

主讲人：刘潇潇

主讲人：刘潇潇



学习目标



- 掌握什么是列表，以及列表的常见操作
- 掌握列表的嵌套使用
- 掌握字典的常见操作以及字典的遍历
- 掌握元组的基本使用



CONTENTS

- 1 列表概述
- 2 列表的常见操作
- 3 元组
- 4 字典

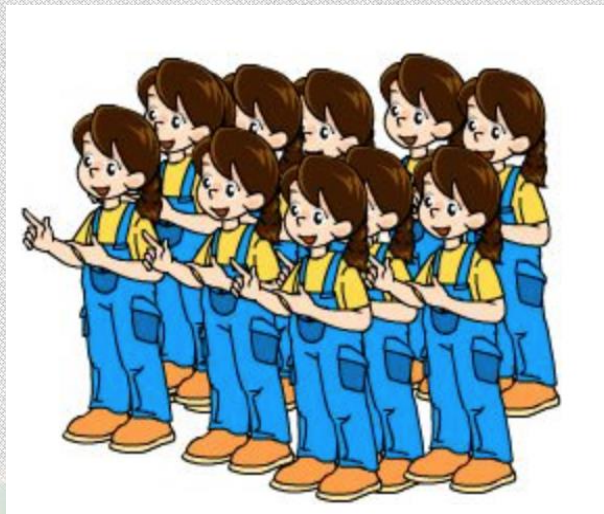




1. 列表概述

什么是 列表?

J18019有41位学生，在存储和使用学生信息数据时，如果每个变量存放一个学生的姓名，是不是很麻烦？如果更多学生，那该怎么办呢？



列表(list)是Python中的一种数据结构，它可以存储不同类型的数据。

```
list_1 = [1,2,3,'happy','@#$%']
```


列表索引是从**0**开始的，我们可以通过下标索引的方式来访问列表中的值。

```
list_1 = [1,2,3,'happy','@#$%']  
print(A[0])
```



请思考

列表中的元素可以是一个列表吗？



列表的嵌套

列表的嵌套指的是一个列表的元素又是一个列表。



列表的嵌套

```
schoolNames = [['北京大学', '清华大学'],  
                ['南开大学', '天津大学', '天津师范大学'],  
                ['山东大学', '中国海洋大学']]
```


练习:

创建一个列表dormitories，用于存放3个男生宿舍的2个人名、与3个女生宿舍的2个人名，要求用嵌套列表，区分男生、女生、宿舍号





2. 列表常见操作

列表的循环遍历

1. 使用for循环遍历列表

```
universities = ['清华', '北大', 'Stanford']  
for uni in universities:  
    print(uni)
```



列表的循环遍历

2. 使用while循环遍历列表

```
universities = ['清华', '北大', 'Stanford']  
length = len(universities)  
i = 0  
while i < length:  
    print(universities[i])  
    i += 1
```


在列表中增加元素



在列表中增加元素的方式有多种：

- 通过**append**（添加、追加）可以向列表添加元素
- 通过**extend**（扩展）可以将另一个列表的元素添加到列表中。
- 通过**insert**（插入）在指定位置index前插入元素object。



在列表中增加元素

1. 通过append向列表添加元素

```
universities = ['清华', '北大', 'Stanford']  
print(universities)  
universities.append('复旦大学')  
print(universities)
```

Lesson1

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled/Lesson1.py  
['清华', '北大', 'Stanford']  
['清华', '北大', 'Stanford', '复旦大学']
```

进程已结束, 退出代码0

在列表中增加元素

2.通过extend将另一个列表的元素添加到列表中

```
universities = ['清华', '北大', 'Stanford']  
college = ['山东电子', '山东商职']  
universities.extend(college)  
print(universities)
```

Lesson1

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled/Lesson1.py  
['清华', '北大', 'Stanford', '山东电子', '山东商职']
```

进程已结束, 退出代码0

在列表中增加元素

3. 通过insert在指定位置index前插入元素object



```
universities = ['清华', '北大', 'Stanford']  
universities.insert(2, '复旦大学')  
print(universities)
```

son1

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled/Lesson1.py  
['清华', '北大', '复旦大学', 'Stanford']
```

进程已结束, 退出代码0



练习：

使用while循环，从客户端接收5个人名（input函数）。将5个人名保存在一个列表names中（append函数）。使用for循环遍历names列表并打印。



练习:

修改names列表，在第3个人名前插入
‘张三’。在列表最后加入另一个人名
列表[‘李四’ , ’ 王五’]



在列表中查找元素



在列表中查找元素的方法包括：

- in（存在）,如果存在那么结果为true，否则为false。
- not in（不存在）， 如果不存在那么结果为true，否则false。



在列表中查找元素

1. 通过in查找列表中是否有“复旦”

