Python 语言程序设计: 列表与元组

李宽
likuan@dgut.edu.cn

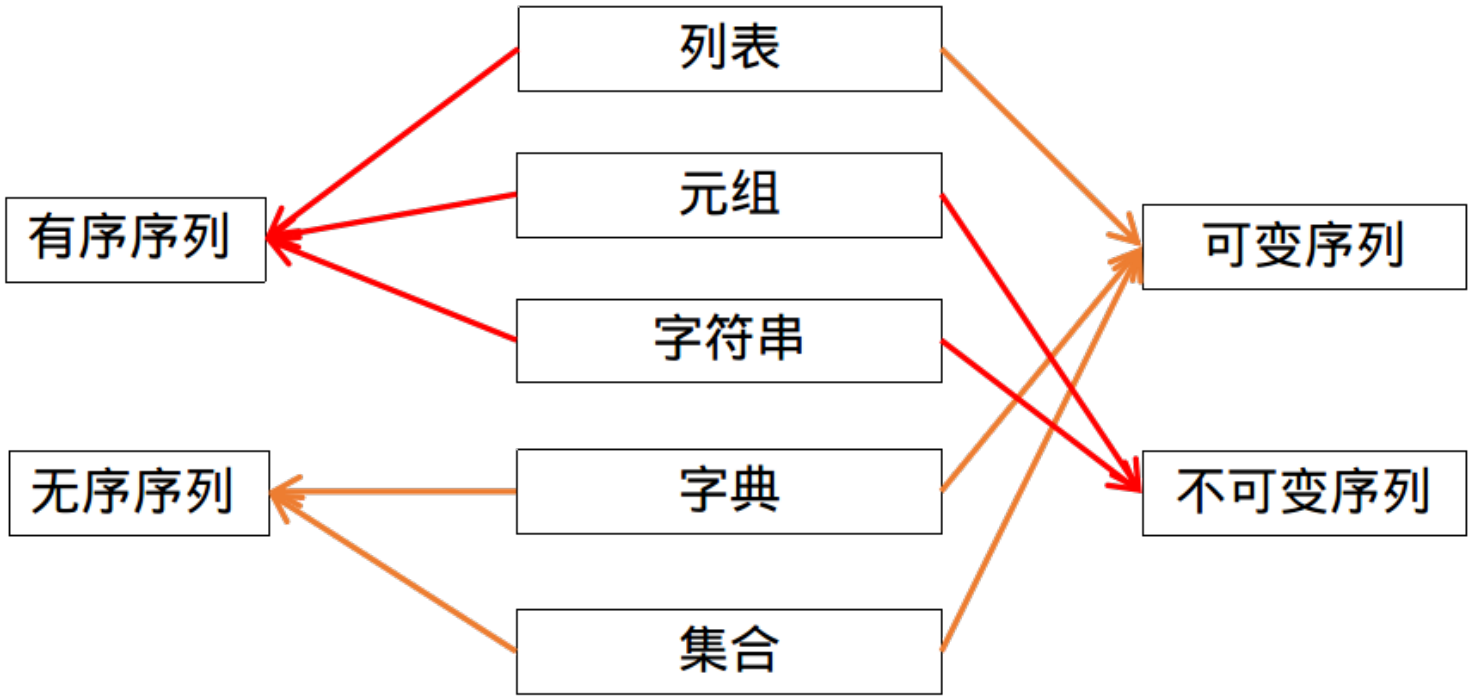
东莞理工学院

2019.10

目录

目录

Python 中的序列类型



关于字典和集合是否可归为序列存在争议
本课程沿用有序序列和无序序列的分类.

