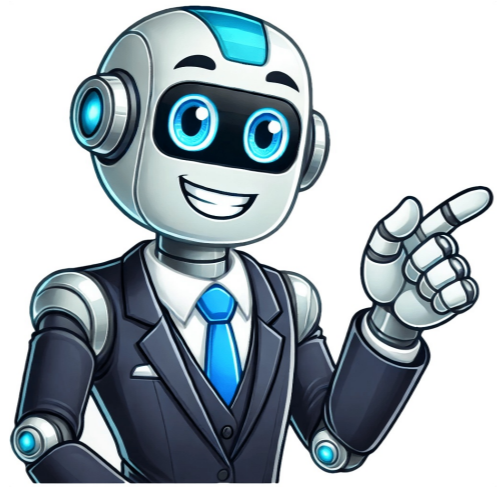


I'm not a bot



Python3 实例 在 Python 中，列表切片是一种非常强大的功能，它允许你从一个列表中提取一部分元素，形成一个新的子列表。

切片操作使用方括号 [] 和冒号 : 来指定起始索引、结束索引和步长。假设我们有一个列表 my_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]，我们想要提取其中的一部分元素，可以使用切片操作。 my_list = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9] # 提取索引 2 到 5 的元素 (不包括索引 5)
sub_list = my_list[2:5]
print(sub_list)
代码解析: my_list[2:5]: 这是一个切片操作，2 是起始索引，5 是结束索引。切片操作会提取从索引 2 开始到索引 5 之前的元素 (即不包括索引 5 的元素)。 sub_list: 这是切片操作后得到的新列表。
输出结果: [2, 3, 4]
Python3 实例 >>> li = ["a", "b", "mpilgrim", "z", "example"]
>>> li[1]
'b'
>>> li["a", "b", "mpilgrim", "z", "example"]
>>> li[-1]
'example'
>>> li[-3]
'mpilgrim'
>>> li["a", "b", "mpilgrim", "z", "example"]
>>> li[1:3]
['b', 'mpilgrim']
>>> li[1:-1]
['b', 'mpilgrim', 'z']
>>> li[0:3]
['a', 'b', 'mpilgrim']
序列是Python中最基本的数据结构。序列中的每个元素都分配一个数字 - 它的位置，或索引，第一个索引是0，第二个索引是1，依此类推。Python有6个序列的内置类型，但最常见的是列表和元组。序列都可以进行的操作包括索引，切片，加，乘，检查成员。此外，Python已经内置确定序列的长度以及确定最大和最小的元素的方法。列表是最常用的Python数据类型，它可以作为一个方括号内的逗号分隔值出现。列表的数据项不需要具有相同的类型 创建一个列表，只要把逗号分隔的不同的数据项使用方括号括起来即可。如下所示： list1 = ['physics', 'chemistry', 1997, 2000]
list2 = [1, 2, 3, 4, 5]
list3 = ["a", "b", "c", "d"]
与字符串的索引一样，列表索引从0开始。列表可以进行截取、组合等。 访问列表中的值 使用下标索引来访问列表中的值，同样你也可以使用方括号的形式截取字符，如下所示： list1 = ['physics', 'chemistry', 1997, 2000]
list2 = [1, 2, 3, 4, 5, 6, 7]
print "list1[0]: ", list1[0]
print "list2[1:5]: ", list2[1:5]
以上实例输出结果： list1[0]: physics
list2[1:5]: [2, 3, 4, 5]
更新列表 你可以对列表的数据项进行修改或更新，你也可以使用append()方法来添加列表项，如下所示： list = []
list.append('Google')
list.append('Runoob')
print list
注意：我们会在接下来的章节讨论append()方法的使用 以上实例输出结果： ['Google', 'Runoob']
删除列表元素 可以使用 del 语句来删除列表的元素，如下实例： list1 = ['physics', 'chemistry', 1997, 2000]
print list1
del list1[2]
print "After deleting value at index 2: "
print list1
以上实例输出结果： ['physics', 'chemistry', 1997, 2000]
After deleting value at index 2: ['physics', 'chemistry', 2000]
注意：我们会在接下来的章节讨论remove()方法的使用 Python列表脚本操作符 列表对 + 和 * 的操作符与字符串相似。+ 号用于组合列表，* 号用于重复列表。 如下所示： Python 表达式结果
描述 len([1, 2, 3])
3
长度 [1, 2, 3]
+ [4, 5, 6]
[1, 2, 3, 4, 5, 6]
组合 ['Hi!' * 4]
['Hi!', 'Hi!', 'Hi!', 'Hi!']
重复 3 in (1, 2, 3)
True
元素是否存在于列表中
for x in [1, 2, 3]:
print x
1
2
3
迭代 Python列表截取 Python的列表截取实例如下： >>>L = ['Google', 'Runoob', 'Taobao']
>>> L[2]
'Taobao'
>>> L[2:]
'Taobao'
>>> L[:2]
['Runoob', 'Taobao']
>>> 描述： Python 表达式结果
描述 L[2]'Taobao'读取列表中第三个元素
L[:2]'Runoob'读取列表中倒数第二个元素
L[1:]['Runoob', 'Taobao']从第二个元素开始截取列表