```
universities = ['清华', '北大', 'Stanford']  
if '复旦' in universities:  
    print('Yes')  
else:  
    print('No')
```

No

在列表中查找元素

2. 通过not in查找列表中是否有“复旦”

=

```
universities = ['清华', '北大', 'Stanford']  
if '复旦' not in universities:  
    print('Yes')  
else:  
    print('No')
```

Yes

练习：

查找dormitories列表，如果‘张龙俊’在列表中，则打印“张龙俊是我的同学”，不在则打印“我不认识张龙俊”。有几种方法可以实现？



在列表中修改元素



列表元素的修改，是通过**下标**来实现的。



在列表中修改元素

将列表中第二个元素改成 “复旦”

=

```
universities = ['清华', '北大', 'Stanford']  
universities[2] = '复旦'  
print(universities)
```

```
['清华', '北大', '复旦']
```



在列表中删除元素



列表元素的常用删除方法有三种，具体如下：

- del: 根据下标进行删除
- pop: 删除最后一个元素
- remove: 根据元素的值进行删除



在列表中删除元素

用del删掉列表首个元素

```
universities = ['清华', '北大', 'Stanford']  
print(universities)  
del universities[0]  
print(universities)
```

```
['清华', '北大', 'Stanford']  
['北大', 'Stanford']
```



在列表中删除元素

用pop删掉列表最后一个元素



```
universities = ['清华', '北大', 'Stanford']  
universities.pop  
print(universities)
```

```
['清华', '北大', 'Stanford']
```



在列表中删除元素

用remove删掉列表中值为“北大”的元素

=

```
universities = ['清华', '北大', 'Stanford']  
universities.remove('北大')  
print(universities)
```

```
['清华', 'Stanford']
```


列表的排序操作



列表的排序可以通过下面两个方法实现：

- sort方法：列表的元素按照特定顺序排列。
- reverse方法：将列表逆置。



列表的排序操作

用sort对列表进行排序



```
numbers = [3, 5, 7, 12, 65, -2]  
numbers.sort()  
print(numbers)
```

[-2, 3, 5, 7, 12, 65]

```
numbers = [3, 5, 7, 12, 65, -2]  
numbers.sort(reverse = True)  
print(numbers)
```

[65, 12, 7, 5, 3, -2]



列表的排序操作

用reverse对列表进行反向转置



```
universities = ['清华', '北大', 'Stanford']  
universities.reverse()  
print(universities)
```

```
['Stanford', '北大', '清华']
```



创建数字列表

使用range()函数可以创建数字列表

```
numbers = list(range(1,6))  
print(numbers)
```

test

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
[1, 2, 3, 4, 5]
```

进程已结束,退出代码0



创建数字列表

使用range()函数可以创建数字列表



```
numbers = list(range(1,11,2))  
print(numbers)
```

it

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
[1, 3, 5, 7, 9]
```

进程已结束, 退出代码0



列表统计

最大值、最小值、总和

```
numbers = list(range(1, 50, 2))  
print(numbers)  
print(min(numbers))  
print(max(numbers))  
print(sum(numbers))
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py
```

```
[1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49]
```

```
1
```

```
49
```

```
625
```

```
进程已结束, 退出代码0
```


列表切片

使用遍历列表



```
players = ['charles', 'martina', 'michael', 'florence', 'eli']  
for ply in players[:3]:  
    print(ply.title())
```

st

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
Charles  
Martina  
Michael
```

进程已结束, 退出代码0



列表切片

遍历列表

```
players = ['charles', 'martina', 'michael', 'florence', 'eli']  
for ply in players[:3]:  
    print(ply.title())
```

st

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py
```

```
Charles
```

```
Martina
```

```
Michael
```

```
进程已结束, 退出代码0
```


列表复制

复制列表



```
players_1 = ['charles', 'martina', 'michael', 'florence', 'eli']  
players_2 = players_1[0:3]  
print(players_2)
```

st

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
['charles', 'martina', 'michael']
```

进程已结束, 退出代码0



3. 元组



什么是元组 (tuple)

