



北京大学
PEKING UNIVERSITY

信息科学技术学院

实用Python程序设计

郭 炜

微信公众号



微博: <http://weibo.com/guoweiofpku>

学会程序和算法，走遍天下都不怕!



算术运算、逻辑运算和分支语句



北京大学
PEKING UNIVERSITY

信息科学技术学院

算术运算



宁夏中卫沙坡头

算术运算符

➤ Python 支持以下算术运算

+ 加法

- 减法(双操作数)， 取相反数（单操作数）

* 乘法

/ 除法 ， 结果是小数。即便能整除也是小数

% 取模(求余数)

// 求商，结果是整数

** 求幂

算术表达式

```
a = (3+2)*(6-3)/2
```

```
print(a) #>>7.5
```

```
print(10/8) #>>1.25
```

```
print(10%8) #>>2
```

```
print(15/4) #>>3.75
```

```
print(15//4) #>>3
```

```
print(3.4/2.2) #>>1.5454545454545452
```

```
print(3.4//2.2) #>>1.0
```

```
print(2**3) #>>8
```

```
print(-9//4) #>>-3 往小里取整
```

算术表达式

➤ / 计算的结果都是小数，哪怕能整除

```
z = 10/2
```

```
print(z)      #>>5.0
```

➤ $-x$ 相当于 $0-x$

```
a = 10
```

```
print(-a)     #>>-10
```

```
print(-a*3)   #>>-30    等价于 (-a) * 3
```

```
print(3+-5)   #>>-2
```

算术表达式

➤ 有小数的算术表达式，结果就是小数

```
z = 10.0 - 10
```

```
print(z) #>>0.0
```

```
z = 7.5 - 2.5
```

```
print(z) #>>5.0
```

```
print(2+0*4.5) #>>2.0
```

```
print(2+10/5) #>>4.0
```

算术运算符优先级

第一级: **

第二级: -(求相反数)

第三级: * / // %

第四级: + - (减法)

可以用 () 指定计算顺序: $(3+4) * (1+2)$

记不得优先级就用 ()

算术运算的同时赋值

+=

a += b

等价于 **a = a + b**

-=

a -= b

等价于 **a = a - b**

***=**

.....

/=

%=

//=

****=**

算术运算的同时赋值

```
a = 6
```

```
a/=3
```

```
print(a)      #>>2.0
```

```
a**=3
```

```
print(a)      #>>8.0
```

常见问题或注意事项

数学上的运算符号或代数式写法，不是拿到程序里就能直接用的！

比如 $|x|$ 在程序里并不能求 x 的绝对值，而是没定义，会出错！

$2(x+3)(4+x)$ 这样的表达式也不能在程序里写！

应该写：

$2 * (x+3) * (4+x)$

乘法就一定要用 $'*'$ ！

常见问题或注意事项

/ 的结果一定是小数。一个算术表达式，只要有一个操作数是小数，其结果就一定是小数。

如果x是小数`print(x)`就会打出小数形式，哪怕 $x = 4/2$ 。如果题目要求是整数，就要注意了，不要把整数当小数打出来。

`print(4/2)`打出 2.0

常见问题或注意事项

/ 是 \div ，不是分数线

因此

$a/b*c$ 是 $a \div b \times c$ ，不是 $a/(b*c)$

print(4+6/2)的结果是

- ☐ A 7
- ☐ B 7.0
- ☐ C 5.0
- ☐ D 5

此题未设答案

提交

print(4+6/2)的结果是

- ☐ A 7
- ☒ B 7.0
- ☐ C 5.0
- ☐ D 5

提交



北京大学
PEKING UNIVERSITY

信息科学技术学院 郭炜

关系运算符 逻辑运算符 逻辑表达式



韩国济州岛火山口

关系运算符和bool 类型

➤六种关系运算符用于比较大小

相等 ==

不等 !=

大于 >

小于 <

大于等于 >=

小于等于 <=

➤比较的结果是bool 类型，成立则为True, 反之为False

➤bool 类型数据只有两种取值，True或False

关系运算符和bool 类型

```
print(3 < 5)           #>>True
print(4 != 7)          #>>True
a = 4
print(2 < a < 6 < 8)   #>>True
print(2 < a == 4 < 6)  #>>True
print(2 < a > 5)        #>>False
b = a < 6
print(b)               #>>True
print( b == 1)         #>>True
print( b == 2)         #>>False
b = a > 6
print( b == 0)         #>>True
a = True
print(a == 1)          #>>True
```