有序序列总览

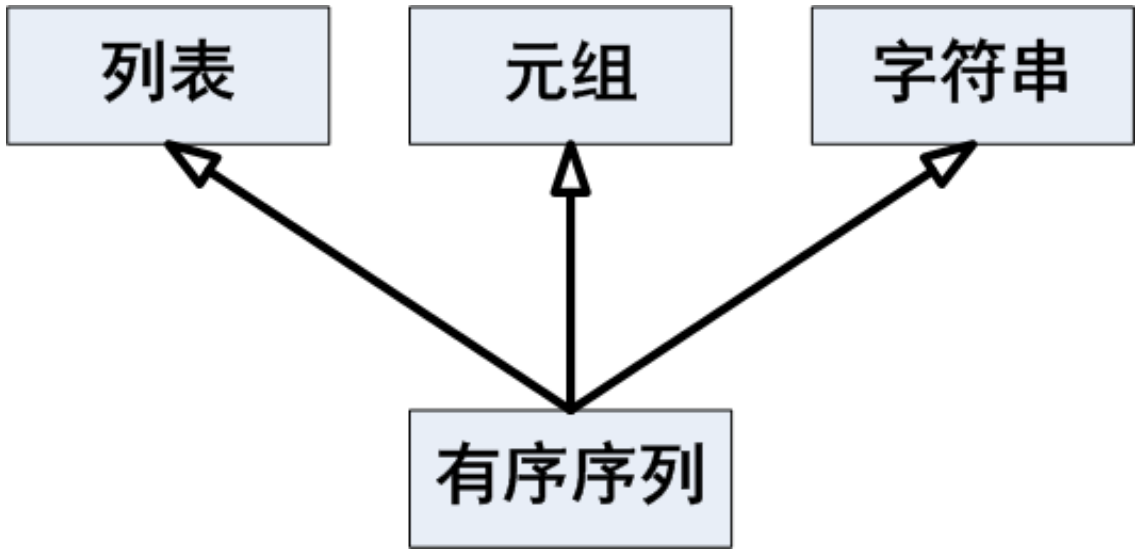
有序序列

有序序列是具有先后关系的一组元素

- 一维元素向量，元素类型可同可不同
- 元素间由序号引导，通过下标访问序列的特定元素

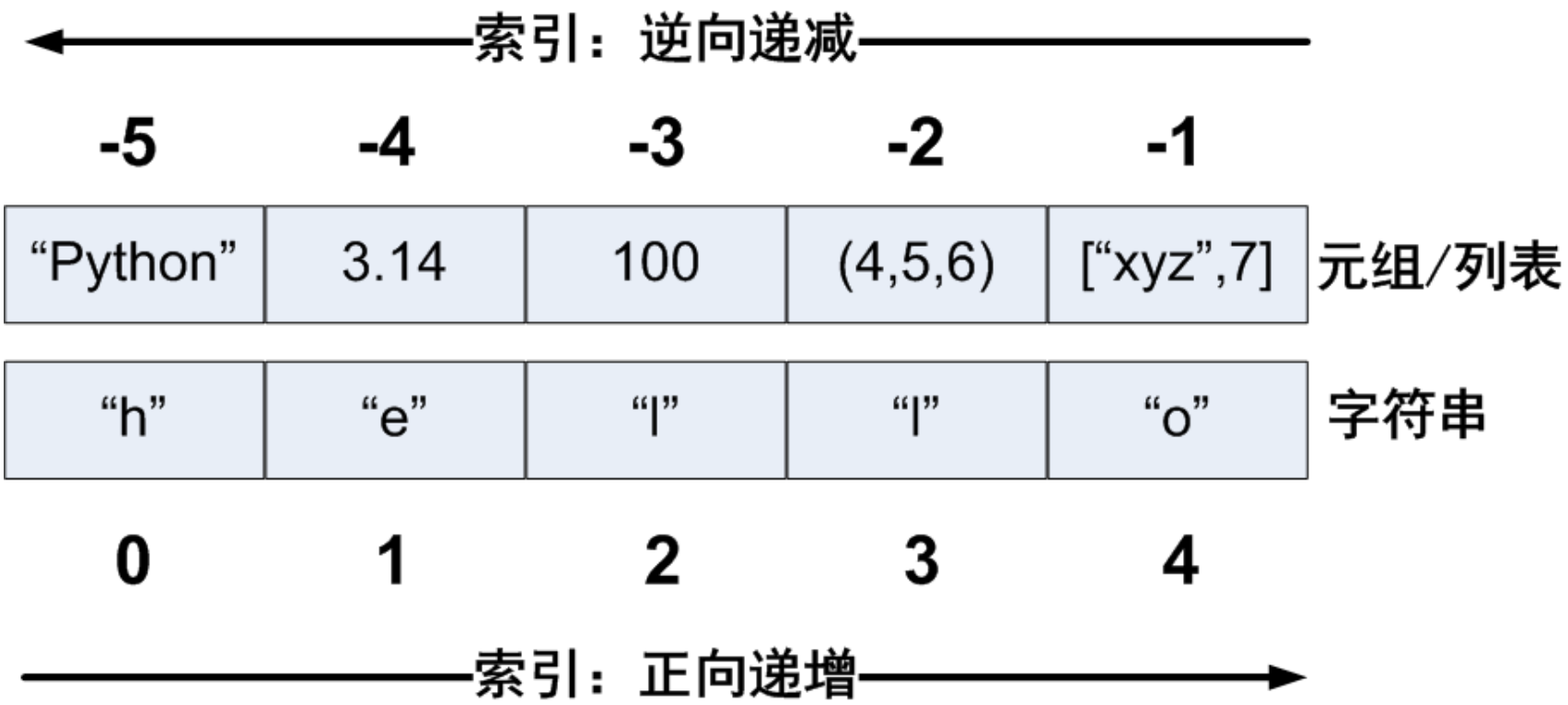
```
1  str1 = "hello,python"   # 字符串可看成一维字符向量，支持索引
2  list1 = ["xyz",1234, 78.9] # 列表，其中元素类型可不同
3  tup1 = ("xyz",1234, 78.9) # 元组，其中元素类型可不同
```

