

一篇文章搞懂面试中leetcode位操作算法题



六尺帐篷 (/u/f8e9b1c246f1)
2017.07.20 15:58* 字数 1831 阅读 246 评论 0 喜欢 12

(/u/f8e9b1c246f1)

编辑文章 (/writer#/notebooks/14296477/notes/14645500)

- Single Number落单的数
- 落单的数 IISingle Number II
- Single Number IIIL落单的数 III
- Number of 1 Bits
- Counting Bits
- Reverse Bits
- Missing Number
- Sum of Two Integers
- Power of Two
- Power of Four

本文将根据题目总结常用的位操作常用的解决算法问题的技巧
如读者对基本的位操作概念还不熟悉，可以先参考笔者的文章浅谈程序设计中的位操作
<http://www.jianshu.com/p/294fc6605bb1> (<http://www.jianshu.com/p/294fc6605bb1>)

Single Number落单的数

给出 $2 * n + 1$ 个的数字，除其中一个数字之外其他每个数字均出现两次，找到这个数字。

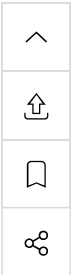
思路：
一个数字和自己进行异或操作会是0，由于异或操作满足交换定律，一个数和0进行异或操作还是本身。所以这道题目的思路就来了，将所有出现两次的数异或就都变成了0，最后剩的那个数和0异或就还是本身。直接将数组所有数异或，就可以找出那个落单的数

```
public class Solution {
    public int singleNumber(int[] nums) {
        int res = 0;
        for(int i=0;i<nums.length;i++)
            res ^= nums[i];
        return res;
    }
}
```

落单的数 IISingle Number II

给出 $3 * n + 1$ 个的数字，除其中一个数字之外其他每个数字均出现三次，找到这个数字。

思路：
java中int是32位的，所以我们利用一个32的数组，分别记录每一位1的情况，如果出现三次就清0，最后留下来的就是那个只出现1次的数字在那一位上的情况，然后进行移位复原




```
public class Solution {  
    // you need to treat n as an unsigned value  
    public int hammingWeight(int n) {  
        int ones = 0;  
        while(n!=0) {  
            ones += (n & 1);  
            n = n >> 1;  
        }  
        return ones;  
    }  
}
```

还有一种方法，我们知道 $n \& (n-1)$ 会把 n 中最后为1的一位变成0。
所以我们调用 $n \& (n-1)$ ，看看调几次这个数会变成0，就说明有几个1。

```
public class Solution {  
    // you need to treat n as an unsigned value  
    public int hammingWeight(int n) {  
        int sum = 0;  
  
        while(n != 0) {  
            sum++;  
  
            n = n & (n-1);  
        }  
        return sum;  
    }  
}
```

Counting Bits

Given a non negative integer number num. For every numbers i in the range $0 \leq i \leq \text{num}$ calculate the number of 1's in their binary representation and return them as an array.

Example:

For num = 5 you should return [0,1,1,2,1,2].

思路:

我们当然可以利用上一题的方法，直接每个数计算一次
但也发现是存在规律的



		count	
0	0	0	
1	1	1	
2	10	1	[2-3]
3	11	2	
4	100	1	
5	101	2	[4-7]
6	110	2	
7	111	3	
8	1000	1	
9	1001	2	
10	1010	2	
11	1011	3	[8-15]
12	1100	2	
13	1101	3	
14	1110	3	
15	1111	4	
16	10000		

image.png

```

public class Solution {
    public int[] countBits(int num) {
        int[] res = new int[num+1];

        for(int i=1;i<=num;i++)
            res[i] = res[i>>1] + (i & 1);

        return res;
    }
}

```

Reverse Bits

Reverse bits of a given 32 bits unsigned integer.

For example, given input 43261596 (represented in binary as 00000010100101000001111010011100), return 964176192 (represented in binary as 00111001011110000010100101000000).

