**规律：**

1. **涉及次数可以考虑哈希表**
2. **涉及环考虑快慢指针**

**注意事项：**

1. **检查条件中的特殊情况，比如为0，为空，为NULL时怎么考虑，**
2. **考虑与条件完全相反的方向,比如两个字符串，看第二个是否是第一个的一部分，要考虑第二个比第一个长**

**1.递归函数解决逆序输出字符串**

void reverse(char\* str)

{

char tmp = \*str;

int len = strlen(str); //1

\*str = \*(str + len - 1); //2

\*(str + len - 1) = '\0'; //3

if(len>1) //老师这里的条件是strlen(str+1)>=2

reverse(str + 1); //4

\*(str+len-1)= tmp;

}

**2.交换两数但不引入新的变量：**

a=a+b

b=a-b; 可能会溢出

a=a-b;

a=a^b;

b=a^b;

a=a^b;

1. **求一个整数存储在内存中的二进制里1的个数**

#include <stdio.h>

循环移位

int count\_bits\_method1(int num) {

int count = 0;

int bits = sizeof(num) \* 8; // 整数的总位数

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

if (num & 1) { // 检查最低位是否为1 未考虑负整数，且低效

count++;

}

num >>= 1; // 右移一位

}

return count;

}

高效：位运算符

int count\_bits\_method3(int num) {

int count = 0;

unsigned int unum = (unsigned int)num;

while (unum) {

count++;

unum = unum & (unum - 1); // 每次消除最低位的1

}

return count;

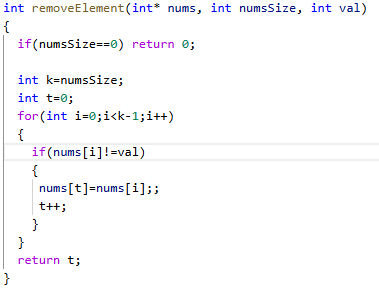
}

%2，/2：和10进制数里求每位的数字一样，但是记得考虑负数，这时候传参应该用无符号整数来计算

去除二进制数的某一位 (num>>i)&1;

1. 双指针

4.1把数组里的特殊元素删去但元素直接相对位置不改变：一个指针遍历数组，一个指针记录特殊元素（或者记录的是有效元素）下标，删去（两个指针共同遍历，但是当有特殊元素出现后，特殊指针跳过它）；

（！！！：不是k-1，应该遍历整个数组，-1去掉）

快慢指针（Floyd 判圈算法）：

链表环问题：slow指向头结点，fast先走一步，slow每次走一步，fast每次走两步，若有环，那么两者一定会指向同一个位置。

快乐数（各位数平方相加得新数，新数重复这个过程，结果是1则是快乐数）