变成了 c，b 还是 b；执行完第二行代码之后，a 仍然是 c，b 变成了 a；执行完第三行代码之后，a 变成了 b，b 仍然是 a。过程结束。

位运算的题目基本上都带有靠经验累积才会做的特征，也就是在准备阶段需要做足够多的题，面试时才会有良好的感觉。

不用任何比较判断找出两个数中较大的数

【题目】

给定两个 32 位整数 a 和 b，返回 a 和 b 中较大的。

【要求】

不用任何比较判断。

【难度】

校 ★★★☆

【解答】

第一种方法。得到 a-b 的值的符号，就可以知道是返回 a 还是返回 b。具体请参看如下代码中的 getMax1 方法。

```
public int flip(int n) {
        return n ^ 1;
}

public int sign(int n) {
        return flip((n >> 31) & 1);
}

public int getMax1(int a, int b) {
        int c = a - b;
        int scA = sign(c);
        int scB = flip(scA);
        return a * scA + b * scB;
}
```

若为1则为0；
若为0则为1。

若为负则为1；
若为正则为0。

sign 函数的功能是返回整数 n 的符号，正数和 0 返回 1，负数则返回 0。flip 函数的功能是如果 n 为 1，返回 0，如果 n 为 0，返回 1。所以，如果 a-b 的结果为 0 或正数，那么

scA 为 1，scB 为 0；如果 a-b 的值为负数，那么 scA 为 0，scB 为 1。scA 和 scB 必有一个为 1，另一个必为 0。所以 return a * scA + b * scB;，就是根据 a-b 的值的状况，选择要么返回 a，要么返回 b。

但方法一是有局限性的，那就是如果 a-b 的值出现溢出，返回结果就不正确。

第二种方法可以彻底解决溢出的问题，也就是如下代码中的 getMax2 方法。

```
public int getMax2(int a, int b) {
        int c = a - b;
        int sa = sign(a);
        int sb = sign(b);
        int sc = sign(c);
        int difSab = sa ^ sb;
        int sameSab = flip(difSab);
        int returnA = difSab * sa + sameSab * sc;
        int returnB = flip(returnA);
        return a * returnA + b * returnB;
}
```

解释一下 getMax2 方法。

如果 a 的符号与 b 的符号不同（difSab==1,sameSab==0），则有：

- 如果 a 为 0 或正，那么 b 为负（sa==1,sb==0），应该返回 a；
- 如果 a 为负，那么 b 为 0 或正（sa==0,sb==1），应该返回 b。

如果 a 的符号与 b 的符号相同（difSab==0,sameSab==1），这种情况下，a-b 的值绝对不会溢出：

- 如果 a-b 为 0 或正（sc==1），返回 a；
- 如果 a-b 为负（sc==0），返回 b；

综上所述，应该返回 a * (difSab * sa + sameSab * sc) + b * flip(difSab * sa + sameSab * sc)。

只用位运算不用算术运算实现整数的加减乘除运算

【题目】

给定两个 32 位整数 a 和 b，可正、可负、可 0。不能使用算术运算符，分别实现 a 和 b 的加减乘除运算。

【要求】

如果给定的 a 和 b 执行加减乘除的某些结果本来就会导致数据的溢出，那么你实现的