

Preview

2018年8月7日 19:29

1. 常用操作符

a. 算术操作符（返回数或者文本）

操作符	名称	用途	实例	备注
+	加法	返回左右相加	3+2返回5	
-	减法	返回左减去右	3-2返回1	
*	乘法	返回左右相乘	3*2返回6	
**	指数	返回左取右次方数	3**2返回9	
/	除法	返回左除以右	3.0/2返回1.5	Python 3后, / 只用于浮点数除法, 返回浮点数
//	地板除（取商）	返回右对左取商	3.0//2返回1.0	Python 3后, // 只用于整数除法, 返回向下取整的整数
%	取模（取余）	返回右对左取模	5/3返回2;	取模即返回左数被右数整除后的余数

b. 赋值操作符

- 简单赋值符：=，将右值赋值给左
- 符合赋值符：+=，-=，*= 等等，将左右先进行算术运算符计算后的返回的值，赋值给左

c. 比较操作符（返回布尔值）

操作符	用途	实例
==	如果左右的值相等则返回True，否则返回False	3==2返回False
!=	如果左右的值不等则返回True，否则返回False	3!=2返回True
<>	同上	3<>2返回True

>	如果左大于右则返回True, 否则返回False	3>2返回True
<	如果左小于右则返回True, 否则返回False	3<2返回False
>=	如果左大于或等于右则返回True, 否则返回False	3>=2返回True
<=	如果左大于或等于右则返回True, 否则返回False	3<=2返回False

d. 逻辑操作符 (返回布尔值)

操作符	名称	用途	实例
and	逻辑与	只有左右均为True时返回True, 否则返回False	True and True = True True and False = False False and False = False
or	逻辑或	只有左右至少一个为True时返回True, 否则返回False	True or True = True True or False = True False or False = False
not	逻辑非	返回逻辑状态相反的布尔值	not True = False not False = True

e. 成员操作符

操作符	用途
in	如果左是在右中则返回True, 否则返回False
not in	如果左不在右中则返回True, 否则返回False

f. 操作符优先级

优先级			操作符		
最高	**				
	*	/	%	//	

	+	-			
	<=	<	>	>	=
	==	!=	<>		
	=	.=			
	in	not in			
最低	not	or	and		

2. 数据结构

a. 变量与赋值

Python 的变量是不可变对象，如果变量的值发生改变，将新创建一个新的对象申请一个新的内存地址，用于储存新的值，并将该变量指向新的内存地址（获得新值），而原有值的内存地址作废，等待回收。不可变对象的设计对程序的执行效率带来一定的影响

b. 数据类型

i. Python 三大基本数据类型：整数，浮点数，布尔值
(Python的浮点数是双精度浮点数，即double)

ii. type(x) 可以返回数据 x 的数据类型

iii. 整数与整数的运算结果仍然是整数，但只要其中一个为浮点数，运算结果即为浮点数

iv. 整数运算的结果是精确的，浮点数运算结果不一定精确。

float: 单精度浮点数，使用32位（4字节）来储存一个数字

double: 双精度浮点数，使用64位（8字节）来储存一个数字

3. 流程控制

a. If 语句

i. 布尔表达式：默认返回False 的值：False, None, 0, "", (), [], {} （其余单值默认返回True）

ii. 条件分支：if, elif, else

b. While 循环：while 后面的布尔表达式为True时，执行，直至布尔表达式为假

- i. break 语句：跳出最内层的循环（函数里用来跳过）
- ii. continue 语句：跳到最内层循环的首行
- c. for 循环
 - i. range() 函数：生成列表，左包含右不包含
 - ii. len()函数：返回列表的长度
 - ! iii. 使用 for l in range(len())可以实现遍历列表中所有元素的同时，可以实现修改元素（通过L[i]来索引访问），例子

L = [1, 2, 3]
for l in range (len (L))
L [i] += 1
print L # 输出的L是[2,3,4]

- ! d. else 语句：当for 遍历整个列表，或者 while 的循环条件为False 时，如果没有break 终止循环，则可以通过 else 来执行终止循环，例子：for 循环结束时通过 else 过渡

简单搜索质数
for n in range (2, 10):
for x in range (2, n):
if n % x == 0:
print n, "equals", x, "*", n/x
break
else:
print n, "是一个质数"

4. 数据结构

- a. 三种数据结构之一——标量（Scaler）：如整数，浮点数
- b. 三种数据结构之二——序列（Sequence）：

i. 列表 (List) : 一个任意类型的对象的位置相关的有序集合, 大小可变

1) 列表是有序的, 通过索引 (下标) 来访问列表中的元素

2) 索引: 从左索引, 从0开始。从右索引, 以负数表示, 从-1开始

3) 切片: $L[0:2]$ → 表示从左起的索引0, 到左起的索引1 (右不包含), 切出新列表;

切片符: 的左右如果不输入, 则表示从左 (或者右) 起第一个索引开始 (结束)

4) 列表方法:

<code>list.append(x)</code>	添加一个元素到列表的末尾
<code>list.extend(L)</code>	将列表 L 拼接到列表 list 的末尾
<code>list.insert(i, x)</code>	在索引 i 元素的前面插入一个元素x
<code>list.remove(x)</code>	删除列表第一个值为x的元素
<code>list.pop([i])</code>	删除列表中指定索引i位置的元素, 并返回, 如不输入索引参数, 默认删除返回最后一个元素
<code>list.index(x)</code>	返回列表第一个值为x的元素的索引
<code>list.count(x)</code>	返回列表中值为x的元素的个数
<code>list.sort()</code>	排序列表中的元素
<code>list.reverse()</code>	反转列表中的元素

5) 列表用作栈 (栈: 后进先出) : 在list中使用 `append()` 进行压入, 使用 `pop()` 进行弹出

★6) 列表用作队列 (队列: 先进先出) : 使用 `collections.deque` (`popleft`和`popright`)

ii. 字符串 (String) :

iii. 元祖 (Tuple) :

iv. Unicode字符串

v. 字节数组

vi. 缓冲区

vii. xrange对象

c. 三种数据结构之三——映射 (Mapping) : 字典 (Dictionary)

d. 独立于三种数据结构之外的数据结构：集合 (Set)