算术运算符

In [5]:

```
print(2+4)#加法运算
print( 2-4)#减法运算
print(2*4)#乘法运算
print(2/2)#除法运算
print(3/2)#除法运算
print(3//2)#5整除运算
print(5%3)#取余运算
print(2**2)#表示的是2的2次方  2*2
print(2**3)#表示的是2的3次方  2*2*2
```

```
6
-2
8
1.0
1.5
1
2
4
8
```

In [6]:

```
print(9//4)
print(-9//-4)
print(-9//4)
print(9//-4)#一正一负向下取整
print(-9 % 4)#一正一负遵循公式 9-(-4)*(-3)9-12--> 3
print(9 % -4)#余数=被除数-除数*商 9-(-4)*(-3) 9-12--> -3
```

```
2
2
-3
-3
3
-3
```

链式赋值

In [7]:

```
#链式赋值
i = 3+4
print(i)
a=b=c=20 #链式赋值
print(a,id(a))#只有一个整数对象却有三个引用，在指向这个位置上
print(b,id(b))
print(c,id(c))
```

```
7
20 140732010604944
20 140732010604944
20 140732010604944
```

支持参数赋值

In [8]:

```
#参数赋值
print('-------支持参数赋值-------')
a = 20
a += 30  #相当于a=a+30
print(a)
a -= 10  #相当于a=a-10
print(a)
a *= 2  #相当于a=a*2
print(a)
print(type(a)) #这里还是int
a /= 3  #相当于a=a除3
print(a)
print(type(a)) #这里变成float
a //= 2  #相当于a整除2
print(a)
print(type(a))
a %= 3  #相当于a=a除3后的余数
print(a)
print(type(a))
```

```
-------支持参数赋值-------
50
40
80
<class 'int'>
26.666666666666668
<class 'float'>
13.0
<class 'float'>
1.0
<class 'float'>
```

In [9]:

```
print('相对于上面的，因为流程中没有了float类型所以全部是int不改变')
a = 20
a += 30  #相当于a=a+30
print(a)
a -= 10  #相当于a=a-10
print(a)
a *= 2  #相当于a=a*2
print(a)
print(type(a)) #这里还是int
a //= 2  #相当于a整除2
print(a)
print(type(a))
a %= 3  #相当于a=a除3后的余数
print(a)
print(type(a))
```

```
相对于上面的，因为流程中没有了float类型所以全部是int不改变
50
40
80
<class 'int'>
40
<class 'int'>
1
<class 'int'>
```

支持系列解包赋值，要求等号两边个数相同

In [11]:

```
a,b,c=20,30,40
print(a,b,c)
```

```
20 30 40
```

In [12]:

```
a,b,c = 20,30,40
print('a,b,c')#加了 ' 就变成了字符串不是赋值了
```

```
a,b,c
```

In [21]:

```
a,c = 20,30,40
print(a,b,c) #两边数量不一
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-21-1a9823f2f126> in <module>
----> 1 a,c = 20,30,40
      2 print(a,b,c) #两边数量不一

ValueError: too many values to unpack (expected 2)
```

解包优点，交换两个变量值

In [14]:

```
a,b = 20,30
print(a,b)
a,b = b,a
print(a,b)
```

```
20 30
30 20
```

比较运算符

-->对变量或表达式的结果进行大小、真假等比较

In [15]:

```
a,b = 10,20
print('a>b,吗?',a>b)
a,b = 10,20
print('a<b,吗?',a<b)
print('a<=b,吗?',a<=b)
print('a>=b,吗?',a>=b)
print('a!=b,吗?',a!=b)
"""一个 = 称为赋值运算符，两个如 == 称为比较运算符（同C）
  一个变量由三部分组成，标识，类型，值
  ==比较的是值还是标识呢？比较的是值，比较标识用的是 is
  疑问——：标识是什么？？
"""
```

```
a>b,吗? False
a<b,吗? True
a<=b,吗? True
a>=b,吗? False
a!=b,吗? True
```

