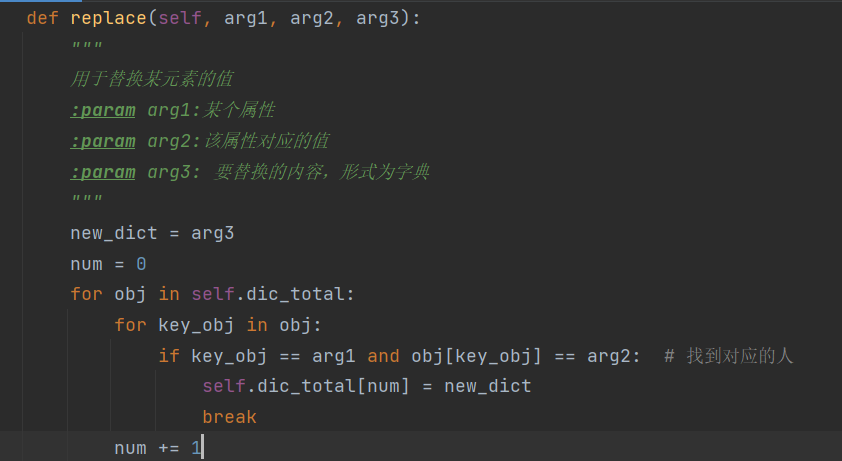
作用是删除元素。主要通过双重的for循环实现，第一层for循环在列表中遍历每一个字典，第二层for循环在每一个字典中遍历每一个键值对，当找到匹配项后，利用列表的remove方法删除掉整个字典，然后长度减一；break是找到这个键值对后，直接跳出遍历下一个字典即可。

update函数：

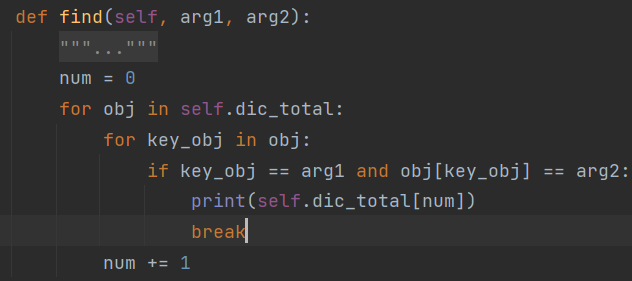


这个函数的利用了字典的keys方法，作用是将要修改的内容的所有键都取出来，然后在更新部分的思路和delete函数基本一样，即利用两层for循环找到修改的项，然后此时将需要修改的字典的序号记录下来，重新对改字典进行遍历，利用存储好的键，进行修改，此处只有一个标记，也就是默认只改一个人的（这里想到了改几个人的思路，就是可以采用标记的方法，即直接去记录每个键值对在每个字典中出现的位置（可以用列表保存），然后直接跳出循环，因为每个字典这个键值对的位置都是一样的，然后修改下一个人的时候可以直接根据标记的号去修改，这样就可以实现修改多个人了，目前采用的标记方法仅仅适用于修改一个人的元素，一般改成绩也就是一个人，但比如说出现其他生活情况的时候，可能会需要修改多个人，如：指定修改工龄为10年的员工的绩效工资为5000元，提成为3000元）

此时参数为（‘age’, 10 , {‘jxgz’:5000,’tc’:3000}）(这种情况下满足工龄为10的肯定不止一个人）

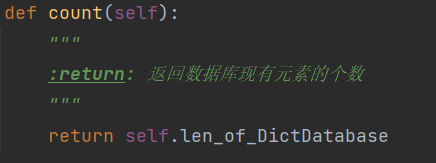


replace函数：

本函数的核心思路仍和前两个函数相同，仍然是通过两重for循环去实现寻找与修改的功能，然后整个替换掉即可。

find函数：

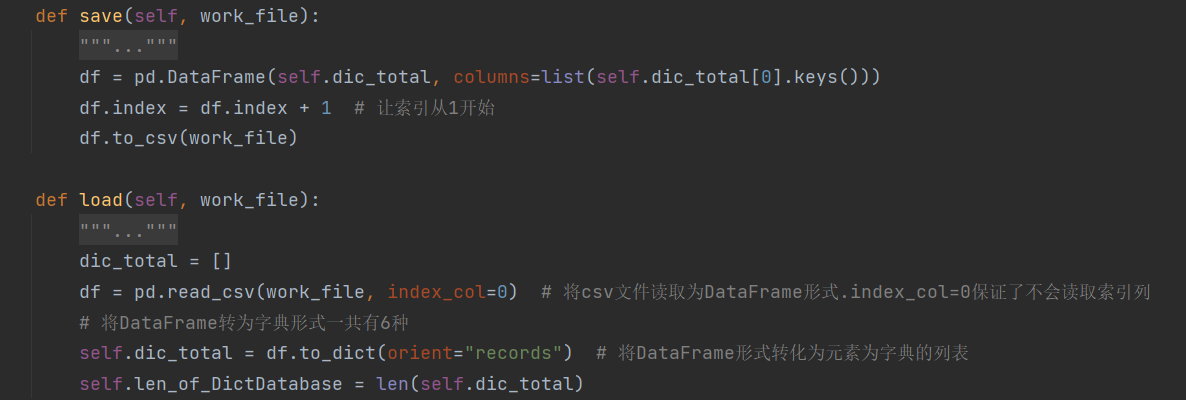
用于查找指定的项，思路同样与前几个函数相同，这里是将找到的变量打印出来，也可以考虑将符合的项先一起存起来（到列表中），然后一起输出



count函数：

由于在创建类的时候有创建记录列表长度的变量，所以直接输出即可

接下来是文件的保存和读取，这里由于我们的数据很多时候都是用表格文件而不是用文本文件，所以考虑用表格文件来输入输出，这里我们在实验指导书中学习了pandas库，这个库中有一种非常方便的数据结构是Dataframe，它是类似于二维的表格形式，并且它可以转化为csv文件（一种表格文件），同样csv文件也可以导入为Dataframe。