有序列表总览：派生关系



基类 (有序序列) 与派生类 (列表/元组/字符串) 的关系

有序序列总览：正逆向索引



有序序列总览：通用操作符

有序序列支持[1]的通用操作符 (6 个)

操作符及应用	描述
x in s	如果 x 是序列 s 的元素，返回 True, 否则, 返回 False
x not in s	如果 x 是序列 s 的元素，返回 False, 否则, 返回 True
s + t	拼接两个序列 s 和 t
s*n 或者 n*s	将序列 s 复制 n 次
s[i]	索引, 返回 s 中的第 i 个元素
s[i:j:k] 或者 s[i:j]	切片, 返回 s 中第 i 到 j 以 k 为步长的元素子序列

复习：字符串相关代码

[1]列表, 元组和字符串三个派生类别均支持

有序序列总览：方法和函数

函数和方法 (5 个)

函数和方法	描述
len(s)	返回序列 s 的长度, 即元素个数
min(s)	返回序列 s 的最小元素,s 中元素需要可比较
max(s)	返回序列 s 的最大元素,s 中元素需要可比较
s.index(x[,i,j])	返回序列 s 从索引 i 开始到索引 j中第一次出现x 的索引
s.count(x)	返回序列 s 中出现 x 的总次数

有序序列: 方法和函数示例

order-list-func.py:

```
# 有序序列支持的函数和方法

str1="hello,python"
list1=["hello",100, 56.78]

# 调用 len 函数，返回字符串和列表的长度（元素个数）
print("字符串\"{}\"的长度为 {}".format(str1, len(str1)))
print("列表 list1 的长度为 {}".format(len(list1)))

# 调用 max 函数
print("max(str1) is {}".format(max(str1)))
# 无法对 list1 调用 max 或 min 函数，因为其中的元素（字符串和数字无法比较大小）

# 调用 index 函数，第一次出现某个元素的位置
print("字符串\"{}\"的第一次出现字符 o 的位置为 {}".format(str1, str1.index("o")))
print("列表 list1 中元素 100 第一次出现的位置为 {}".format(list1.index(100)))

# 调用 count 函数，出现某个元素的总次数
print("字符串\"{}\"的出现字符 o 的次数为 {}".format(str1, str1.count("o")))
print("列表 list1 的元素 100 的出现次数为 {}".format(list1.count(100)))
```

目录

列表类型

列表类型作为有序序列类型的扩展 (派生)[2]

- 创建后可以随意被修改
- 列表中各元素类型可以不同，无长度限制

create-list.py:

```
# 创建列表

# 方法 1 使用 [] 或 list() 创建空列表，然后调用列表的方法为列表添加元素
list1 = []
list2 = list()

# 方法 2 创建时指定元素，元素间用逗号隔开
list3 = ['huawei', 'xman', 19.87, 2017]
print(list3)

# 备注：列表中可以嵌套列表，字典，元组等
list4 = [2018, 'A', True, list3]
print(list4)
```

