# 题目

一次位翻转定义为将数字 x 二进制中的一个位进行翻转操作，即将 0 变成 1 ，或者将 1 变成 0 。

比方说，x = 7 ，二进制表示为 111 ，我们可以选择任意一个位（包含没有显示的前导 0 ）并进行翻转。比方说我们可以翻转最右边一位得到 110 ，或者翻转右边起第二位得到 101 ，或者翻转右边起第五位（这一位是前导 0 ）得到 10111 等等。

给你两个整数 start 和 goal ，请你返回将 start 转变成 goal 的 最少位翻转 次数。

示例 1：

输入：start = 10, goal = 7

输出：3

解释：10 和 7 的二进制表示分别为 1010 和 0111 。我们可以通过 3 步将 10 转变成 7 ：

- 翻转右边起第一位得到：1010 -> 1011 。

- 翻转右边起第三位：1011 -> 1111 。

- 翻转右边起第四位：1111 -> 0111 。

我们无法在 3 步内将 10 转变成 7 。所以我们返回 3 。

示例 2：

输入：start = 3, goal = 4

输出：3

解释：3 和 4 的二进制表示分别为 011 和 100 。我们可以通过 3 步将 3 转变成 4 ：

- 翻转右边起第一位：011 -> 010 。

- 翻转右边起第二位：010 -> 000 。

- 翻转右边起第三位：000 -> 100 。

我们无法在 3 步内将 3 变成 4 。所以我们返回 3 。

提示：

0 <= start, goal <= 109

相似题目：

面试题 05.06. 整数转换

# 分析

class Solution {

public:

int minBitFlips(int start, int goal) {

bitset<32> m,n;

int cnt = 0;

m = start;

n = goal;

for (int i = 0; i < 32; i++) {

//统计二者不相同的数量，即需要反转的数量（不管是哪个翻转）

if (m[i] != n[i]) cnt++;

}

return cnt;

}

};