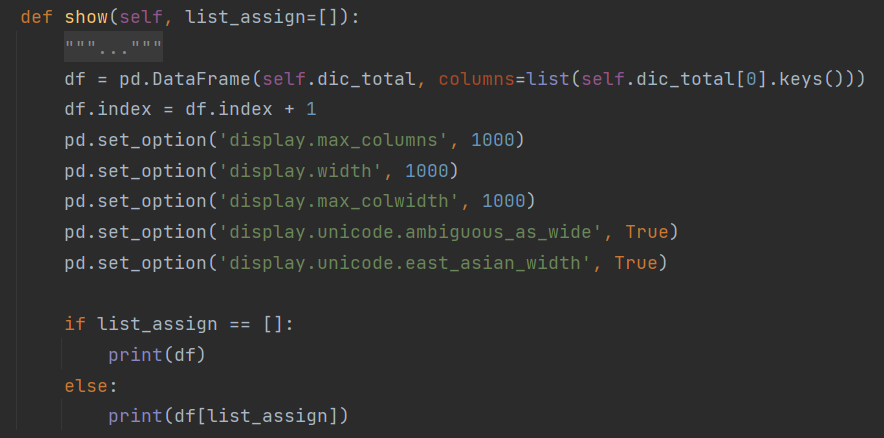
save函数：

这里可以直接将我们以字典为元素的列表转化为dataframe形式，然后columns参数是表格文件中每一列的标题，我们利用keys方法取出所有键，然后作为每一列的标题即可。

然后index是表格的索引，我们这里让index从1开始；然后用to\_csv函数就可以导出为csv文件了。

load函数

利用read\_csv函数可以将表格文件转成dataframe的形式，然后利用to\_dict函数可以直接将dataframe转换成字典形式（根据orient参数一共有六种选择），这里恰好可以转成元素为字典的列表形式（其他几种是不同形式的种类）



show函数：

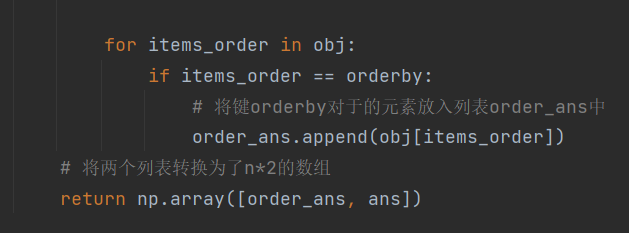
自己添加的一个函数，用于展示当前的数据库，可以指定全部输出，或者是只输出某些属性；

同样可以实现根据dataframe的index实现只输出其中几行（本函数没有实现，由于find函数有类似的功能）

中间的五行set\_option是用来调整dataframe在pycharm中的显示问题，当每一列的标题是汉字是会出现对不齐的问题，此处经调整后还有略微对不齐（待解决）。

**2.3第二部分**

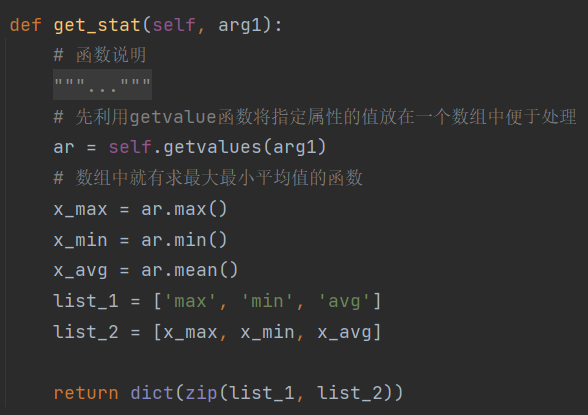




getvalues函数：

这个函数的作用是获取某一个指定key的所有值，输出为n\*1的数组；当有第二个参数时，根据第二个参数对于提取的数据进行排序。这里利用到了numpy库，这个库的主要是含有了一个数据结构数组，无参数的情况思路同样是两重循环的方式，然后利用append方法去不断添加key的值就可以了。

当有第二个参数时，需要根据第二个元素进行排序这里采用的是排序函数sorted，其中的参数lambda指明了排序方式是根据orderby的降序，最后利用同样的方式输出，两个n\*1的数组经过array函数可以组成n\*2的数组。



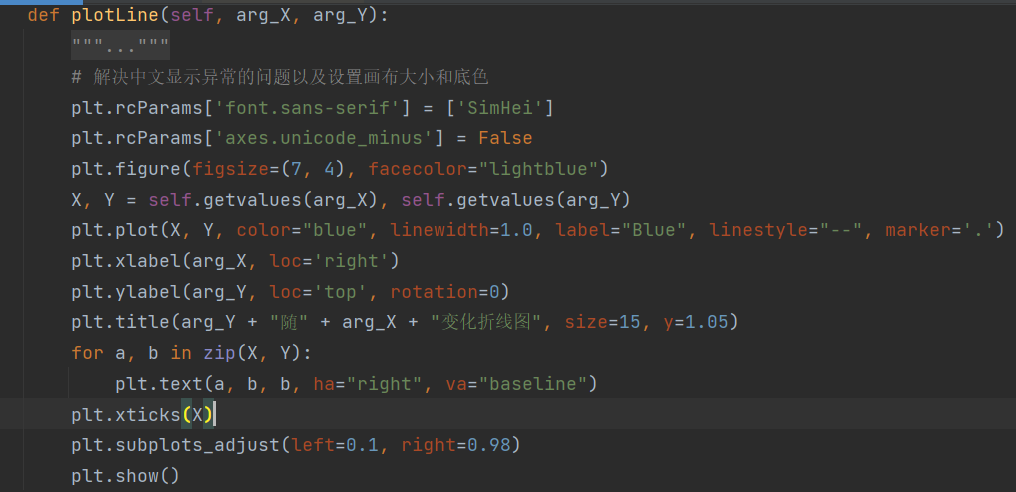
get\_stat函数

函数功能：获取某一属性的最大，最小，平均值

该函数利用了上一个函数，将某个属性的所有值都放到一个数组中，然后利用数组内置的最大、最小、平均值的函数可以直接求出，然后在将结果存储为字典的时候，利用了zip函数，将键和其对应的值打包成元组，然后用dict函数转化为了字典。（创建字典的方法，了解到了6种）

**2.4第三部分**

第三部分是绘图部分，利用的库主要是matploblib.plot库。



plotLine函数

用于绘制折线图，前面的两行主要是为了解决中文显示异常的问题。figure用来设置画布的大小，背景颜色。利用getvalues可以取出指定属性对应的值，然后利用plot函数就可以绘制折线图，其中可以自己设置多个参数，如线宽，线的类型，线的颜色等，marker是指每个点的类型（点、圆、三角形都可以选择）