Out[15]:

```
'一个 = 称为赋值运算符，两个如 == 称为比较运算符（同C）\n  一个变量由三部分组成，标识，类型，值\n  ==比较的是值还是标识呢？比较的是值，比较标识用的是 is\n  疑问——：标识是什么？？\n'
```

In [16]:

```
a = 10
b = 10
print(a==b) #True说明，a与b的value相等
print(a is b) #True说明，a与b的id标识，相等
print(a is not b) # False  a的id与b的id是不相等的
list1 = [11,22,33,44]
list2 = [11,22,33,44]
print(list1==list2)  #比较的是value 值
print(list1 is list2) #比较的是id  标识
print(id(list1)) # id不同
print(id(list2))
print(a is not b) # False  a的id与b的id是相等的
print(list1 is not list2) # True list1的id与list2的id是不相等的
```

```
True
True
False
True
False
1924451050944
1924450825216
False
True
```

布尔运算符

In [3]:

```
a,b = 1,2
print('---------------and并且---------------')
print(a==1 and b==2) #True  True  and True-->True
print(a==1 and b<2)  #False True  and False -->False
print(a!=1 and b==2) #False False and True-->False
print(a!=1 and b!=2) #False False and False -->False
print('---------------or或者---------------')
print(a==1 or b==2) #True  or True -->True
print(a==1 or b<2)  #True  or False -->True
print(a!=1 or b==2) #False or True -->True
print(a!=1 or b!=2) #False or False -->False
print('-------not 对bool类型操作数取反------')
f1=True
f2=False
print(not f1)
print(not f2)
print('---------------in 与not in------------')
s = 'Hello world'
print('w' in s)
print('k' in s)
print('w' not in s)
print('k' not in s)# in 表示是否在这个字符串内 not in 是否不在
```

```
---------------and并且---------------
True
False
False
False
---------------or或者---------------
True
True
True
False
-------not 对bool类型操作数取反------
False
True
---------------in 与not in------------
True
False
False
True
```

成员检测

In [8]:

```
list1 = ['a','b']
list2 = ['b','a']
list2.reverse()#位置取反
print("列表1和列表2的值相同：1",list2==list1)
print("列表1和列表2的储存位置相同：1",list2==list1)
```

```
列表1和列表2的值相同：1 True
列表1和列表2的储存位置相同：1 True
```

位运算符

————将数据转成二进制进行计算

In [18]:

```
位与& —>对应数位都是1，结果数位才是1，否则为0
位或| -->对应数位都是0,结果数位才是0，  否则为1
左移位运算符<< -->高位溢出舍弃,低位补0
右移位运算符>> -->低位溢出舍弃,高位补0
```

```
  File "<ipython-input-18-c635a453415f>", line 1
    位与& —>对应数位都是1，结果数位才是1，否则为0
        ^
SyntaxError: invalid character in identifier
```

In [2]:

```
print(4 << 1)
#00000100左移动1位,变成00001000（8   相当于 * 2）
print(4 << 2)
#（4）00000100左移动2位,变成00001000（16  相当于 * 4）
print(4 >> 1)
#（4）00000100右移动1位,变成00000010（2   相当于 / 2）
print(4 >> 2)
#（4）00000100右移动2位,变成（1）00000001（1   相当于 / 4）
print(8 << 2)
#（8）00001000左移动2位,变成38)00100000（32  相当于 * 4）

print(4 & 8)
#00000100（4）
#00001000（8）位与，没有都是1，则结果
#00000000（0）

print(4 | 8)
#00000100（4）
#00001000（8）位或，有一个是1，则结果
#00001100（12）


print(~6)#取反   三位来表示
#(进制)000000110
#(取反)111111001
#(减一)111111000
#除了首位其他位取反10000111这个是-7


print(10^20)#按位异或运算  如果两数位置不一样则1 一样则0
#00001010
#00010110
#00011100  这个是30
```

```
8
16
2
1
32
0
12
-7
30
```

