

第5课 变形金刚战队——列表和字典

课程目标

1. 熟练掌握列表、字典中元素的增删改查
2. 理解列表和字典的区别

课程难点

1. 列表与字典增删改查的异同
2. 正确使用切片，深刻理解切片时冒号左右数字的意义

课程知识点总结

一、列表

1. 代码格式

数据存储在**中括号[]**里，用逗号隔开并使用等号赋值给列表。中括号里面的每一个数据叫作“元素”。

列表中的元素是有自己明确的“位置”的，元素相同，在列表中排列顺序不同，就是两个不同的列表。

列表中字符串、整数、浮点数都可以存储。

```
1 list = ['李雷','韩梅梅',180,3.5]
```

1. 提取元素

1) 下标。每一个元素都有自己的位置标号，这个位置标号就叫做下标。

- 下标从0，1，2开始逐渐递增
- 列表名后加带下标的中括号，就能取到相应位置的元素。
- 结果是一个元素

2) 切片：用冒号来截取列表元素的操作。

- 冒号左边空（或者为0），就要从下标为0的元素开始取
- 右边空（或者为0），就要取到列表的最后一个元素
- 冒号左右都有数字时，[A: B]，表示从下标为A的元素开始取，取到下标为B的前一个元素的值。
- 冒号左边数字对应的元素要拿，右边的不动
- 切片截取了列表的一部分，所以得到的结果仍然是一个列表。（即使只有一个元素，也是在列表里的，要与用下标取单个元素的方法区别开）

```

1 transformers = ['擎天柱','大黄蜂','救护车']
2 print(transformers[1])
3 #使用下标提取单一元素，结果显示：
4 #大黄蜂
5 print(transformers[:])
6 #['擎天柱','大黄蜂','救护车']
7 print(transformers[2:])
8 #使用切片，即使结果仍然只有一个元素，但显示为列表：
9 #['救护车']

```

3) 特别地，a,b,c=students，也可以提取出列表中的元素，变量分别用逗号隔开，且变量的数量与列表元素个数一致，最终列表元素会分别赋值给变量，例如：

```

1 name = ['李雷','lily','lucy']
2 a,b,c=name
3 print(a)
4 print(b)
5 print(c)
6 print(a,b,c)
7 #结果显示为
8 #李雷
9 #lily
10 #lucy
11 #李雷 lily lucy

```

1. 增加/删除元素

1) 增加元素

列表名.append()。注意：这里是 . 不是空格！

append后的括号里只能接受一个参数，结果是让列表末尾新增一个元素。列表长度可变，理论容量无限，所以支持任意的嵌套。

```

1 transformers = ['擎天柱','大黄蜂','救护车']
2 #添加'萨克巨人'这个元素
3 transformers.append('萨克巨人')
4 print(transformers)
5 #结果显示：['擎天柱','大黄蜂','救护车','萨克巨人']
6 transformers.append([4, 5])
7 print(transformers)
8 #添加'[4, 5]'这个列表，也就是append()的参数为一个列表，也是一个参数，所以不会报错
9 #['擎天柱','大黄蜂','救护车','巨无霸福特','红蜘蛛',[4, 5]]
10 transformers.append(4, 5)
11 #代码会出现报错

```

因为给了两个元素（没有作为一个整体，所以算两个参数）。注意！！千万不能：
a=transformers.append(3)，这样a里只有none。

2) 删除元素

del 列表名[元素的下标]。注意这里是空格不是.了！

与append()函数类似，能删除单个元素、多个元素（切片）、整个列表。

3) 修改元素

使用下标修改对应位置的元素。

```

1 transformers = ['擎天柱','大黄蜂','救护车','巨无霸福特','红蜘蛛']
2 #增加:
3 transformers.append('凤凰女')
4 print(transformers)
5 #['擎天柱', '大黄蜂', '救护车','巨无霸福特','红蜘蛛', '凤凰女']
6 #删除:
7 del transformers[2]
8 #结果是:
9 print(transformers)
10 #['擎天柱', '大黄蜂', '巨无霸福特', '红蜘蛛','凤凰女']
11 #修改“
12 transformers[3] = '灭霸'
13 print(transformers)
14 #['擎天柱', '大黄蜂', '巨无霸福特', '灭霸', '凤凰女']

```

二、字典

字典所存储的两种数据若存在**一一对应**的情况，用字典储存会更加方便。唯一的键和对应的值形成的整体，我们就叫做【键值对】。键具备唯一性，而值可重复。

1. 代码格式 {键: 值}

- 字典外层是**大括号{}**，用等号赋值；
- 列表中的元素是自成一体的，而字典的元素是由**键值对**构成的，用英文冒号连接。有多少个键值对就有多少个元素。如 '擎天柱': 95，其中我们把'擎天柱'叫**键 (key)**，95叫**值 (value)**。
- 键值对间用逗号隔开

字典中数据是随机排列的，调动顺序也不影响。所以列表有序，要用下标定位；字典无序，便通过唯一的键来定位。

Tip: len()函数用于获取数据长度

```

1 transformers = ['擎天柱','大黄蜂','救护车','巨无霸福特','红蜘蛛']
2 fc = { '擎天柱': 95 , '大黄蜂':90 , '救护车':86, '巨无霸福特':80}
3 print(len(transformers))
4 print(len(fc))
5 #结果显示
6 #5
7 #4
8 #字典的元素个数，数冒号就行了

```

1. 提取元素

字典没有下标，所以在提取元素时，中括号中应该写键的名称，即字典名[字典的键]。提取出来的是**key对应的value**，而不会显示键的数据！

```

1 fc = { '擎天柱': 95 , '大黄蜂':90 , '救护车':86, '巨无霸福特':80 }
2 print(fc['擎天柱'])
3 #结果显示
4 #95

```

2. 增加/删除元素、

1) 新增元素

字典名[键] = 值。每次只能新增一个键值对。fc['红蜘蛛','灭霸']=92,85, 这样是不对的, 最终会输出('红蜘蛛','灭霸':(92,85))作为一个键值对。

```
1 fc = { '擎天柱': 95 , '大黄蜂':90 , '救护车':86, '巨无霸福特':80 }
2 fc['铁皮'] = 90
3 print(fc)
4 print(fc['救护车'])
5 #结果显示
6 #{'擎天柱': 95, '大黄蜂': 90, '救护车': 86, '巨无霸福特': 80, '铁皮': 90}
7 #86
```

2) 删除元素

del 字典名[键]

```
1 fc = { '擎天柱': 95 , '巨无霸福特':80, '红蜘蛛':80 }
2 del fc['巨无霸福特']
3 print(fc)
4 #结果显示
5 #{'擎天柱': 95 , '红蜘蛛':80 }
```

3) 修改元素

如果不是整个键值对都不需要, 只需要改变对应key下的value, 修改就可以, 不需要del。

```
1 fc = { '擎天柱': 95 , '巨无霸福特':80, '红蜘蛛':80}
2 fc['擎天柱'] = '100'
3 print(fc)
4 #字典没有下标, 只能通过key找到元素位置
5 #{'擎天柱': '100', '巨无霸福特': 80, '红蜘蛛': 80}
```

三、列表与字典的异同

1. 不同点:

- 1)、列表外面是[]中括号, 字典外面是{}大括号。
- 2)、列表中每个元素只有一部分, 只有值, 每个值有一个对应的小标。
字典中每个元素都是【键: 值】存在的, 每个值都有一个对应的键。

2. 相同点:

- 1)、列表与字典中的元素都是用逗号隔开。
- 2)、删除的方法del相同。