然后可以利用text函数标注出每个点的数据，其中a，b是两个轴的数据，第三个参数指明要标出的数据，ha和va是水平、竖直对齐方式。

title函数是用于标明标题（图的名称）的函数。

xlabel、ylabel是用来标注横纵轴各自的名称的，loc参数是指对齐方式，rotation参数是旋转角度，这里发现用xlabel标注的横坐标不是每个值都显示，所以又用xticks函数又标注了一次。subplots\_adjust函数是用来调整画出的图离窗口四边的距离。最后show函数呈现出绘的图。



plotBars函数

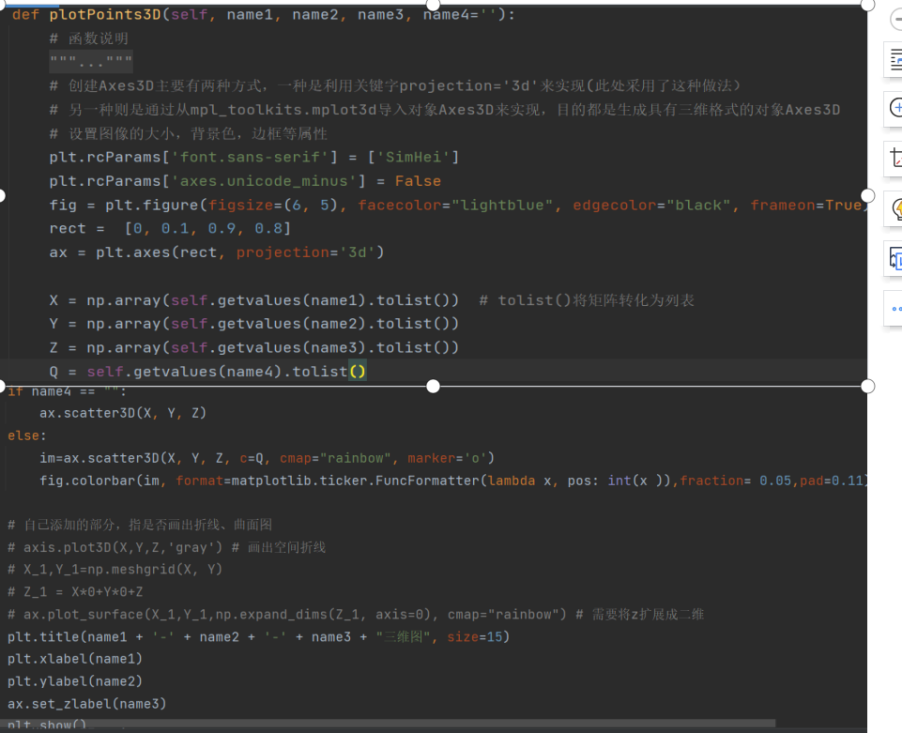
本函数的功能是绘制条形图，和绘制折线图时候相同的部分就不再赘述。

绘制条形图利用的函数主要是bar函数，这里比条形图不一样的参数有线宽，以及可以设定各个条形的颜色，在这里用到了最开始在头文件部分介绍的colors\_all列表，可以从其中取出颜色，这里由于总列表中有10种颜色，为了防止横轴的数据过多（如一个月的温度变化，那么横轴会有30个左右的值），采用取模的方法保证不会发生越界问题。

 plotPie函数

该函数是用于绘制饼图的函数，图1中的内容主要是为了统计各个分数区间各有多少人，这里关键思想就是上界和下界去作差，即小于上界的人数减去小于下界的人数，那就是中间这个区间的人数。

这个函数中的关键部分是各个扇形名称的标注，这里要分三种情况讨论，即第一个区间，那么范围就是0-所给区间的第一个数，这里做了特殊判断，就是如果第一个数是0，那么直接输出“0分人数为”，否则输出“0-m分”；中间区间的话处理是正常的，标签输出采用字符串相连的形式输出。第三部分就是最后一个区间，范围就是所给的最后一个数--100（这里假定分数上限是100），同样需要做特殊处理，那就是如果所给的最后一个数就是100，那么直接输出“100分的人数为”，否则输出“m--100分的人数为”。（这里特殊处理的方式是单独输出0分和100分的，也可以在统计各分段人数前就处理，让这种情况归入区间中，即变为相应的闭区间）

而输出饼图用的是pie函数，这里的参数explode是用来表示每一部分扇形突出的程度，这里本人凸出表示了最低分数段的人数，autopct是输出每部分所占百分比的数据类型，startangle是表示第一块扇形开始的位置角度，countercloc用来表示顺时针还是逆时针。

plotpoint3D函数

这个函数是用来绘制三维散点图的，这里需要创建Axes3D对象，方式有两种，这里是利用axes函数中的“projection=‘3d’”实现的，列表rect的目的是调整图形的位置及大小。然后3维的对象和前面几种2维的图形不同，在利用ax的scatter3D函数时，传入的形式需要是数组，所以这里用array函数先将三个轴的数据转换为数组。

然后这个题要求用第四个参数来给前三个参数形成的散点图涂色，这里利用的是scatter3D函数的c参数，cmap参数是指用来涂色的颜色（为渐变色）。

而fig.colorbar则是绘出颜色的比例轴，其中第二个参数可以用于修改比例的范围，如让x乘上100，那么范围就会变为0-10000（因为当前的范围是0-100），如果加上数值就会改变最小值，fraction参数是调整比例轴的大小，而pad参数是调整比例轴离图形的距离。

注释部分是自己添加的部分，可以实现绘制出散点的折线图，以及覆盖散点的曲面。

**2.5菜单函数**

菜单部分由于较长，就不贴出代码的图了，详细代码见第四部分的源代码。

基本思路就是先打印出菜单的样式，然后通过一个while循环来实现基本的交互，用户输入想要选取的功能，然后根据if-else结构跳转到对应的功能，然后这里直接将每个功能函数的参数写为input（），方便处理，对于大部分参数来说，都是字符串和数值，input（）可以直接识别使用，对于部分几个字典或者是列表类型的参数，使用到了类型转换函数。

对于参数变量可变的函数，input（）的数量为最大参数个数，如不需要后几个参数，直接多摁一次回车即可。

菜单部分有部分的文字提醒输入内容与方式，基本输入方式是严格按照作业要求中各个函数的参数形式来写的，所以交互性不是特别强，有待改善。

**2.6试验函数**

本函数是一个试验函数，其中涵盖了所有的功能，此处注意的地方是由于读入文件时候是读取的csv文件，则读到的学号是数字的类型，所以为了方便统一处理，在利用其它函数涉及到学号的操作时都以数值类型进行（当然也可以读入之后在利用dataframe去整体修改学号这一列的数据类型）。具体测验效果见第三部分。

