【Python函数总结】

**目录**

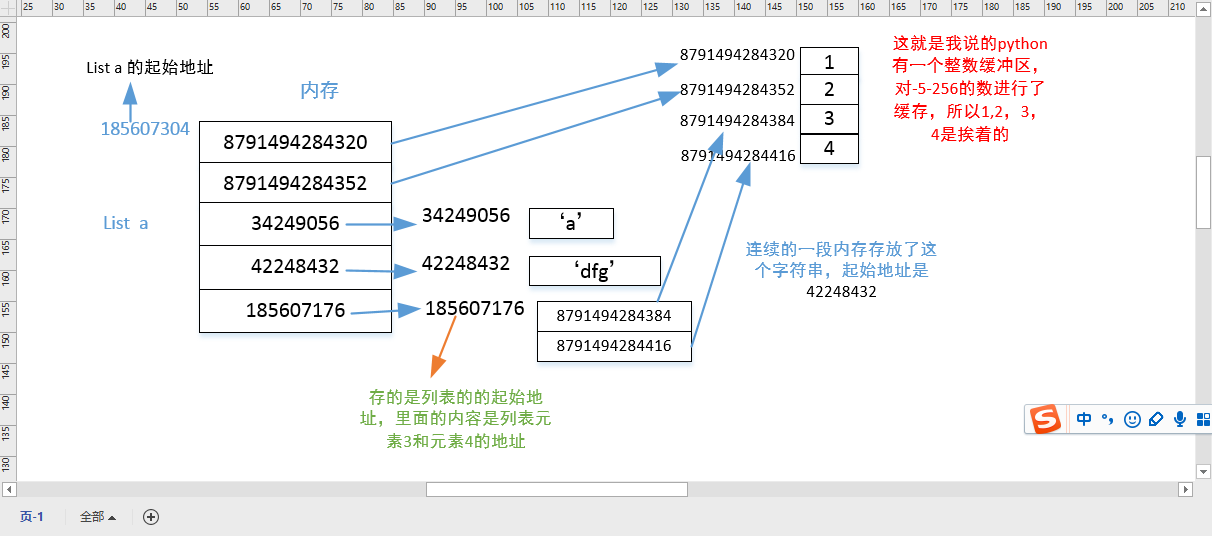
[Random包 2](#_Toc147769268)

[os.path包 2](#_Toc147769269)

[numpy包 2](#_Toc147769270)

## 【相关知识】

list 就是个指针数组，指针数组里面对应存放了每个元素所在的地址



Python并不支持++或者是--运算

Unicode 是国际标准字符集

UTF-8: 是一种变长字符编码

Python中的内存管理是基于引用计数的。简单来说，每个对象都有一个引用计数器，记录当前有多少个引用指向该对象。当引用计数为0时，表示该对象没有任何引用，可以被垃圾回收。

在Python中，当我们创建一个对象并将其赋值给一个变量时，变量实际上保存的是对象的引用，而不是对象本身。如果我们将同一个对象赋值给多个变量，它们共享同一个引用。当我们创建新的变量并将其赋值为原始变量时，会创建一个新的引用，引用计数会增加。当某个变量不再使用，或者重新赋值为其他对象时，原来对象的引用计数会减少。

当一个对象的引用计数变为0时，Python的垃圾回收机制（Garbage Collector）会自动回收该对象所占用的内存空间，并将其释放供其他对象使用。垃圾回收机制会定期地扫描内存中的对象，识别出无法访问到的对象，并将其回收。

除了引用计数，Python还使用了其他的内存管理技术，如分代回收和循环垃圾收集。分代回收根据对象的生命周期将其分为不同的代，通常是0代、1代和2代，根据不同的代执行不同频率的垃圾回收操作。循环垃圾收集用于检测和回收那些存在循环引用的对象，即使它们的引用计数不为0，也可以通过循环垃圾收集器进行回收。

总的来说，Python基于值的内存管理模式通过引用计数和垃圾回收机制来管理内存，并在需要时自动释放不再使用的对象所占用的内存空间。这种模式可以减少内存泄漏的风险，提高代码的效率和可靠性。

在Python中，运算符/和//都用于进行数值除法运算，但有一些区别。

/运算符执行的是浮点数除法。无论操作数是整数还是浮点数，返回的结果都是浮点数。例如：5 / 2的结果为2.5。

//运算符执行的是整数除法（向下取整）。无论操作数是整数还是浮点数，返回的结果都是一个整数，向下取整到最接近的整数值。例如：5 // 2的结果为2，-5 // 2的结果为-3。

Python中的运算符 % 可以对浮点数进行求余操作，其结果也会是一个浮点数

数字 5 是 Python 中的一个合法表达式

## 内置函数包

print() 打印函数

type() 判断类型

encode() 编码函数

decode() 解码函数

append() 列表新增成员函数

input() 接收函数，返回字符串

ord() 查看Unicode编码函数

chr() 根据编码数返回字符函数

map(function, iterable, ...) 根据提供的函数对指定序列做映射

function -- 函数

iterable -- 一个或多个序列

zip(a, b) zip 返回的是一个对象，需要手动将其list列表化，或者字典化，是一一对应的关系 注意：zip对象只能遍历一次

join()： 连接字符串数组

list() tuple() dict() set() 强制转换元素类型

frozenset() 冻结集合

isinstance(3, int) 判断类型

max() 最大函数

min() 最小函数

sum() 求和函数

len() 判断长度函数

eval() 字符串转换函数 🡪’xx’ 转化为list\dict\tuple\set

sorted() 排序函数

reversed() 逆序函数，返回的是一个对象，需要对其list字典化

repr() 转义字符函数，将对象转化为供解释器读取的形式

filter() 过滤函数，返回的是可迭代对象

split() 拆分字符串函数

## 匿名函数

lamdba

## Random包

random() 随机函数

randint() 随机整数函数

randrange() 范围随机函数

shuffle() 打乱函数

## os.path包

isfile() 判断目录是否存在

## numpy包

array() 将元组/列表转化为数组

## math包

gcd() 最大公约数

abs() 取模

## fractions分数包

Fraction() 创建分数函数

## Operator包

add() 相加函数

## functools包

reduce() 累计函数