## 7 整数反转

思路：源自官网

<https://leetcode-cn.com/problems/reverse-integer/solution/zheng-shu-fan-zhuan-by-leetcode/>

比如要把数字123，反转成数字321。思路就是从数字末尾取处一个数字，然后放在头部，有点像pop和push

取出：

int pop = x % 10;

x /= 10;

放入：

rev = rev \* 10 + pop;

问题就在这个rev = rev \* 10 + pop;这个地方，要考虑整形溢出的问题，什么时候会溢出呢？

如果rev > INT\_MAX / 10, 那铁定会溢出

如果rev < INT\_MAX / 10, 那铁定不溢出

如果rev == INT\_MAX / 10, 这时候就是+7、-8两种情况之一了

class Solution {

public:

    int reverse(int x) {

        int pop, rev = 0;

        int TOP = INT\_MAX / 10;

        while(x) {

            pop = x % 10;

            x /= 10;

            if(rev > TOP || (rev == TOP && pop > 7)) return 0;

            if(rev < -TOP || (rev == -TOP && pop < -8)) return 0;

            rev = rev \* 10 + pop;

        }

        return rev;

    }

};

/\*

思路：源自官网

https://leetcode-cn.com/problems/reverse-integer/solution/zheng-shu-fan-zhuan-by-leetcode/

比如要把数字123，反转成数字321。思路就是从数字末尾取处一个数字，然后放在头部，有点像pop和push

取出：

int pop = x % 10;

x /= 10;

放入：

rev = rev \* 10 + pop;

问题就在这个rev = rev \* 10 + pop;这个地方，要考虑整形溢出的问题，什么时候会溢出呢？

如果rev > INT\_MAX / 10, 那铁定会溢出

如果rev < INT\_MAX / 10, 那铁定不溢出

如果rev == INT\_MAX / 10, 这时候就是+7、-8两种情况之一了

\*/

## 8 字符串转换整数atoi

class Solution {

public:

    int myAtoi(string s) {

        int i=0;

        int sign = 1, base = 0;

        while(s[i] == ' ') ++i;

        if(s[i] == '+') {

            sign = 1;

            ++i;

        }

        else if(s[i] == '-') {

            sign = -1;

            ++i;

        }

        while(i < s.size() && s[i] >= '0' && s[i] <= '9') {

            int temp = s[i] - '0';

            if(base > INT\_MAX/10 || (base == INT\_MAX/10 && temp > 7)) {

                return sign > 0 ? INT\_MAX : INT\_MIN;

            }

            base = base \* 10 + temp;

            ++i;

        }

        return sign \* base;

    }

};

/\*

思路：

顺序执行，先过滤空格 : while(s[i] == ' ') ++i;

接下来是符号！

接下来是数字，还是考虑整形溢出即可。

这里7和8的地方很微妙也很巧妙

\*/