Python的元组与列表类似，不同之处在于元组的元素不能修改。元组使用小括号，列表使用方括号。



访问元组

元组可以使用下标索引来访问元组中的值



```
players_1 = ('charles', 'martina', 'michael', 'florence', 'eli')  
print(players_1)
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
( 'charles', 'martina', 'michael', 'florence', 'eli' )
```

进程已结束, 退出代码0



访问元组

元组的内置函数



方法	描述
<code>len(tuple)</code>	计算元组元素个数
<code>max(tuple)</code>	返回元组中元素最大值
<code>min(tuple)</code>	返回元组中元素最小值
<code>tuple(seq)</code>	将列表转为元组

4.字典



什么是字典 (dict)

字典是一种存储数据的容器，它和列表一样，都可以存储多个数据。每个元素都是由两部分组成的，分别是键和值。



创建字典

‘name’ 为键， ‘班长’ 为值。



```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(info)
```

t

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
{'name': '班长', 'sex': '男', 'address': '济南'}
```

进程已结束, 退出代码0



练习

创建一个字典info，用于存放一位同学的所有信息：



姓名	刘文
性别	女
班级	J17010
语文	38
数学	83
英语	96
宿舍号	1111
舍友	张芳、李玉、范甜甜



访问字典

根据键访问值



```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(info['name'])  
print(info['address'])
```

E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py

班长
济南



字典常用操作

添加字典的元素

```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
new_id = input('请输入新学号: ')  
info['id'] = new_id  
print(info)
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
请输入新学号: J1800101  
{'name': '班长', 'sex': '男', 'address': '济南', 'id': 'J1800101'}  
  
进程已结束,退出代码0
```


字典常用操作

修改字典的元素

```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
new_add = input('请输入新地址: ')  
info['address'] = new_add  
print(info)
```

E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py

请输入新地址: 天津

{'name': '班长', 'sex': '男', 'address': '天津'}

进程已结束, 退出代码0



字典常用操作

删除字典元素



➤ **del**: 用于删除字典；删除后，字典完全不存在了，无法再根据键访问字典的值。

➤ **clear**: 只是清空字典中的数据，字典还存在，只不过没有元素。



字典常用操作

删除字典元素



```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
del info['name']  
print(info)
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
{'sex': '男', 'address': '济南'}
```

进程已结束, 退出代码0



字典常用操作

删除字典元素

```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
info.clear()  
print(info)
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
{}
```

进程已结束, 退出代码0

字典常用操作

计算字典中键值对的个数

=

```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(len(info))
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
3
```

进程已结束, 退出代码0



字典常用操作

. 获取字典中键的列表



```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(info.keys()) #获取字典中所有键  
print(type(info.keys())) #查看键的数据类型  
print(list(info.keys())[0]) #将字典的键转化为列表，并访问其中元素
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
dict_keys(['name', 'sex', 'address'])  
<class 'dict_keys'>  
name
```

进程已结束, 退出代码0



字典常用操作

. 获取字典中值的列表



```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(info.values())  
print(type(info.values()))
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
dict_values(['班长', '男', '济南'])  
<class 'dict_values'>
```




字典常用操作

items()方法返回字典的(键, 值)元组对的列表

```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(info.items()) #查看键值对  
print(type(info.items())) #查看键值对的数据类型  
for item in info.items(): # 遍历items  
    print(item) #查看每个键值对  
    print(type(item)) #注意这里的数据类型是元组
```

```
t  
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
dict_items([('name', '班长'), ('sex', '男'), ('address', '济南')])  
<class 'dict_items'>  
( 'name', '班长')  
<class 'tuple'>  
( 'sex', '男')  
<class 'tuple'>  
( 'address', '济南')  
<class 'tuple'>
```



字典常用操作

遍历字典的键值对

```
info = {'name': '班长', 'sex': '男', 'address': '济南'}  
print(info.items()) #查看键值对  
print(type(info.items())) #查看键值对的数据类型  
for key, value in info.items(): # 遍历items  
    print('key:', key)  
    print('value', value)
```

```
E:\Python\python.exe C:/Users/Administrator/PycharmProjects/untitled3/test.py  
dict_items([('name', '班长'), ('sex', '男'), ('address', '济南')])  
<class 'dict_items'>  
key: name  
value 班长  
key: sex  
value 男  
key: address  
value 济南
```



小节

本章主要介绍了列表、元组和字典三种类型，希望大家通过本章的学习，能够清楚的知道这三种类型各自的特点，这样在后续开发过程中，可以选择合适的类型对数据进行操作。



练习

获取 'J17010' , 和 '李玉'

=





感谢聆听
Thank You