运算符优先级

In [20]:

```
（）括号优先
①算术（**(冥运算)>* / % > + -）
②位运算<<,>>
③比较（关系）<,>,>=,<=,==,!=, is, is not
④布尔 and, or, not, in, not in
⑤赋值  =
```

```
  File "<ipython-input-20-03ea5787a1f9>", line 1
    （）括号优先
     ^
SyntaxError: invalid character in identifier
```

In [1]:

```
a=20
b=2
c=15
result_01 = (a - b) + c  # 先执行圆括号中的表达式，再执行相加运算
result_02 = a / b % c    # 先执行除法运算，再执行取余运算
result_03 = c ** b * a   # 先执行幂运算，再执行相乘运算
print(result_01)
print(result_02)
print(result_03)
```

```
33
10.0
4500
```

例子

In [5]:

```
num1 = int(input("请输入第一个整数:"))
num2 = int(input("请输入第一个整数:"))
sum = 0
for i in range(1,num1+1):          # 这里num1 是指从1到第一个数的累加
    sum+=i
print("从1到第一个数的累加求和结果为:",sum)
result1 = 2**num2-sum
print("2的第二个数次幂的结果减去从1到第一个数的累加求和结果为:",result1)
x = num1%2
y = num2%2
result2 = x==y      # 从上面的记录 == 为比较运算符号
print("第一个数和第二个数都能被2整除吗？",result2)
```

```
请输入第一个整数:8
请输入第一个整数:6
从1到第一个数的累加求和结果为: 36
2的第二个数次幂的结果减去从1到第一个数的累加求和结果为: 28
第一个数和第二个数都能被2整除吗？ True
```

math库

导入和 ceil floor fmof modf trunc

In [ ]:

```
# ① 调调用方法一

import math
print(math.ceil(10.2))#将小数部分一律向整数部分进位
print(math.ceil(10.8))
print(math.floor(10.8))#舍去小数，仅取整数部分
print(math.floor(10.2))

# ② 调调用方法二

from math import floor
floor(10.2)

print(math.fmod(7,4))# fmod对浮点数取余
#rem和mod函数是mtalab的？？
print(math.modf(12.3))#  取整数部分和小数部分 小数部分误差 计算机浮点运算
print(math.trunc(12.3))#  取整数部分
```

round函数

In [2]:

```
float('inf') + float('-inf')
print(round(10.0/3,4))
print(math.round(10.0/3,4))  # 这里不能加math
```

```
3.3333

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-2-5ec0e908067a> in <module>
      1 float('inf') + float('-inf')
      2 print(round(10.0/3,4))
----> 3 print(math.round(10.0/3,4))  # 这里不能加math

NameError: name 'math' is not defined
```

In [ ]:

In [ ]:

In [ ]:

```
```

In [ ]:

```
```

inf与nan函数

In [ ]:

```
print(math.inf) #inf无穷大  inf正无穷 -inf负无穷
'''nan代表Not A Numberfloat('inf') + 100（不是一个数），它并不等于0，因为nan不是一个数，所以相关计算

#当涉及 > 和 < 运算时， 所有数都比 -inf 大 ，所有数都比 +inf 小。
print(float('nan'))
print(math.nan)  #  两种表达方

print(float('inf') + 100)
print(float('inf') - 100)
print(float('inf') * 100)
print(float('inf') / 100)
print(float('-inf') + 100)
# 对于一个无穷大 加减乘除一百还是无穷大负  ，，无穷也是如此

print(float('inf') + float('-inf'))  #最后是nan
print(float('inf') / float('-inf'))  #最后是nan
print(float('-inf') - float('inf'))
print(float('nan') + 100)#所有涉及nan的操作，返回的都是nan

# 比较操作时，返回的都是False，哪怕两个float('nan')互相比较都不相等。
print(float('nan') > float('inf'))
print(float('nan') < float('inf'))
print(float('nan') < float('-inf'))
print(float('nan') == float('nan')) # 注意 这里是false

#Python中可以用math.isinf()与math.isnan()来判断数据是否为inf或nan。
print(math.isinf(float('inf')))
print(math.isinf(float('-inf')))
print(math.isnan(float('nan')))

print(math.isnan(100))
print(math.isinf(100))#判断是否无穷大或者无穷小
```