1. **测试效果**

id name age 高数 线代 模电 概率论 马原 离散 数据结构

1 20190001 张三 19 97 89 87 80 93 87 80

2 20190002 李四 18 82 80 69 84 87 90 90

3 20190003 王五 19 74 78 87 75 76 88 76

4 20190004 赵六 19 50 62 65 71 78 66 74

5 20190005 李小光 20 36 57 64 77 80 44 92

6 20190006 王大鹏 21 58 70 71 76 90 69 82

7 20190007 李云 37 63 79 76 81 93 63 83

8 20190008 刘红 38 96 98 97 93 57 94 88

9 20190009 王杰 6 73 80 75 83 60 77 74

10 20190010 小花 5 37 48 87 49 50 64 58

11 20190011 张强 25 100 85 70 94 93 91 87

12 20190012 段玉 30 68 89 87 70 93 87 74

13 20190013 小明 19 40 54 30 58 72 50 63

14 20190014 张彪 32 70 55 78 82 94 73 71

15 20190015 李华 17 100 99 97 100 95 100 100

16 20190016 赵明 35 95 89 88 92 100 85 98

增添了一位同学

id name age 高数 线代 模电 概率论 马原 离散 数据结构

1 20190001 张三 19 97 89 87 80 93 87 80

2 20190002 李四 18 82 80 69 84 87 90 90

3 20190003 王五 19 74 78 87 75 76 88 76

4 20190004 赵六 19 50 62 65 71 78 66 74

5 20190005 李小光 20 36 57 64 77 80 44 92

6 20190006 王大鹏 21 58 70 71 76 90 69 82

7 20190007 李云 37 63 79 76 81 93 63 83

8 20190008 刘红 38 96 98 97 93 57 94 88

9 20190009 王杰 6 73 80 75 83 60 77 74

10 20190010 小花 5 37 48 87 49 50 64 58

11 20190011 张强 25 100 85 70 94 93 91 87

12 20190012 段玉 30 68 89 87 70 93 87 74

13 20190013 小明 19 40 54 30 58 72 50 63

14 20190014 张彪 32 70 55 78 82 94 73 71

15 20190015 李华 17 100 99 97 100 95 100 100

16 20190016 赵明 35 95 89 88 92 100 85 98

17 20190017 旺财 15 99 90 72 77 93 50 59

张三同学转专业了，删除掉

id name age 高数 线代 模电 概率论 马原 离散 数据结构

1 20190002 李四 18 82 80 69 84 87 90 90

2 20190003 王五 19 74 78 87 75 76 88 76

3 20190004 赵六 19 50 62 65 71 78 66 74

4 20190005 李小光 20 36 57 64 77 80 44 92

5 20190006 王大鹏 21 58 70 71 76 90 69 82

6 20190007 李云 37 63 79 76 81 93 63 83

7 20190008 刘红 38 96 98 97 93 57 94 88

8 20190009 王杰 6 73 80 75 83 60 77 74

9 20190010 小花 5 37 48 87 49 50 64 58

10 20190011 张强 25 100 85 70 94 93 91 87

11 20190012 段玉 30 68 89 87 70 93 87 74

12 20190013 小明 19 40 54 30 58 72 50 63

13 20190014 张彪 32 70 55 78 82 94 73 71

14 20190015 李华 17 100 99 97 100 95 100 100

15 20190016 赵明 35 95 89 88 92 100 85 98

16 20190017 旺财 15 99 90 72 77 93 50 59

修改了李华同学的成绩

id name age 高数 线代 模电 概率论 马原 离散 数据结构

1 20190002 李四 18 82 80 69 84 87 90 90

2 20190003 王五 19 74 78 87 75 76 88 76

3 20190004 赵六 19 50 62 65 71 78 66 74

4 20190005 李小光 20 36 57 64 77 80 44 92

5 20190006 王大鹏 21 58 70 71 76 90 69 82

6 20190007 李云 37 63 79 76 81 93 63 83

7 20190008 刘红 38 96 98 97 93 57 94 88

8 20190009 王杰 6 73 80 75 83 60 77 74

9 20190010 小花 5 37 48 87 49 50 64 58

10 20190011 张强 25 100 85 70 94 93 91 87

11 20190012 段玉 30 68 89 87 70 93 87 74

12 20190013 小明 19 40 54 30 58 72 50 63

13 20190014 张彪 32 70 55 78 82 94 73 71

14 20190015 李华 17 100 99 97 100 95 100 89

15 20190016 赵明 35 95 89 88 92 100 85 98

16 20190017 旺财 15 99 90 72 77 93 50 59

寻找了高数满分的同学成绩如下：

{'id': 20190011, 'name': '张强', 'age': 25, '高数': 100, '线代': 85, '模电': 70, '概率论': 94, '马原': 93, '离散': 91, '数据结构': 87}

{'id': 20190015, 'name': '李华', 'age': 17, '高数': 100, '线代': 99, '模电': 97, '概率论': 100, '马原': 95, '离散': 100, '数据结构': 89}

元素个数为：16

将学号为007的同学换成了023的

id name age 高数 线代 模电 概率论 马原 离散 数据结构

1 20190002 李四 18 82 80 69 84 87 90 90

2 20190003 王五 19 74 78 87 75 76 88 76

3 20190004 赵六 19 50 62 65 71 78 66 74

4 20190005 李小光 20 36 57 64 77 80 44 92

5 20190006 王大鹏 21 58 70 71 76 90 69 82

6 20190023 李龙 23 73 80 75 82 90 74 78

7 20190008 刘红 38 96 98 97 93 57 94 88

8 20190009 王杰 6 73 80 75 83 60 77 74

9 20190010 小花 5 37 48 87 49 50 64 58

10 20190011 张强 25 100 85 70 94 93 91 87

11 20190012 段玉 30 68 89 87 70 93 87 74

12 20190013 小明 19 40 54 30 58 72 50 63

13 20190014 张彪 32 70 55 78 82 94 73 71

14 20190015 李华 17 100 99 97 100 95 100 89

15 20190016 赵明 35 95 89 88 92 100 85 98

16 20190017 旺财 15 99 90 72 77 93 50 59

模电成绩如下：

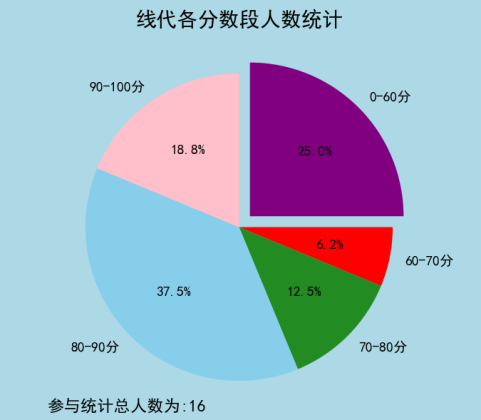
[69. 87. 65. 64. 71. 75. 97. 75. 87. 70. 87. 30. 78. 97. 88. 72.]