关系运算符和bool 类型

➤关系运算符也能比较字符串（按字典序，大小写相关）

```
a = "k"
print(a == "k")           #>>True
a = "abc"
print(a == "abc")         #>>True
print(a == "Abc")         #>>False
print("abc" < "acd")      #>>True
print("abc" < "abcd")     #>>True
print("abc" > "Abc")      #>>True
```

逻辑运算符和逻辑表达式

➤逻辑运算符用于表达式的逻辑操作，有 `and` `or` `not` 三种，操作的结果是True或False

●与： `exp1 and exp2`

当且仅当exp1和exp2的值都为True（或相当于True）时，结果为True（或相当于True）

```
n = 4
```

```
n > 4 and n < 5 # false
```

```
n >= 2 and n < 5 and n%2 == 0 # true
```

```
print(5 and False) #>>False
```

```
print(4 and True) #>>True
```

什么相当于True或False

- 0, "" (空字符串), [] (空表) 都相当于False (但除0以外都不等于False)
- 非0的数, 非空的字符串和非空列表, 都相当于True (但除1以外, 都不等于True)
- True 可以看作 1, False 可以看作 0

True == 1	#True
False == 0	#True
"" == False	#False
2 == True	#False
[] == False	#False
[2, 3] == True	#False

逻辑运算符和逻辑表达式

●或 : exp1 **or** exp2

当且仅当exp1和exp2的值都为False（或相当于False）时，结果为False（或相当于**False**）

```
n = 4
```

```
n > 4 or n < 5      #True
```

```
n <= 2 or n > 5      #False
```

逻辑运算符和逻辑表达式

●非 : `not exp`

exp值为True(或相当于True)时, 结果为False(或相当于False)
exp值为False(或相当于False)时, 结果为True(或相当于True)

```
not 4 < 5           #False
not 5                #False
not 0                #True
not "abc"            #False
not ""               #True
not 4 < 5 and 4 > 6   #False
not []               #True
not [1]              #False
```

逻辑运算符的优先级

➤ not > and > or

```
print(3 < 4 or 4 > 5 and 1 > 2 )           #>>True  
print((3 < 4 or 4 > 5) and 1 > 2 )         #>>False
```

not 4 < 5 and 4 > 6 即 (not 4 < 5) and (4 > 6)

逻辑运算符和逻辑表达式

逻辑表达式是**短路计算**的，即对逻辑表达式的计算，在整个表达式的值已经能够断定的时候即会停止

- `exp1 and exp2` : 如果已经算出表达式`exp1`为假，那么整个表达式的值肯定为假，于是表达式`exp2`就不需要再计算
- `exp1 or exp2` : 如果已经算出`exp1`为真，那么整个表达式必定为真，于是`exp2`也不必计算

各种运算符的优先级

➤从高到低：

- 算术运算符 + - * / // % **
- 关系运算符 < > == != <= >=
- 逻辑运算符 and or not

记不得就勤用 ()

```
print( 3 + 2 < 5 )
```

```
#>>False
```

```
print( 3 + (2 < 5) )
```

```
#>>4 ,因 2 < 5相当于1
```

以下程序输出几个True

```
print( [] == False)
print( not [])
print( 0 == False)
print( 1 == True)
print( not "")
```

- ☐ A 5
- ☐ B 4
- ☐ C 3
- ☐ D 2

提交

以下程序输出几个True

```
print( [] == False)
print( not [])
print( 0 == False)
print( 1 == True)
print( not "")
```

- ☐ A 5
- ☒ B 4
- ☐ C 3
- ☐ D 2

提交



北京大学
PEKING UNIVERSITY

信息科学技术学院 郭炜

条件分支语句 (if 语句)



