高精度减法

示例：计算12345-999939

算法思路

1、将数字按字符串进行逆序存储，我们可以得到两个char数组

数组A	5	4	3	2	1	
数组B	9	3	9	9	9	9

2、判断正负，并交换位置

a、比较两个数组的长短，短的减长的则为负
b、相同长度的直接比较字符串的大小strcmp，当s1小于s2时，为负。
c、记录最大长度
d、记录正负号

很明显，本题为负数。我们需要将两个数组进行交换

数组A	9	3	9	9	9	9
数组B	5	4	3	2	1	

3、将字符串数组转为数字数组，方法就是使用数组中的每一个字符减去'0'，进而我们得到了两个新的数组

数组A	9	3	9	9	9	9
数组B	5	4	3	2	1	

4、从索引0开始对位相减，同时需要处理退位

数组A	9	3	9	9	9	9		9	3	9	9	9	9
数组B	5	4	3	2	1			5	4	3	2	1	
结果	4	-1	6	7	8	9		4	9	5	7	8	9

5、需要判断最高位是否为0，如果为0，则需要将最大长度减1。

6、因为我们存储数组的时候是按照逆序进行存储的，因此，在打印的时候，我们需要将其修正过来，因此我们同样需要逆序打印。

结果	-	9	8	7	5	9	4

编码

```
#include <bits/stdc++.h>
```

```
using namespace std;

string numA, numB;
int lenA, lenB, a[201], b[201];//105位数字，是一个极大的数字
bool sign = false; //标记是否答案为负数

void HighAccuracyAlgorithm() {
    lenA = numA.size();
    lenB = numB.size();
    if (lenA < lenB) { //a小于b，结果为负
       sign = true;
        swap(numA, numB); //交换两数的位置，保证大数在前
       swap(lenA, lenB); //交换两数的长度
   }
        //注意字符串比较的问题，例如"10"和"2"比较
   else if (lenA == lenB) {
        if (numA.compare(numB) < 0) {
            sign = true;
            swap(numA, numB); //交换两数的位置，保证大数在前
       }
    }
    //逆序存储
   for (int i = 0; i < lenA; i++) {
        a[i] = numA[lenA - 1 - i] - '0';
    }
    for (int i = 0; i < lenB; i++) {
        b[i] = numB[lenB - 1 - i] - '0';
    }
    for (int i = 0; i < lenA; i++) {
        a[i] -= b[i]; //按位进行减法运算
       if (a[i] < 0) {
            a[i + 1]--; //高位减1
            a[i] += 10; //低位加10
        }
    }
    while (lenA > 1 && a[lenA - 1] == 0) { //处理最高位的退位
       lenA--;
    }
    if (sign) { //当前值为负数
       cout << "-";
    }
    //希望大家连接字符串
   string res = "";
    for (int i = lenA - 1; i >= 0; i--) {
        res += a[i] + '0';
    }
    //输出结果
   printf("%s", res.c_str());
```

```
}

int main(int argc, char **argv) {
    cin >> numA >> numB;
    HighAccuracyAlgorithm();
    return 0;
}
```