[2] 在通用的方法和函数基础上，有自己独特的方法

访问列表

列表（有序序列）中的每个元素都分配一个数字索引，复习：正向/逆向
可以通过索引号访问列表的值，如 index-list.py：

```
1 l = ['huawei', 'xman', 4.56, 2018]
2 print('l[0]=', l[0])
3 print('l[1]=', l[1])
4 print('l[3]=', l[3])
5 print('l[-1]=', l[-1])   # 倒序访问列表，-1 表示最后一个
6
7 # 此行会报错，抛出 IndexError 异常，列表下标越界
8 print('l[4]=', l[4])
```

修改或新增列表的值

可对列表的数据项进行修改:

- 通过索引号修改列表中某项的值
- 通过 append() 方法对列表末尾增加值
- 通过 insert() 方法向列表中某位置插入值

revise-list.py:

```
l = ['huawei', 'xman', 4.56, 2018]
print('原列表', l)

l[1] = '莞工' # 修改列表第二个值
print('修改后的列表', l)

l.append('末尾追加元素') # 在列表的末尾追加值
l.append('yzncxb') # 在列表的末尾追加值
print('新增值后的列表', l)

l.insert(1, "广东") # 在第二个位置插入，插入后，后面元素向后移一位
print("插入新元素后的列表", l)
```

删除列表元素 1/3

删除元素的方法有:

- del() 通过" 列表名+索引" 或者" 列表名 + 切片" 删除 (函数)
- pop() 通过索引号删除, 一次只能删除一个元素 (方法)
- remove() 删除首个符合条件的元素, 一次只能删除一个元素 (方法)

del() 用法示例 del-list.py：

```
1 l = ['huawei', 'xman', 4.56, 2018]
2 print('原列表', l)
3
4 del l[0:2]   # 删除列表第一到第二个元素
5 print('删除前两个元素后的列表', l)
6
7 del l   # 删除整个列表，其他类型的变量亦可通过 del 删除
8 print(l)   # 列表无法访问，抛出 NameError，提示无 l 这个变量
```

删除列表元素 2/3

pop-list.py:

```
1 l = ['huawei', 'xman', 4.56]
2 print("原列表为",l)
3
4 var1=l.pop() # 不给定参数时，默认删除（弹出）最后一个元素
5 print("列表为", l, "弹出值为",var1)
6
7 l.pop(0)  # 删除列表第一个元素
8 print("列表为", l)
9
10 # 继续删除列表的值，只余 1 个元素，pop(0) 等价于 pop(-1)
11 l.pop()
12 print("列表为", l)
```

删除列表元素 3/3

remove-list.py:

```
1 l = ['huawei', 'xman', 4.56, 'xman']
2 print("原列表为",l)
3
4 l.remove('xman') # 只删除等于 xman 的第一个元素
5 print("列表为", l)
6
7 l.remove(2) # 删除不存在的值会抛出 ValueError 异常
```

列表切片及常用运算

slice-list.py:

```
1 l = ['huawei', 'xman', 4.56, 2018]
2 print("原列表为",l)
3
4 #l[1:2] = 88.8 # 本句用法错误，切片赋值必须是用列表做右值
5 l[1:3] = [77.7]
6 print("列表为", l)
7
8 l1=["1","2","3"]
9 l2=["a","b","c"]
10
11 l3=l1 + l2
12 print(l3)
13
14 l4=4*l2
15 print(l4)
```

列表类型相关函数与方法总览

ls: 待操作的列表变量名

函数及方法	描述
ls[i] = x	修改列表中索引 i 位置元素为 x
ls[i:j:k] = lt	lt 替换 ls 切片后所对应元素子列表
ls.append(x)	列表后面追加 x
ls.insert(i,x)	索引 i 的位置插入元素 x
del ls[i]	删除第 i 个元素
del ls[i:j:k]	删除某个切片对应的元素
ls.pop(i)	取出索引为 i 对应的元素并将它从列表中删除
ls.remove(x)	列表中出现的第一个 x 元素删除
ls.clear()	删除所有元素
ls.copy()	生成一个新列表，复制 ls 中所有元素
ls.reverse()	列表中元素反转
ls.sort([func])	列表中元素排序
ls.extend(seq)	在列表 ls 末尾一次性追加序列 seq 中的多个值

