专题一：赋值语句

运算	解释
b=10	基本形式
a,b=10,20----(a,b)=(10,20)	元祖赋值
a,b=[10,20]	列表赋值
a,b="AB"	序列赋值
a,*b='hello'	序列解包（Python3）
a=b=c=10	多目标赋值
a+=1	增强赋值

- 序列赋值

可以利用一个序列给另一个序列赋值

```
>>> a,b=10,20
>>> a,b,c=10,20  #报错
>>> a,b=10,20,30  #报错
```

等号右面的值不能过多，也不能过少(等号两边的元素数量要相等)

- 高级赋值语句

```
>>> s="SPAN"
>>> a,b=s   #报错
>>> a,b=s[:2],s[2:]
>>> a
'SP'
>>> b
'AN'


>>> (a,b),c=s[:2],s[2:]
>>> (a,b),c=s[:2],s[1:]



a,b,c=range(3)

l=[1,2,3,4,5,6]
while l:
    a,l=l[:1],l[1:]
    print(a,l)
```

```
l=[[1,2,3],[4,5,6],[7,8,9]]
for (a,b,c) in l:
    print(a,b,c)
```

- 序列解包

```
*b：获取分配完之后的一些剩余数据---封装成列表进行输出
>>> a,*b=[1,2,3,4]
>>> *a,b,c=[1,2]   #a=[]
>>> a,*b,c=[1,2,3,4,5,6]
>>> *b,a,c=[1,2,3,4,5,6]
>>> *a,b,*c=[1,2,3,4,5,6]#报错
# 不能同时出现多个*（只能有一个）
>>> *b=[1,2,3]#报错
#    *不能参与基本赋值，只能序列赋值
```

- 多目标赋值

```
a=b=c=10
c=10
b=c  ===10
a=b  ===10
多目标赋值首地址是公用的
```

- 增强赋值

```
+=   -=   *=  /=   //=  %=  **=
1. 执行效率高
2. 被用作系统优化
3. 减少代码量（优雅）
```