美国加州太浩湖

条件分支语句

有时，并非所有的程序语句都要被顺序执行到，会希望满足某种条件就执行这部分语句，满足另一条件就执行另一部分语句。这就需要“条件分支语句”

条件分支语句

```
if 表达式1:
    语句组1
elif 表达式2:
    语句组2
... #可以有多个 elif
elif 表达式n:
    语句组n
else:
    语句组n+1
```

依次计算表达式1、表达式2...只要碰到一个表达式i为真，则执行语句组i（前面为假的表达式对应的语句组不会被执行），**且后面的表达式不再计算**，后面的语句组也都不会被执行。

若所有表达式都为假，则执行语句组n+1

注意，缩进的前一行末尾有 ' : '

条件分支语句

可以没有 `elif`，也可以没有 `else`，也可以都没有

```
if 表达式1 :  
    语句组1  
else :  
    语句组2
```

```
if 表达式1:  
    语句组1  
elif 表达式2:  
    语句组2
```

```
if 表达式1 :  
    语句组1
```


条件分支语句的缩进

➤Python程序的语句前面不能加空格或制表符，除非：

1. 它在if语句中的某个“语句组”里面
2. 在 `for` , `while` 等语句的语句组里面
3. 在函数体里面
4.

条件分支语句的缩进

➤ if 语句中的语句组，每条语句左边必须缩进，且缩进情况必须一样(对齐)

```
if int(input()) == 5:  
    print("a",end="")  
    print("b")
```

输入： 5 输出： ab

输入： 4 无输出

条件分支语句的缩进

➤ if 语句中的语句组，每条语句左边必须缩进，且缩进情况必须一样(对齐)

```
if int(input()) == 5:  
    print("a",end="")  
print("b")
```

输入： 4 输出： b

条件分支语句的缩进

➤ if 语句中的语句组，每条语句左边必须缩进，且缩进情况必须一样（对齐）

```
if int(input()) == 5:  
    print("a",end="")  
    print("b")
```

出错！没有对齐的缩进！

什么相当于True或False

<code>if "ok":</code>	
<code>print("ok")</code>	<code>#>>ok</code>
<code>if "":</code>	
<code>print("null string")</code>	<code>#无输出</code>
<code>a = [4,2]</code>	
<code>if a:</code>	
<code>print(a)</code>	<code>#>>[4,2]</code>
<code>if 20:</code>	
<code>print(20)</code>	<code>#>>20</code>
<code>if 0:</code>	
<code>print(0)</code>	<code>#无输出</code>

if 语句示例

例题：写一个判断整数奇偶性的程序，要求输入一个整数，如果是奇数，就输出 “It's odd.”，如果是偶数，就输出 “It's even.”。

```
if int(input()) % 2 == 1:  
    print("It's odd")  
else:  
    print("It's even")
```

if 语句嵌套

➤在一条if语句的某个分支(语句组)里,还可以再写if语句。

```
a = int(input())
if a > 0:
    if a % 2:
        print("good")
    else:
        print("bad")
```

输入: 4	输出: bad
输入: 3	输出: good
输入: -1	无输出

if 语句嵌套

➤在一条if语句的某个分支(语句组)里,还可以再写if语句。

```
a = int(input())
if a > 0:
    if a % 2:
        print("good")
else:
    print("bad")
```

输入: 4	无输出
输入: 3	输出: good
输入: -1	bad

if 语句实例

➤ 例：输入密码

```
password = "python"
userInput = input()    #等待用户输入密码
if userInput == password:  # == 判断两边的表达式值是否相等
    print("对了!")
else:
    print("错了")
```



北京大学
PEKING UNIVERSITY

信息科学技术学院 郭炜

分支语句例题



美国黄石公园大棱镜温泉

if语句实例：温度转换程序

```
tmpStr = input("请输入带有符号的温度值： ") #tmpStr是变量名，随便取啥都行
if tmpStr[-1] in ['F','f']: #如果输入华氏温度
    C = ((float(tmpStr[0:-1])) - 32 ) / 1.8
    print("转换后的温度是" + str(C) + "C")
elif tmpStr[-1] in "Cc": #如果输入摄氏温度
    F = 1.8 * eval(tmpStr[0:-1]) + 32
    print("转换后的温度是" + str(F) + "F") #str将F转字符串
else:
    print("输入格式错误")
# 用float替代 eval 也可以
```