根据数据结构成绩降序排布的模电成绩（第一行为数据结构成绩）

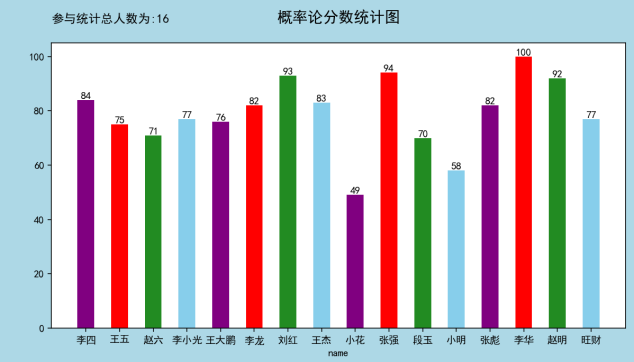
[[98 92 90 89 88 87 82 78 76 74 74 74 71 63 59 58]

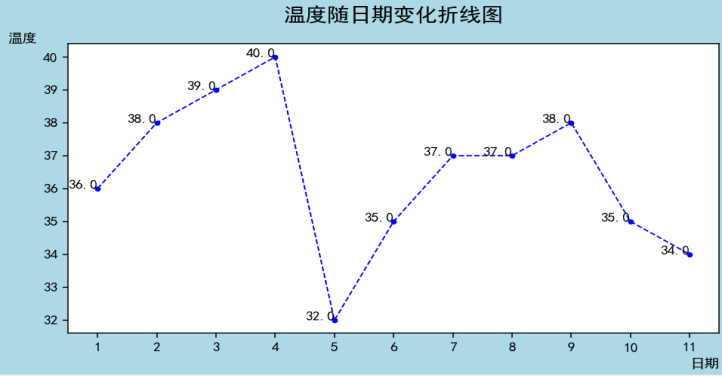
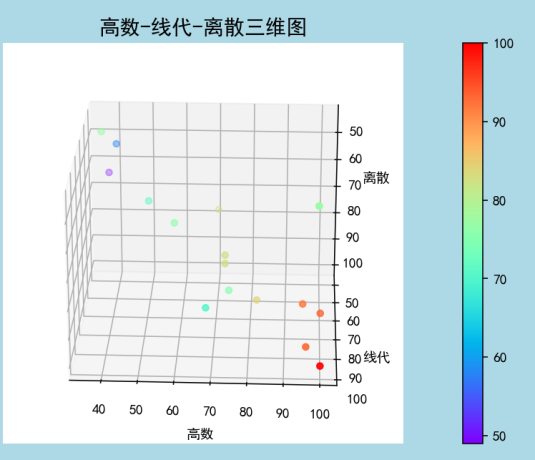
[88 64 69 97 97 70 71 75 87 65 75 87 78 30 72 87]]

离散的最高，最低，平均分：

{'max': 100.0, 'min': 44.0, 'avg': 75.125}

绘图部分贴图如下





**4.源代码:**

# 同济大学 人工智能

# 2151406刘卓明

# 开发时间：2022/11/17 18:47

import matplotlib.pyplot as plt

import numpy as np

from mpl\_toolkits.mplot3d import Axes3D

import pandas as pd

import json

import matplotlib

colors\_all = ["purple", "red", "forestgreen", "skyblue", "pink", "yellow", "chocolate", "greenyellow", "khaki", "aqua"]

class DictDatabase:

len\_of\_DictDatabase = 0

dic\_total = [] # 创建空列表，列表中的每个元素是字典

def \_\_init\_\_(self):

self.len\_of\_DictDatabase = 0

self.dic\_total = list(dict() for i in range(self.len\_of\_DictDatabase))

# 第一部分：基本操作

def add(self, new\_dic):

"""

功能：增加元素

:param new\_dic: 形式为字典，是增添的元素

"""

self.len\_of\_DictDatabase += 1

self.dic\_total.append(new\_dic)

def delete(self, arg1, arg2):

"""

删除元素

:param arg1: 是某个属性

:param arg2: 属性对于的值

"""

for obj in self.dic\_total:

for key\_obj in obj:

if key\_obj == arg1 and obj[key\_obj] == arg2:

self.dic\_total.remove(obj)

self.len\_of\_DictDatabase -= 1

break

def update(self, arg1, arg2, arg3):

"""

用于更新某元素的值

:param arg1: 某个属性

:param arg2: 该属性对应的值

:param arg3: 要修改的内容，形式为字典

"""

new\_dict = dict(arg3) # 将第三个参数保存在了一个新的字典中

new\_list = list(new\_dict.keys()) # keys的作用是将key全取出来

num = 0

sign = 0

for obj in self.dic\_total:

for key\_obj in obj:

if key\_obj == arg1 and obj[key\_obj] == arg2: # 找到对应的人,这里默认修改的人只有一个人

sign = num

break

num += 1

for i in self.dic\_total[sign]:

for j in new\_list:

if j==i:

self.dic\_total[sign][i]=new\_dict[j]

def replace(self, arg1, arg2, arg3):

"""

用于替换某元素的值

:param arg1:某个属性

:param arg2:该属性对应的值

:param arg3: 要替换的内容，形式为字典

"""

new\_dict = arg3

num = 0

for obj in self.dic\_total:

for key\_obj in obj:

if key\_obj == arg1 and obj[key\_obj] == arg2: # 找到对应的人

self.dic\_total[num] = new\_dict

break

num += 1

def find(self, arg1, arg2):

"""

查找具体的某个元素

:param arg1: 某个属性

:param arg2: 该属性对应的值

"""

num = 0

for obj in self.dic\_total:

for key\_obj in obj:

if key\_obj == arg1 and obj[key\_obj] == arg2:

print(self.dic\_total[num])

break

num += 1

def count(self):

"""

:return: 返回数据库现有元素的个数

"""

return self.len\_of\_DictDatabase

def show(self, list\_assign=[]):

"""

呈现当前的数据库

可以选择全部呈现或者是呈现某些列

利用DataFrame实现，也可以选择呈现某些行（本函数暂未实现）

:param list\_assign: 可选参数，可以选择要查看的列，将键以列表形式输入即可

"""

df = pd.DataFrame(self.dic\_total, columns=list(self.dic\_total[0].keys()))

df.index = df.index + 1

pd.set\_option('display.max\_columns', 1000)

pd.set\_option('display.width', 1000)

pd.set\_option('display.max\_colwidth', 1000)

pd.set\_option('display.unicode.ambiguous\_as\_wide', True)

pd.set\_option('display.unicode.east\_asian\_width', True)

if list\_assign == []:

print(df)

else:

print(df[list\_assign])

def save(self, work\_file):

