C 高精度减法

分析

类比加法，同样建议倒序并左对齐，即123456转换为{'6','5','4','3','2','1','\0','\0',.....}
所以需要写一个函数用来倒序字符串

然后，减法进行的时候分两种情况：
①a >= b，结果是非负的，不需要加负号
②a < b，结果是负的，需要加负号，而且a-b=-(b-a)，只要ab颠倒一下就可以转化为①的问题，按①运算后只需要再加个负号即可
需要写一个函数来比较两个数的大小

实现

```c
#include <stdio.h>
#include <string.h>

void reverse(char *a){
    int l = strlen(a);
    int h = l / 2;
    int i;
    char c;
    for(i = 0; i < h; i++){
        c = a[i];
        a[i] = a[l - i - 1];
        a[l - i - 1] = c;
    }
}

int GEqual(char *a, char *b){
    int al = strlen(a);
    int bl = strlen(b);
    if(al > bl) return 1;
    if(al < bl) return 0;
    return strcmp(a, b) >= 0;
}

char* sub(char *a, char *b){
    if(GEqual(a, b)){
        int al, bl, ai, bi, i, s, up = 0;
        al = strlen(a);
        //模拟竖式减法
        for(i = 0; i < al; i++){
            //取得a的第i位数字
            ai = a[i] - '0';
            //如果b的对应位没了，就认为b的这一位是0，否则取得b的第i位数字
            bi = b[i] == '\0' ? 0 : (b[i] - '0');
            //加上退位，算出结果
            s = up + ai - bi;
            up = 0;
            //如果小于零，退位，设up=-1
            if(s < 0){
                s += 10;
                up = -1;
            }
            a[i] = s + '0';
        }
        //从高位遍历到低位，去除多余的前导0
        for(i = al - 1; i > 0; i--){
            //遇到0就设为'\0'
            if(a[i] == '0'){
                a[i] = '\0';
            }
            else{
                //一旦遇到不是0的，就退出循环
                break;
            }
        }
```

```
        return a;
    }
    else{
        sub(b, a);//如果a<b，两数交换，递归
        int l = strlen(b);
        b[l] = '-';//末尾加一个负号
        return b;
    }
}

int main(){
    char a[233] = {0};
    char b[233] = {0};
    scanf("%s", &a);
    scanf("%s", &b);
    //倒序
    reverse(a);
    reverse(b);
    //进行运算
    char* c = sub(a, b);
    //结果再倒过来变成正序的
    reverse(c);
    printf("%s", c);
}
```