列表类型相关函数与方法示例

misc-list.py:

```
l = ['huawei', 'xman', 4.56, 2018]
print("原列表为",l)

l1 = [7, 8, 9]
l.extend(l1)  # 在列表末尾一次性追加另一序列中的多个值（用新列表扩展原来的列表）
print(l)

l.reverse()  # 反向列表中元素
print(l)

l2 = l.copy()  # 复制列表
print(l2)

l.clear()  # 清空列表
print(l)

l3 = [3, 4, 2, 9, 4]
l3.sort()  # 对原列表进行排序，前提是可排序，字符串和数字无法比较
print(l3)
```

目录

元组类型

元组 (tuple) 是有序序列类型的一种扩展

- 一旦创建就不能被修改
- 元组中各元素类型可以不同，无长度限制

创建元组

create-tuple.py:

```
# 创建元组

# 方法 1 使用 () 或 tuple() 创建空元组
tuple1 = ()
tuple2 = tuple()
print(tuple1)

# 方法 2 创建时指定元素，元素间用逗号隔开
tuple3 = ('huawei', 'xman', 19.87, 2017)
print(tuple3)

# 可不加括号
tuple4 = 'huawei', 'xman', 19.87, 2017
print(tuple4)

# 注意：元组中可以包含 list 列表，元组中 list 列表的值时可变的；
tuple5 = ('huawei', 'xman', 19.87, [1,2,3,4])
print(tuple5)

tuple6 = ('xman')  # 无逗号
print(type(tuple6)) # tuple5 是 str 类型

tuple6 = ('xman',)  # 当定义元组只有一个值时，要在后面加上逗号
print(type(tuple6))
```

元组操作 1/5

1. 访问元组: 与列表访问方法一致
2. 修改元组: 元组不能修改，但可通过元组相加创建新元组

tuple-add.py:

```
# tuple 元组元素不可以修改
# 但可同过元组相加创建一个新元组

t1 = ('huawei', 'xman')
t2 = (1, 2, 3, '莞工')

t3 = t1 + t2
print(t3)

t4 = t1 * 4
print(t4)
```