"""

将数据存储到文件中

:param work\_file: 文件名

"""

df = pd.DataFrame(self.dic\_total, columns=list(self.dic\_total[0].keys()))

df.index = df.index + 1 # 让索引从1开始

df.to\_csv(work\_file)

def load(self, work\_file):

"""

用于读取文件中的数据

:param work\_file: 文件名称

"""

dic\_total = []

df = pd.read\_csv(work\_file, index\_col=0) # 将csv文件读取为DataFrame形式.index\_col=0保证了不会读取索引列

# 将DataFrame转为字典形式一共有6种

self.dic\_total = df.to\_dict(orient="records") # 将DataFrame形式转化为元素为字典的列表

self.len\_of\_DictDatabase = len(self.dic\_total)

# 第二部分：数值处理

def getvalues(self, arg1, orderby=""):

# 函数说明

"""

用于获取数据库中所有dict的某一个属性的值

:param arg1:是要获取的key，返回值是一个numpy的array，里面包含了数据库里所有记录中的该key的值

:param orderby:用于指明是否要根据另一个属性的值来排序,返回的结果是一个n\*2的numpy的array，

:return:当没有第二个参数时,返回一维数组，有第二个参数时候，返回二维数组

"""

# 分两种情况，就是有没有参数的情况，下面是没有参数的情况，返回的是一个n\*1的数组

# 获取的数组是各个字典中键arg1对应的值

if orderby == "":

ans = np.array([])

for obj in self.dic\_total:

# 寻找指定的键值

for items in obj:

if items == arg1:

ans = np.append(ans, obj[items])

return ans

# 下面是有参数的情况

else:

# 新建两个列表，第一个列表用于存放arg1对应的值，第二个列表用于存放键orderby对应的值

ans = []

order\_ans = []

# 下面的列表mid\_dic是dic\_total的copy，为了保证不改变原先的列表

mid\_dic = self.dic\_total.copy()

"""

sorted函数返回值是一个列表，其功能是对序列（列表、元组、字典、集合、还包括字符串）进行排序。

第一个参数是迭代对象，key是用来比较，reverse代表升序还是降序

L=[('b',2),('a',1),('c',3),('d',4)]

sorted(L, cmp=lambda x,y:cmp(x[1],y[1])) # 利用cmp函数

[('a', 1), ('b', 2), ('c', 3), ('d', 4)]

sorted(L, key=lambda x:x[1]) # 利用key，此处是指x的第一个元素

[('a', 1), ('b', 2), ('c', 3), ('d', 4)]

对于key，冒号前的是用于比较的（键），冒号后的则是键对应的值

"""

mid\_dic = sorted(mid\_dic, key=lambda item: item[orderby], reverse=True) # 根据orderby降序排序

for obj in mid\_dic:

for items in obj:

if items == arg1:

# 将键arg1对于的元素值放入列表ans中

ans.append(obj[items])

for items\_order in obj:

if items\_order == orderby:

# 将键orderby对于的元素放入列表order\_ans中

order\_ans.append(obj[items\_order])

# 将两个列表转换为了n\*2的数组

return np.array([order\_ans, ans])

def get\_stat(self, arg1):

# 函数说明

"""

用于统计所有dict中某一个属性的最小值、最大值、平均值

:param arg1:需要参与统计的属性

:return:返回一个dict，其中包含min、max、avg三个属性

"""

# 先利用getvalue函数将指定属性的值放在一个数组中便于处理

ar = self.getvalues(arg1)

# 数组中就有求最大最小平均值的函数

x\_max = ar.max()

x\_min = ar.min()

x\_avg = ar.mean()

list\_1 = ['max', 'min', 'avg']

list\_2 = [x\_max, x\_min, x\_avg]

return dict(zip(list\_1, list\_2))

# 第三部分：绘图

def plotLine(self, arg\_X, arg\_Y):

"""

指定两个属性（属性对应的值都是数值）分别作为横轴和纵轴，绘制曲线图。例如，数据库里存有一年内每天的气温，下面的代码将画出气温随天数的变化曲线。

:param arg\_X:横轴的名称，对应的值是数值

:param arg\_Y:纵轴的名称，对应的值是数值

"""

# 解决中文显示异常的问题以及设置画布大小和底色

plt.rcParams['font.sans-serif'] = ['SimHei']

plt.rcParams['axes.unicode\_minus'] = False

plt.figure(figsize=(7, 4), facecolor="lightblue")

X, Y = self.getvalues(arg\_X), self.getvalues(arg\_Y)

plt.plot(X, Y, color="blue", linewidth=1.0, label="Blue", linestyle="--", marker='.')

plt.xlabel(arg\_X, loc='right')

plt.ylabel(arg\_Y, loc='top', rotation=0)

plt.title(arg\_Y + "随" + arg\_X + "变化折线图", size=15, y=1.05)

for a, b in zip(X, Y):

plt.text(a, b, b, ha="right", va="baseline")

plt.xticks(X)

plt.subplots\_adjust(left=0.1, right=0.98)

plt.show()

def plotBars(self, arg\_X, arg\_Y):

"""

这个函数是用来绘制柱形图的，

例如，数据库里存有每个国家的名字和人口数量，下面的代码将绘制柱状图，反映每个国家人口数量的多少。

:param arg\_X:X轴的名称，对应的值是字符串

:param arg\_Y:Y轴的名称，对应的值是数值

:return:

"""

plt.rcParams['font.sans-serif'] = ['SimHei']

plt.rcParams['axes.unicode\_minus'] = False

plt.figure(figsize=(10, 5), facecolor="lightblue")

X, Y = self.getvalues(arg\_X), self.getvalues(arg\_Y)

plt.xlabel(arg\_X)

colors = ["red"] \* len(arg\_X)

for i in range(len(arg\_X)):

colors[i] = colors\_all[i % 10]

for a, b in zip(X, Y):

plt.text(a, b, int(b), ha="center", va="bottom") # x,y文本所在位置，s为显示内容，ha水平对齐方式，va垂直对齐方式

plt.title(arg\_Y + "分数统计图", size=15, y=1.05)

plt.bar(X, Y, width=0.5, color=colors)

plt.title("参与统计总人数为:" + str(self.len\_of\_DictDatabase), y=1.05, size=12, loc='left')

plt.show()

def plotPie(self, arg\_1, arg\_list):

