文章目录

一、math模块

> math库是Python提供的内置数学类函数库，因为复数类型常用于科学计算，一般计算并不常用，因此math库 不支持 复数类型，仅支持 整数和浮点数运算。

```
import math
```

1、数学常数

常数	说明	实例
math.pi	圆周率 π	>>> `math.pi` 输出结果：`3.141592653589793`
math.e	自然常数e	>>> `math.e` 输出结果：`2.718281828459045`
math.inf	正无穷大，负无穷大为：`-math.inf`	>>> `math.inf` 输出结果：`inf`
math.nan	非浮点数标记，NaN	>>> `math.nan` 输出结果：`nan`

2、常用函数

math.ceil(浮点数)

> 向上取整操作；返回值：整数

```
>>> import math
>>> math.ceil(13.14)
14
>>> math.ceil(9.9)
10
>>> math.ceil(19) # 整数无效
19
```

math.floor(浮点数)

> 向下取整操作；返回值：整数

```
>>> import math
>>> math.floor(13.14)
13
>>> math.floor(9.9)
9
>>> math.floor(19) # 整数无效
19
```

round(浮点数)

> 四舍五入操作；返回值：整数

```
>>> import math
>>> round(13.14)
13
>>> round(9.9)
10
>>> round(11.936, 2) # 保留两位小数的方式
11.94
>>> round(9) # 整数无效
9
```

math.fabs(数值)

> 获取数值绝对值操作；返回值：浮点数

```
>>> import math
>>> math.fabs(-9)
9.0
>>> math.fabs(9)
9.0
>>> math.fabs(-9.9)
9.9
>>> math.fabs(9.9)
9.9
```

abs(数值)

> 获取数值绝对值操作；返回值：整数、浮点数(根据原数据的类型而定)

```
>>> import math
>>> abs(-9)
9
>>> abs(-9.9)
9.9
```

math.fmod(x, y)

> 返回 `x/y` 的余数；返回值：浮点数

```
>>> import math
>>> math.fmod(4, 2)
0.0
>>> math.fmod(5, 2)
1.0
>>> math.fmod(10, 3)
1.0
```

math.pow(底数,幂)

> 计算一个数值的N次方；返回值：浮点类型

```
>>> import math
>>> math.pow(2,4)
16.0
>>> math.pow(3,2)
9.0
>>> math.pow(5, 3)
125.0
```

math.sqrt(数值)

> 开平方；返回值：浮点数

```
>>> import math
>>> math.sqrt(9)
3.0
>>> math.sqrt(4)
2.0
>>> math.sqrt(16)
4.0
```

fsum(序列)

> 返回序列中所有元素的和；返回值：浮点数

```
>>> import math
>>> math.fsum((1, 2, 3, 4, 5))
15.0
>>> math.fsum(range(1,11))
55.0
>>> math.fsum(range(1,101))
5050.0
```

sum(序列)

> 将一个序列的数值进行相加求和；返回值：数值类型（根据序列中数值的类型变化）

```
>>> import math
>>> sum([1,2,3,4,5])
15
>>> sum(range(1,11)
... )
55
>>> sum([1.0,2.0,3.0,4.0,5.0])
15.0
```

math.modf(数值)

> 将一个浮点数拆成小数和整数部分组成的元组；返回值：元组

```
>>> import math
>>> math.modf(10.1)
(0.09999999999999964, 10.0)
>>> math.modf(9.9)
(0.9000000000000004, 9.0)
>>> math.modf(9)
(0.0, 9.0)
```

math.trunc(浮点数)

> 返回浮点数的整数部分；返回值：整数

```
>>> import math
>>> math.trunc(2.1)
2
>>> math.trunc(9.9)
9
>>> math.trunc(10.0)
10
```

math.copysign(值1，值2)

> 将第二个数的正负号复制给第一个数；返回值：浮点数（值1 符号是值2的正负号）

```
>>> import math
>>> math.copysign(-2, 1)
2.0
>>> math.copysign(2,-1)
-2.0
```

math.factorial(x)

> 返回 x 的阶乘，如果 x 不是整数或为负数时则将引发 ValueError；返回值：整数

```
>>> import math
>>> math.factorial(4)
24
>>> math.factorial(3)
6
>>> math.factorial(1)
1
```

math.gcd(x, y)

> 返回整数 x 和 y 的最大公约数；返回值：整数

```
>>> import math
>>> math.gcd(2,4)
2
>>> math.gcd(3,9)
3
>>> math.gcd(9,6)
3
```

二、decimal模块

> decimal 模块提供了一个Decimal数据类型用于浮点数计算。相比内置的二进制浮点数实现float这个类型有助于金融应用和其它需要精确十进制表达的场合，控制精度，控制舍入以适应法律或者规定要求，确保十进制数位精度，或者用户希望计算结果与手算相符的场合。Decimal重现了手工的数学运算，这就确保了二进制浮点数无法精确保有的数据精度。高精度使Decimal可以执行二进制浮点数无法进行的模运算和等值测试。

1、什么时候使用decimal

python中小数相加可能会计算出结果不对，那就是由于科学计算精度问题

```
>>> 2.02 + 3.01
5.029999999999999
>>>
```

如上：我们需要得要的值是`5.03`，如果需要处理这个问题的话就需要用到`decimal`模块了

2、使用decimal

设置精度：`decimal.getcontext().prec = num` （num为有效数字个数）

```
>>> import decimal

>>> decimal.getcontext().prec = 3
>>> print(decimal.Decimal(2.02) + decimal.Decimal(3.01))
5.03

>>> decimal.getcontext().prec = 2
>>> print(decimal.Decimal(2.02) + decimal.Decimal(3.01))
5.0
```

设置小数位数：`quantize()`

```
import decimal

print(decimal.Decimal(1.1234567890).quantize(decimal.Decimal("0.000")))  # 设置3位小数
print(decimal.Decimal(1.1234567890).quantize(decimal.Decimal("0.00")))  # 设置2位小数
print(decimal.Decimal(1.1234567890).quantize(decimal.Decimal("0.0")))  # 设置1位小数
```

输出结果：

```
1.123
1.12
1.1
```