Python列表函数&方法 Python包含以下函数。Python包含以下方法: C++ 标准库提供了丰富的功能，其中 是一个非常重要的容器类，用于存储元素集合，支持双向迭代器。是 C++ 标准模板库 (STL) 中的一个序列容器，它允许在容器的任意位置快速插入和删除元素。与数组或向量 () 不同， 不需要在创建时指定大小，并且可以在任何位置添加或删除元素，而不需要重新分配内存。 语法 以下是 容器的一些基本操作： 包含头文件：#include 声明列表：std::list mylist，其中 T 是存储在列表中的元素类型。 插入元素：mylist.push_back(value); 删除元素：mylist.pop_back(); 或 mylist.erase(iterator); 访问元素：mylist.front(); 和 mylist.back(); 遍历列表：使用迭代器 for (auto it = mylist.begin(); it != mylist.end(); ++it) 特点 双向迭代： 提供了双向迭代器，可以向前和向后遍历元素。 动态大小：与数组不同， 的大小可以动态变化，不需要预先分配固定大小的内存。 快速插入和删除：可以在列表的任何位置快速插入或删除元素，而不需要像向量那样移动大量元素。 声明与初始化的声明和初始化与其他容器类似： #include #include int main() { std::list lst1; // 空的list
std::list lst2(5); // 包含5个默认初始化元素的list
std::list lst3(5, 10); // 包含5个元素，每个元素为10
std::list lst4 = {1, 2, 3, 4}; // 使用初始化列表
return 0; }
实例 下面是一个使用的简单示例，包括创建列表、添加元素、遍历列表和输出结果。 #include #include int main() { // 创建一个整数类型的列表
std::list numbers; // 向列表中添加元素
numbers.push_back(10);
numbers.push_back(20);
numbers.push_back(30); // 访问并打印列表的第一个元素
std::cout << L[2] << endl; // 列表还支持拼接操作： >>> squares = [1, 4, 9, 16, 25]
>>> squares += [36, 49, 64, 81, 100]
>>> squares
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
>>> 嵌套列表 使用嵌套列表即在列表里创建其它列表。例如： >>> a = ['a', 'b', 'c']
>>> n = [1, 2, 3]
>>> x = [a, n]
>>> x
[['a', 'b', 'c'], [1, 2, 3]]
>>> x[0][1]
'b'
>>> x[0][1]
'b'
列表比较 列表比较需要引入 operator 模块的 eq 方法 (详见: Python operator 模块)： # 导入 operator 模块
import operator
a = [1, 2]
b = [2, 3]
c = [2, 3]
print(operator.eq(a,b), operator.eq(a,b))
print(operator.eq(c,b), operator.eq(c,b))
以上代码输出结果为： operator.eq(a,b): False
operator.eq(c,b): True
Python列表函数&方法 Python包含以下函数: Python包含以下方法: Python 列表 描述 sort() 函数用于对列表进行排序，如果指定参数，则使用比较函数指定的比较函数。 语法 sort()方法语法： list.sort(cmp=None, key=None, reverse=False)
参数 cmp -- 可选参数，如果指定了该参数会使用该参数的方法进行排序。 key -- 主要是用来进行比较的元素，只有一个参数，具体的函数的参数就是取自于可迭代对象中，指定可迭代对象中的一个元素来进行排序。 reverse -- 排序规则，reverse = True 降序，reverse = False 升序 (默认)。 返回值 该方法没有返回值，但是会对列表的对象进行排序。 实例 以下实例展示了 sort() 函数的使用方法： aList = ['123', 'Google', 'Runoob', 'Taobao', 'Facebook']
aList.sort();
print("List : ")
print(aList)
以上实例输出结果如下： List : ['123', 'Facebook', 'Google', 'Runoob', 'Taobao']
以下实例降序输出列表： vowels = ['e', 'a', 'u', 'o', 'i']
vowels.sort(reverse=True)
print("降序输出:")
print(vowels)
以上实例输出结果如下： 降序输出: ['u', 'o', 'i', 'e', 'a']
以下实例演示了通过指定列表中的元素排序来输出列表： def takeSecond(elem):
return elem[1]
random = [(2, 2), (3, 4), (4, 1), (1, 3)]
random.sort(key=takeSecond)
print("排序列表:")
print(random)
以上实例输出结果如下： 排序列表： [(4, 1), (2, 2), (1, 3), (3, 4)]
Python 列表