# 题目

颠倒给定的 32 位无符号整数的二进制位。

**示例 1：**

输入: 00000010100101000001111010011100

输出: 00111001011110000010100101000000

解释: 输入的二进制串00000010100101000001111010011100表示无符号整数43261596，因此返回 964176192，其二进制表示形式为 00111001011110000010100101000000。

**示例 2：**

输入：11111111111111111111111111111101

输出：10111111111111111111111111111111

解释：输入的二进制串 11111111111111111111111111111101 表示无符号整数 4294967293，因此返回 3221225471 其二进制表示形式为 10111111111111111111111111111111 。

**提示：**

请注意，在某些语言（如 Java）中，没有无符号整数类型。在这种情况下，输入和输出都将被指定为有符号整数类型，并且不应影响您的实现，因为无论整数是有符号的还是无符号的，其内部的二进制表示形式都是相同的。

在 Java 中，编译器使用二进制补码记法来表示有符号整数。因此，在上面的 示例 2 中，输入表示有符号整数 -3，输出表示有符号整数 -1073741825。

**进阶:**

如果多次调用这个函数，你将如何优化你的算法？

# 分析

## 方法一：逐位颠倒

class Solution {

public:

uint32\_t reverseBits(uint32\_t n) {

uint32\_t ret = 0, pos = 31;

while(n!=0)

{

ret += (n & 1) << pos;

n = n >> 1;

pos = pos -1;

}

return ret;

}

};

复杂度分析

时间复杂度：O(log2N)。在算法中，我们有一个循环来迭代输入的最高非零位，即log2N。

空间复杂度：O(1)，因为不管输入是什么，内存的消耗是固定的。