思路:

利用位操作, 先交换相邻的两位, 再交换的四位, 再交换相邻的八位。

举个例子;

我们交换12345678

可以先变成 21436587

再变成43218765

最后87654321, 交换成功

对于32位也是如此的思路。

关键如何用位操作实现, 首先交换两位的话, 可以先分别取出前一位

x & (10101010101010101010101010101010。。。。)



换成16进制就是

$x \& (0xaaaaaaaa)$ 取出前一位，因为要与要有后一位交换，所以右移一位，因为只是单纯的交换，所以是逻辑右移

```
(x & 0xaaaaaaaa) >>> 1
```

然后对后一位也进行相应的操作，很容易得出

```
(x & 0x55555555) << 1
```

最后分别将前一位后一位合起来，使用或操作就可以了

所以，第一次交换后

```
x = ((x & 0xaaaaaaaa) >>> 1) | ((x & 0x55555555) << 1);
```

然后进行第二次交换：

取出前两位

$x \& (1100110011001100.....)$ 也就是 $x \& 0xcccccccc$ 。

后面的步骤都是一样的思路

```
x = ((x & 0xcccccccc) >>> 2) | ((x & 0x33333333) << 2);
```

第三次交换

```
x = ((x & 0xf0f0f0f0) >>> 4) | ((x & 0x0f0f0f0f) << 4);
```

第四次交换

```
x = ((x & 0xffff0000) >>> 8) | ((x & 0x00ff00ff) << 8);
```

第四次交换

```
x = ((x & 0xffff0000) >>> 16) | ((x & 0x0000ffff) << 16);
```

交换成功

代码就是上面的交换的过程

```
public class Solution {
    // you need treat n as an unsigned value
    public int reverseBits(int x) {
        x = ((x & 0xaaaaaaaa) >>> 1) | ((x & 0x55555555) << 1);
        x = ((x & 0xcccccccc) >>> 2) | ((x & 0x33333333) << 2);
        x = ((x & 0xf0f0f0f0) >>> 4) | ((x & 0x0f0f0f0f) << 4);
        x = ((x & 0xffff0000) >>> 8) | ((x & 0x00ff00ff) << 8);
        x = ((x & 0xffff0000) >>> 16) | ((x & 0x0000ffff) << 16);
        return x;
    }
}
```

Missing Number

给出一个包含 $0 \dots N$ 中 N 个数的序列，找出 $0 \dots N$ 中没有出现在序列中的那个数。

```
public class Solution {
    public int missingNumber(int[] nums) {
        int xor = 0, i = 0;
        for (i = 0; i < nums.length; i++) {
            xor = xor ^ i ^ nums[i];
        }

        return xor ^ i;
    }
}
```

Sum of Two Integers

位操作实现 $A+B$ 的操作是常见的算法题。

lintcode 上就有一道容易题是这样。



```

class Solution {
    /*
     * param a: The first integer
     * param b: The second integer
     * return: The sum of a and b
     */
    public int aplusb(int a, int b) {
        // write your code here, try to do it without arithmetic operators.
        if(a==0)return b;
        if(b==0)return a;
        int x1 = a^b;
        int x2 = (a&b)<<1;
        return aplusb(x1,x2);
    }
};

```

上述代码就实现了不用+操作符，利用位操作实现两个数的相加操作。

现在我们来讲解位操作实现两个数相加的原理

首先，十进制中，我们知道，7+8，不进位和是5，进位是1，然后我们可以根据不进位和和进位5+1*10算出最后的结果15。

类似二进制也可以采取这种方法

比如

a = 3, b = 6

a : 0011

b : 0110

不进位和：0101 也就是5

进位：0010 也就是2

所以a+b变成5 + (2<<1)

5 0101

2<<1 0100

不进位和 0001 = 1

进位 0100 = 4

因此 a + b就变成了1 + 4 << 1

然后有

1 0001

4<<1 1000

不进位和 1001 = 9

进位 0000 = 0

当时进位为0时，不进位和为9即a + b之和。

可以发现上述是一个递归的过程，所以也就不难写出代码了。求两个数的不进位和实际上就是将两个数异或操作即可。

Power of Two

Given an integer, write a function to determine if it is a power of two.

要为2的次方

1, 2, 4, 8, 16

也就是每位分别单独为1

1

10

100

1000

10000

所以n & (n-1)必须为0

```

public class Solution {
    public boolean isPowerOfTwo(int n) {
        if(n<=0)
            return false;
        return (n & (n-1)) == 0;
    }
}

```



Power of Four

Given an integer (signed 32 bits), write a function to check whether it is a power of 4.