# 函数说明

"""

绘制饼状图，反映该属性的值位于每个区段的百分比。

例如，数据库里存着每位同学的成绩分数，下面的代码将画出位于各个分数区间的人数

:param arg\_1:是一个属性（属性对应的值是数值）

:param arg\_list:代表了统计区段的分隔点

"""

# 解决中文显示异常的问题以及设置画布大小和底色

plt.rcParams['font.sans-serif'] = ['SimHei']

plt.rcParams['axes.unicode\_minus'] = False

plt.figure(figsize=(6, 5), facecolor="lightblue")

num = len(arg\_list) + 1

data = self.getvalues(arg\_1)

labels = [''] \* num

new\_data = []

colors = ["red"] \* num

explode = [0] \* num

explode[0] = 0.1 # 突出显示一下最低分数段的人

# 这个循环是为了统计出各分数段的人数

for i in range(len(arg\_list)):

if i == 0:

new\_data.append(np.count\_nonzero(data < arg\_list[i]))

else:

low = (np.count\_nonzero(data < arg\_list[i - 1]))

hign = (np.count\_nonzero(data < arg\_list[i]))

new\_data.append(hign - low)

new\_data.append(np.count\_nonzero(data >= arg\_list[len(arg\_list) - 1]))

# 各个扇形的颜色划定

for i in range(num):

colors[i] = colors\_all[i % 10]

# 下面的部分是为了标明各个扇形的标签

for i in range(num):

if i == 0:

labels[i] = str(0) + '-' + str(arg\_list[0]) + '分'

elif i == num - 1:

if arg\_list[i - 1] == 100:

labels[i] = '100分'

else:

labels[i] = str(arg\_list[i - 1]) + '-100分'

else:

labels[i] = str(arg\_list[i - 1]) + '-' + str(arg\_list[i]) + '分'

plt.title(arg\_1 + "各分数段人数统计", size=15)

plt.title("参与统计总人数为:" + str(self.len\_of\_DictDatabase), y=0, size=12, loc='left')

plt.pie(new\_data, labels=labels, explode=explode, colors=colors, autopct='%1.1f%%', startangle=90,

counterclock=False)

plt.show()

def plotPoints3D(self, name1, name2, name3, name4=''):

# 函数说明

"""

绘制三维图

:param name1:x轴属性

:param name2:y轴属性

:param name3:z轴属性

:param name4:可选参数，决定前三个参数是否要根据第四个元素来涂色

"""

# 创建Axes3D主要有两种方式，一种是利用关键字projection='3d'来实现(此处采用了这种做法）

# 另一种则是通过从mpl\_toolkits.mplot3d导入对象Axes3D来实现，目的都是生成具有三维格式的对象Axes3D

# 设置图像的大小，背景色，边框等属性

plt.rcParams['font.sans-serif'] = ['SimHei']

plt.rcParams['axes.unicode\_minus'] = False

fig = plt.figure(figsize=(6, 5), facecolor="lightblue", edgecolor="black", frameon=True)

rect = [0, 0.1, 0.9, 0.8]

ax = plt.axes(rect, projection='3d')

X = np.array(self.getvalues(name1).tolist()) # tolist()将矩阵转化为列表

Y = np.array(self.getvalues(name2).tolist())

Z = np.array(self.getvalues(name3).tolist())

Q = self.getvalues(name4).tolist()

if name4 == "":

ax.scatter3D(X, Y, Z)

else:

im=ax.scatter3D(X, Y, Z, c=Q, cmap="rainbow", marker='o')

fig.colorbar(im, format=matplotlib.ticker.FuncFormatter(lambda x, pos: int(x )),fraction= 0.05,pad=0.11)

# 自己添加的部分，指是否画出折线、曲面图

# axis.plot3D(X,Y,Z,'gray') # 画出空间折线

# X\_1,Y\_1=np.meshgrid(X, Y)

# Z\_1 = X\*0+Y\*0+Z

# ax.plot\_surface(X\_1,Y\_1,np.expand\_dims(Z\_1, axis=0), cmap="rainbow") # 需要将z扩展成二维

plt.title(name1 + '-' + name2 + '-' + name3 + "三维图", size=15)

plt.xlabel(name1)

plt.ylabel(name2)

ax.set\_zlabel(name3)

plt.show()

# 菜单函数

def menu():

print("""

1.增加元素

2.删除元素

3.更新元素

4.替换元素

5.查找元素

6.元素个数

7.存储到文件

8.读取文件

a.getvalues

b.getStat

x.绘制折线图

y.绘制柱形图

z.绘制饼图

d.绘制3D图

q.显示当前的数据库

0.退出

""")

ddb = DictDatabase()

while 1:

print("请选择需要的功能：")

prime = input()

if prime == '0':

break

elif prime == '1':

print("请输入增加的元素（以字典形式，为方便处理，请把学号以数字输入，不要带引号")

ddb.add(json.loads(input()))

elif prime == '2':

print("请输入要删除的元素（某个键及其的值）")

ddb.delete(input(), input())

elif prime == '3':

print("请输入要更新的元素，前两个参数为某个键及其的值，第三个参数请以字典的形式输入，为要更新的内容")

ddb.update(input(), input(), json.loads(input()))

elif prime == '4':

print("请输入要替换的元素，前两个参数为某个键及其的值，第三个参数请以字典的形式输入，为要替换的元素")

ddb.replace(input(), input(), json.loads(input()))

elif prime == '5':

print("请输入要查找的元素，两个参数为某个键及其的值")

a=input()

b=input()

print(a+"为"+b+"的人的信息为：")

ddb.find(a, b)

elif prime == '6':

print("元素的总个数为：", ddb.count())

elif prime == '7':

print("请输入保存的文件名")

ddb.save(input())

elif prime == '8':

print("请输入文件名称")

ddb.load(input())

elif prime == 'a':

print("请输入需要统计的属性（以及orderby）")

ddb.getvalues(input(), input()) # 此处注意，如果只输入一个参数时候，第二个多敲一下回车

elif prime == 'b':

print("请输入需要查看的属性")

a = input()

print(a+"的最大值、最小值、平均值为：")

print(ddb.get\_stat(a))

elif prime == 'x':

print("请输入横轴和纵轴对应的属性")

ddb.plotLine(input(), input())

elif prime == 'y':

print("请输入对应的属性")

ddb.plotBars(input(), input())

elif prime == 'z':

print("请输入需要统计的属性以及分数区间分隔点，分隔点请以列表形式输入")

ddb.plotPie(input(), eval(input()))

elif prime == 'd':

print("请依次输入三个轴的属性（以及第四个属性）")

ddb.plotPoints3D(input(), input(), input(), input())

elif prime == 'q':

print("当前的数据库为：")

ddb.show()