请输入带有符号的温度值： 45F
转换后的温度是7.222222222222222C

请输入带有符号的温度值： 8.2C
转换后的温度是46.76F

字符串切片

若s是一个字符串,

则:

`s[x:y]` 是s的从下标x到下标y的左边那个字符构成的子串 (切片)

```
print("12345"[1:3])  #>> 23
```

```
a = "abcdef"
```

```
print(a[2:-1])       #>> cde
```

```
print(a[0:6])        #>> abcdef
```

常见错误

➤ 不要把 `if ... else` 或 `if...elif...else` 写成多个 `if`

```
if a > 5:
```

```
....
```

```
print("xxx")
```

```
a = 3
```

```
.....
```

```
else:
```

```
print(b)
```

~~=~~

```
if a > 5:
```

```
....
```

```
print("xxx")
```

```
a = 3
```

```
.....
```

```
if a <= 5
```

```
print(b)
```

.... 中可能把 `a` 的值改成了 `<= 5`。写多个 `if`，哪些代码会被执行，哪些不会被执行，很容易搞错。

if 语句嵌套

例题:请写一个程序，该程序输入一个年份，根据该年份是否是建国整十周年、建党整十周年以及是否是闰年给出不同的输出。

-2✓

Illegal year.

1959✓

Legal year.

Luky year.

1931✓

Legal year.

Good year.

2008✓

Legal year.

Leap year.

2011✓

Legal year.

Common

year.

```
year = int(input())
if year <= 0:
    print("Illegal year")
else:
    print("Legal year.")
    if year > 1949 and (year - 1949) % 10 == 0 : #建国整十
        print("Luky year.")
    elif year > 1921 and not ((year - 1921) % 10) : #建党整十
        #只是为了演示not的用法，没必要这么写
        print("Good year.")
    elif year % 4 == 0 and year % 100 or year % 400 == 0:
        # year % 100 若不为0，则 year % 100 就相当于True
        print("Leap year") #闰年
    else:
        print("Common year.")
```




北京大学
PEKING UNIVERSITY

信息科学技术学院

输出格式控制



甘肃张掖七彩丹霞（航拍）

输出格式控制

➤ 字符串中的格式控制符：

%s 表示此处要输出一个字符串

%d 表示此处要输出一个整数

%f 表示此处要输出一个小数

%.nf 表示此处要输出一个小数，保留小数点后面n位，**四舍六入，五则可能入也可能舍。注意，有 '.'**

.....

格式控制符只能出现在字符串中！

输出格式控制

```
h = 1.746
```

```
print("My name is %s,I am %.2fm tall." % ("tom",h))
```

```
print("My age is %d." % 18)
```

```
print("%d%s" % (18,"hello"))
```

```
print("%.2f,%.2f" % (5.225, 5.325)) #>> 5.22,5.33
```

输出:

```
My name is tom,I am 1.75m tall.
```

```
My age is 18.
```

```
18hello
```

```
5.22,5.33
```

输出格式控制

```
name = "tom"
```

```
h = 1.746
```

```
"My name is %s,I am %.2fm tall." % ("tom",h)
```

是个字符串。比下面这个等价字符串简洁：

```
"My name is %s" % name + "I am %.2fm tall." % h
```

常见问题或注意事项

◆ 题目： 在一行输入两个小数 x, y ， 请输出 $(x+y) * x$ 的值， 保留小数点后面5位

◆ 错误解法：

```
s = input().split()
```

```
x,y = float(s[0]),float(s[1])
```

```
m = '%.5f' % (x+y)
```

```
z = float(m) * x
```

```
print("%.5f" % z)
```

常见问题或注意事项

◆ 错误解法:

```
s = input().split()
x,y = float(s[0]),float(s[1])
m = '%.5f' % (x+y)
z = float(m) * x
print("%.5f" % z)
```

x+y本来就是小数没有必要先转成字符串，又转成小数。结果要保留小数点后面5位，并非中间的计算过程也要保留小数点后面5位