**[Maximum and Minimum values for ints](https://stackoverflow.com/questions/7604966/maximum-and-minimum-values-for-ints)**

For Python 3, it is

import sys

max = sys.maxsize

min = -sys.maxsize -1

# python中的float除法和整除法

从python2.2开始，便有两种除法运算符：**"/"、"//"**。两者最大区别在：

1. python2.2前的版本和python2.2以后3.0以前的版本的默认情况下，**"/"所做的除法是以一种两个数或者多个数出现一个浮点数结果就以浮点数的形式表示，即float除法**
2. "//"所做的除法则不相同，**"//"不管两者出现任何数，都以整除结果为准，不对小数部分进行处理，直接抛弃，也就是整除法**

**以下是笔者在编译器测试的数据，测试的版本是python2.7**

关于"/"：

>>> 3/2

1

>>> 3/2.0

1.5

>>> 3.0/2

1.5

>>> 10/3.0

3.3333333333333335

由以上的例子我们可得出，只要除数中**有一个数**是浮点数，那么得到的结果也是浮点数

以下是关于"//":

>>> 3//2

1

>>> 3//2.0

1.0

>>> 3.0//2

1.0

>>> 3.0//2.0

1.0

由以上的例子，我们可以看出，两个整型相除，得到的还是整型，但是。如果一个浮点数除以一个非浮点数得到的还会是一个浮点数，但是，计算的结果却是忽略小数部分，运算的结果类似两个整型相除，但是得到一个浮点数。另外**"//"对于两个浮点数也是不例外的。**

还需要注意的是，在**pyhton3**中，**"/"表示的就是float除**，不需要再引入模块，**就算分子分母都是int，返回的也将是浮点数**

# [Print a list in reverse order with range()?](https://stackoverflow.com/questions/7286365/print-a-list-in-reverse-order-with-range)

Use the 'range' built-in function. The signature is range(start, stop, step). This produces a sequence that yields numbers, starting with start, and ending if stop has been reached, excluding stop.

>>> **range(9,-1,-1)**

[9, 8, 7, 6, 5, 4, 3, 2, 1, 0]

>>> range(-2, 6, 2)

[-2, 0, 2, 4]

In Python 3, this produces a non-list range object, which functions effectively like a read-only list (but uses way less memory, particularly for large ranges).

# Python sqrt() 方法

## 語法

以下是sqrt()方法的語法：

import math

math.sqrt( x )

注意：此函數是無法直接訪問的，所以我們需要導入math模塊，然後需要用math的靜態對象來調用這個函數。

## 例子

下麵的例子顯示了sqrt()方法的使用。

#!/usr/bin/python

import math # This will import **math** module

print "math.sqrt(100) : ", math.sqrt(100)

print "math.sqrt(7) : ", math.sqrt(7)

print "math.sqrt(math.pi) : ", math.sqrt(math.pi)

當我們運行上麵的程序，它會產生以下結果：

math.sqrt(100) : 10.0

math.sqrt(7) : 2.64575131106

math.sqrt(math.pi) : 1.77245385091

# Python round() 函数

## 描述

**round()** 方法返回浮点数x的四舍五入值。

## 实例

以下展示了使用 round() 方法的实例：

#!/usr/bin/python

print "round(80.23456, 2) : ", round(80.23456, 2)

print "round(100.000056, 3) : ", round(100.000056, 3)

print "round(-100.000056, 3) : ", round(-100.000056, 3)

以上实例运行后输出结果为：

round(80.23456, 2) : 80.23

round(100.000056, 3) : 100.0

round(-100.000056, 3) : -100.0

# python int() round() floor()函数的比较

猛的一看  int（） round()  math.floor() 这几个函数函数好像做的是同一件事情，很容易将他们弄混，下面是他们的一些不同之处：

* int（）函数直接截去小数部分
* floor（） 得到最接近原数但是小于原数的部分
* round() 得到最接近原数的整数（返回为浮点类型）

# [How can I reverse a list in Python?](https://stackoverflow.com/questions/3940128/how-can-i-reverse-a-list-in-python)

<https://stackoverflow.com/questions/3940128/how-can-i-reverse-a-list-in-python>

>>> L = [0,10,20,40]

>>> **L[::-1]**

[40, 20, 10, 0]

Extended slice syntax is explained well in the Python [What's new Entry for release 2.3.5](http://docs.python.org/release/2.3.5/whatsnew/section-slices.html)

By special request in a comment [this is the most current slice documentation](http://docs.python.org/library/functions.html#slice).

[share](https://stackoverflow.com/a/3940137)[improve this answer](https://stackoverflow.com/posts/3940137/edit)