# 测试所有的功能

def trial():

ddb = DictDatabase()

ddb.load("grade.csv") # 导入数据

print('成绩列表如下')

ddb.show() # 显示一下目前的数据库

ddb.add({'id': 20190017, 'name': '旺财', 'age': 15, '高数': 99,

'线代': 90, '模电': 72, '概率论': 77,

'马原': 93, '离散': 50, '数据结构': 59}) # 增加元素

print('增添了一位同学')

ddb.show() # 显示一下目前的数据库

ddb.delete('name', '张三') # 删除元素

print('张三同学转专业了，删除掉')

ddb.show() # 显示一下目前的数据库

print('修改了李华同学的成绩')

ddb.update('name', '李华', {'数据结构': 89}) # 更新元素

ddb.show() # 显示一下目前的数据库

print("高数满分的同学的信息为：")

ddb.find('高数', 100) # 查找高数满分的人

print("元素个数为：")

print(ddb.count()) # 统计元素个数

print("将学号为007的同学换成了023的")

# 此处特别特别注意，从csv文件读取出来的学号，而自己输入的是字符串类型（有引号的）这可能会导致寻找不到的问题，

# 所以为了不出现bug，全改成数字类型

ddb.replace('id', 20190007, {'id': 20190023,

'name': '李龙', 'age': 23, '高数': 73, '线代': 80,

'模电': 75, '概率论': 82, '马原': 90,

'离散': 74, '数据结构': 78}) # 替换元素

ddb.show()

ddb.save("update.csv") # 将修改后的数据库存到新文件中

print("模电成绩如下：")

print(ddb.getvalues("模电")) # 打印出所有模电的成绩

print("根据数据结构排的模电成绩")

print(ddb.getvalues("模电", "数据结构")) # 根据数据结构排的模电成绩

print("离散的最大，最小，以及平均分")

print(ddb.get\_stat("离散")) # 返回离散的最大，最小，以及平均分

ddb.plotBars("name", "概率论") # 画出不同同学的概率论成绩的条形图

ddb.plotPie("线代", [60, 70, 80, 90]) # 画出线代的各个分数段的饼图，并突出显示不及格的

dds = DictDatabase()

dds.load("tianqi.csv")

dds.plotLine("日期", "温度") # 画出天气随温度变化的折线图

ddb.plotPoints3D("高数", "线代", "离散", "概率论") # 画出三维图，根据第四个元素赋予颜色

# 想要交互时可去掉菜单函数前面的注释符号

# menu()

trial()

**5.心得与体会**

通过本次课程大作业我认为我在其中收获了很多东西，也有一些心得与体会

5.1通过大作业的编写了解了很多库的作用，也掌握了一些库的使用方法，其中包括numpy库，主要是与数组有关，可以进行数据处理的库；pandas库，其中包含二维的类似于表格的数据结构dataframe，也同时与csv文件直接相关；matploblib的plot库：主要是用来进行可视化绘图的库。

5.2了解了很多与Python相关的知识，比如说在输入大量数据时，感觉到很麻烦，所以就考虑到了用表格文件与pandas库来处理，而即便用表格数据的话，如果是现成的数据自然是方便，如果是自己去编一些数据，输入还是很麻烦；这里就考虑如何可以获取数据，然后就了解到了Python的爬虫模块，也进行了初步的学习。由于本次的大作业是进行数据分析，所以也了解到了关于数据分析方面的一些知识，比如说数据处理的流程：数据获取、数据预处理，数据综合处理、得出结果（绘图只是其中的一种表现形式）；进行数据分析的软件或者是编程语言也很多，除Python外，还有Excel，matlab，sPss，R语言等。

5.3提升了代码编写能力，由于本学期的前一个学期没有进行编程的学习导致编程能力有所退步，通过对大作业的编写，又重新熟悉了一些基本的编程思路，以及重新熟悉了C++语言的一些语法（因为在编写Python程序时候会与C++语言混淆，所以顺便进行了复习）。

5.4对于Python的几种存储数据的类型进行了熟悉，Python的列表，元组，字典都是常见的存储数据的类型，这里通过编写大作业，对于这几种数据结构尤其是列表与字典的用法都有了进一步的了解。

5.5对于如何去写一个函数有了更多的认知，由于本次大作业中使用的是类，而其中有很多类函数的编写。对于一个函数来说，应该提高它的效率，包括时间复杂度和空间复杂度，这是算法应该考虑的问题；还应该去提高的它的可用性，比如说它能适用于多种情况，这就要涉及到Python的函数参数问题了，在很多次调用函数的时候，尤其是在绘图部分，发现很多Python函数都有非常多的参数，一开始的时候也对这些参数表示头疼，所以去了解了Python参数的类型。其中位置参数就是最基本的参数；默认值参数就是指这个参数已经有了默认参数，传值是可以不传，通过这种方式就可以实现参数个数可变的情况；关键参数是指实参顺序和形参顺序不一样的参数，示例如下：

def showArgs(a, b, c):

print(a, b, c)

showArgs(1, 2, c=3) # 1 2 3

showArgs(a=1, c=22, b=33) # 1 33 22

而第四种参数是可变长度的参数，即其参数的长度不固定，其有两种形式，示例如下。

def showArgs(\*args):

print(args)

showArgs(1, 2, 3) # (1, 2, 3)

showArgs(7, 8) # (7, 8)

这是第一种情况，其接受的是位置参数

def showArgs(\*\*args):

print(args)

showArgs(a=1, b=2, c=3) # {'a': 1, 'b': 2, 'c': 3}

showArgs(a=7, b=8) # {'a': 7, 'b': 8}

这是第二种情况，其接受的参数必须是关键参数。对于第四种参数来说，经常见于Python的各类库函数中。

5.6通过完成大作业，体会到了Python语言和之前学习过的C语言或者是C++语言的差异，Python的代码量是要远远少于其他语言的，首先它的语法更为简单，逻辑也更为简单，并且有很多功能强大的库，里面有非常多的函数，每一个函数适用的情况也很多，这样使它成为了一种简单而又好用的语言，当然这些好处的代价是其运行速度比较慢，在调试时候就可以明显感觉出来；还有就是Python语言对于缩进的要求比较严格，因为它要用严格的缩进来代替省略的括号等符号（与C语言相比）。