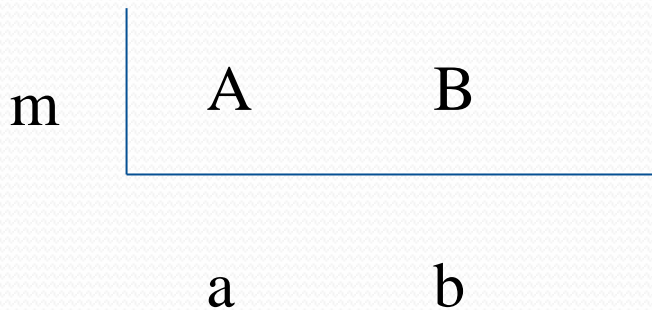


# 少儿编程

21工作室出品

# 辗转相减法

- 两个数A、B，它们的最大公因数是m，那么



- $A=ma$ ,  $B=mb$ ,  $A-B=m(a-b)$ , 所得结果是不会丢了公因数的。

- 操作方法，首先按照大小，大数放到A，小数放到B
- 可以得到A-B
- 因为A、B、A-B三个数都有共同的公因数，所以我们继续观察最小的两个数，按照大小，大数放到A，小数放到B
- 如此反复
- 最后会发现总会等于0，我们取等于0之前的那个数，就是最大公因数。

```
def xiangjian(p, q):  
    while p != q:  
        if p > q:  
            pass  
        else:  
            p, q = q, p  
            p, q = q, p - q  
    return p
```

# 辗转相除法

- 有了上面的经验，
- 乘法就是快速的加法、除法就是快速的减法，
- 一个数除以另一个数所得的余数，就是最快拿到包含两个数最大公因数的一个比较小的数
- 例：72、56，如果用辗转相减，会有如下过程：
- $72-56=16$
- $56-16=40$
- $40-16=24$
- $24-16=8$
- $16-8=8$
- $8-8=0$

- 而如果用除法：
- $72 \div 56 = 1 \dots 16$
- 和辗转相减道理相同，因为A、B、A-B三个数都有共同的公因数，所以我们继续观察最小的两个数，按照大小，大数放到A，小数放到B
- 继续 $A \div B$ ，取得余数
- 如此反复
- 最后会发现总会等于0，我们取等于0之前的那个数，就是最大公因数。



- 取被除数、除数、余数这三个数当中较小的两个数
- $72 \div 56 = 1 \dots 16$
- $56 \div 16 = 3 \dots 8$
- $16 \div 8 = 2 \dots 0$

```
- def xiangchu(p, q):  
    if p < q:  
        p, q = q, p  
    temp = p % q  
- while temp != 0:  
    p, q = q, p % q  
-     temp = p % q  
- return q
```