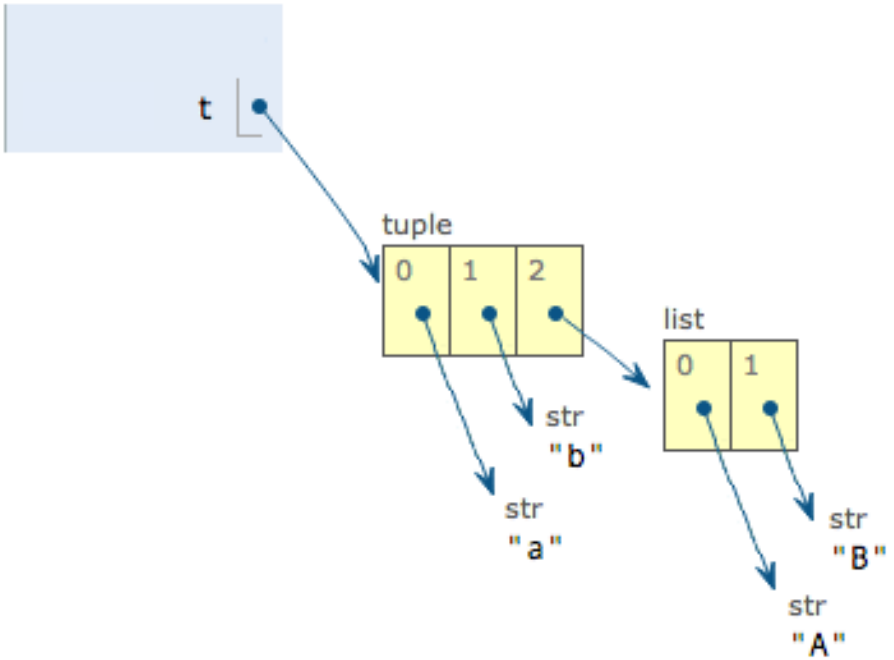
元组操作 2/5

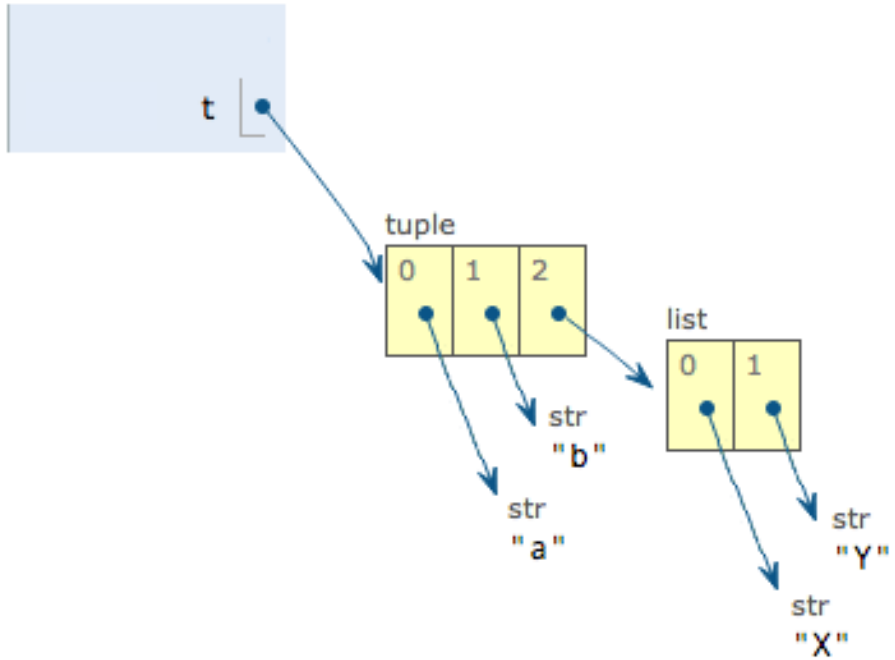
tuple-list.py:

```
# 元组中嵌套列表
t = ('a', 'b', ['A', 'B'])

t[2][0] = 'X'
t[2][1] = 'Y'

#t[2]=['C','D'] 思考，为什么本行会引发错误

print(t)
```

表面上看，tuple 的元素确实变了，但变的不是 tuple 的元素，而是 list 的元素。tuple 一开始指向的 list 并没有改成别的 list，所以，tuple 所谓的“不变”是说，tuple 的每个元素，指向永远不变。

元组操作 3/5


t
tuple
0
1
2
str
"b"
str
"a"
list
0
1
str
"B"
str
"A"


t
tuple
0
1
2
str
"b"
str
"a"
list
0
1
str
"Y"
str
"X"

元组操作 4/5

3. 删除操作. tuple 元组中的元素值是不允许删除，但可使用 del 语句来删除整个元组.

del-tuple.py:

```
t = ('huawei', 'xman', 4.56, 2018)
# del(t[1]) 错误，不能对 t 中的元素增删改

del t # 可删除整个元组
print(t)  # 元组已被删除，变量 t 无所指，抛出 NameError 异常
```

4. 切片：同列表

元组操作 5/5

- tuple(list1) 将列表转换为元组
- list(tuple1) 将元组转换为列表

convert.py:

```
1 t = ('huawei', 'xman', 4.56, 2018)
2 l = list(t)
3 l[1] = "莞工"
4 print("tuple: ", t)
5 print("list: ", l)
6
7 l1=tuple(l)
8 print(l1)
```

如无修改需求，尽量使用元组，效率更高

目录

应用场景

作用：表示一组有序数据，进而操作它们
遍历循环 (for...in...)
example.py:

```
# 定义列表或元组
#numbers = ['零', '一', '二', '三', '四', '五', '六', '七', '八', '九']
numbers = ('零', '一', '二', '三', '四', '五', '六', '七', '八', '九')

str_in = input("请输入一个数字:") # 接受用户输入，存入字符串 str_in

#num_in = int (str_in) # 将字符串 str_in 转换为整数 num_in

str_out=""
for s in str_in:  # 遍历输入字符串
    n = int (s) # 将字符转换为数字
    if n in range(len(numbers)): # 如果 n 在 numbers 长度范围内
        str_out = str_out + numbers[n] # 下标索引，+ 拼接

print(str_out)
```