按照上一题的思路，我们先列举出几个4的次方数，观察他们的规律

1
100
10000
1000000

我们发现不仅要2的次方的性质，还要满足 1所在的位必须是奇数位，所以我们取出奇数位，由于，1只在奇数位，所以取出奇数位后，应该还和原来的数相等
取奇数位的方法在反转bit那题中已经讲过，就是x & 0x55555555

```
public class Solution {
    public boolean isPowerOfFour(int num) {
        return (num > 0) && ((num & (num - 1)) == 0) && ((num & 0x55555555) == num);
    }
}
```

📖 数据结构与算法 (14296477) © 著作权归作者所有



六尺帐篷 (/u/f8e9b1c246f1)

写了 245418 字，被 15240 人关注，获得了 1429 个喜欢 (/u/f8e9b1c246f1)

如果觉得我的文章对您有用，请随意赞赏。您的支持将鼓励我继续创作！

赞赏支持

🤍 喜欢 12



更多分享

(http://cwb.assets.jianshu.io/notes/images/1464550



写下你的评论...

评论 关闭评论

智慧如你，不想发表一点想法咩~

被以下专题收入，发现更多相似内容

✚ 收入我的专题

 Leetcod... (/c/9447943bff28?utm_source=desktop&utm_medium=notes-included-collection)

⚙ 投稿管理

^



📌

🔖

🔗

http://www.jianshu.com/p/525a04acaf37

7/8

-  Android知识 (/c/3fde3b545a35?utm_source=desktop&utm_medium=notes-included-collection)
-  Android... (/c/5139d555c94d?utm_source=desktop&utm_medium=notes-included-collection)
-  程序员 (/c/NEt52a?utm_source=desktop&utm_medium=notes-included-collection)
-  首页投稿 (/c/bDHhpK?utm_source=desktop&utm_medium=notes-included-collection)
-  算法 (/c/d47f272c54c8?utm_source=desktop&utm_medium=notes-included-collection)
-  简单算法 (/c/f256fe9732d4?utm_source=desktop&utm_medium=notes-included-collection)

展开更多

推荐阅读 更多精彩内容 > (/)

LintCode 字符大小写排序 (/p/badafb18ab1a?utm_campaign=maleskine...

题目 给定一个只包含字母的字符串，按照先小写字母后大写字母的顺序进行排序。注意事项 小写字母或者大写字母他们之间不一定要保持在原始字符串中的相对位置。样例给出"abAcD"，一个可能的答案为"acBA..."

六尺帐篷 (/u/f8e9b1c246f1?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

Top 10 Methods for Java Arrays (/p/e7ff7421e110?utm_campaign=mal...

1 声明一个array 2 打印一个array 3 从array创建一个list 4 检查array中是否存在某个元素 5 连接两个array 6 Declare an array inline 7 将一个list转为array 8 array转为set

六尺帐篷 (/u/f8e9b1c246f1?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

从入门到大神，你要的PPT快捷操作都在这了... (/p/100... (/p/1004df57a6f3?utm_campaign=maleskine&utm_content=note&utr

最近和一个设计创意学院的同学聊天，看他把PS、AI玩得6到飞起，非常佩服，问他是不是有大佬手把手教学，他说，并没有。软件操作这种东西，从入门...

蹇之途 (/u/fb038c413759?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

是这些年坚持下来的小习惯让我变得不一样 (/p/80cd10... (/p/80cd104b623d?utm_campaign=maleskine&utm_content=note&utr

生活篇： 1.每天和爸爸妈妈视频 有些人可能不太相信，跟父母哪有那么多话要每天说，跟他们视频并不需要聊很久，短一分钟，长半小时，其实全是你来决...

我要笑遍全世界 (/u/e26492f5dab8?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

和母亲的最后55天 (/p/6a1c31bb6695?utm_campaig... (/p/6a1c31bb6695?utm_campaign=maleskine&utm_content=note&utr

9月16日 母亲因为腹痛已经在三门峡中心医院住院一周了，经过四次洗肠，ct，胃镜，肠镜各种检查，被折腾的不成样子，可最后依然没有查出病因。我人在...

刘都 (/u/e8ac278005e4?utm_campaign=maleskine&utm_content=user&utm_medium=pc_all_hots&utm_source=recommendation)

↑

↑

🔖

🔗