

位运算

n 的二进制表示中第 k 位数字是几 (0, 1)

(这个后面的快速幂会用到)

$$n = 15 = \overset{3}{(1)} \overset{2}{(1)} \overset{1}{(1)} \overset{0}{(1)}_2 \quad (k \text{ 从 } 0 \text{ 开始})$$

① 先把第 k 位移到最后一位 $n \gg k$

② 看最后一位是几 (二进制个位) $x \& 1$

lowbit:

lowbit(x): 返回 x 的最后一位 1 是多少

$$x = (1010)_2 \quad \text{lowbit}(x) = (10)_2 = 2$$

$$x = (101000)_2 \quad \text{lowbit}(x) = (1000)_2$$

$$\text{实现: } x \& (-x) = x \& (\sim x + 1)$$

设 $x \geq 0$, 则 $-x$ 是 x 的二进制按位取反后最低位 + 1 (符号位 0 → 1)

$$x: 0 \overset{|}{:} 1010 \quad -x: 1 \overset{|}{:} 010 \overset{+1}{1} = 1 \overset{|}{:} 0110$$

$x \& (-x) = 0 \overset{|}{:} 0010$ x 最低位的 1, 对 $-x$ 按位取反后最低位的 0, 则 0 → 1 变 1, 那么 $-x$ 按位取反 + 1 后, 01...1 变成 10...0

此时 x 与 $-x$ 关系: x 的最低位的 1 上对应 $-x$ 也为 1, 其余位上 x 为 0, $-x$ 为 1; $-x$ 为 0, x 为 1, 所以 & 后 只有最低位 1 仍为 1, 其余全为 0

```

1 #include <stdio.h>
2
3 void show_binary(int x) {
4     while(x) {
5         // x & 1相当于取出n二进制最低位
6         //注意! 这里的二进制是倒着输出的
7         // (100)2 -> 001
8         printf("%d", x & 1);
9         x = x >> 1;
10    }
11    printf("\n");
12    return ;
13 }
14
15 int main() {
16     int n; scanf("%d", &n);
17     show_binary(n);
18     show_binary(n & (~n));
19     show_binary(n & (~n + 1));
20     return 0;
21 }

```

801. 二进制中1的个数

题目

提交记录

讨论

题解

视频讲解

给定一个长度为 n 的数列，请你求出数列中每个数的二进制表示中 1 的个数。

输入格式

第一行包含整数 n 。

第二行包含 n 个整数，表示整个数列。

输出格式

共一行，包含 n 个整数，其中的第 i 个数表示数列中的第 i 个数的二进制表示中 1 的个数。

数据范围

$1 \leq n \leq 100000$,
 $0 \leq \text{数列中元素的值} \leq 10^9$

输入样例：

```
5
1 2 3 4 5
```

输出样例：

```
1 1 2 1 2
```

难度：

1s / 64MB

总通过数：

114203

总尝试数：

135123

来源：

模板题

算法标签

```
1 #include <stdio.h>
2 const int N = 1e6 + 10;
3 int n;
4
5 int main() {
6     scanf("%d", &n);
7     for(int i = 0; i < n; i++) {
8         int x, sum = 0;
9         scanf("%d", &x);
10        while(x) {
11            if(x & 1) sum += 1;
12            x = x >> 1;
13        }
14        printf("%d ", sum);
15    }
16
17    return 0;
18 }
```

```

23 // 优化
24 #include <stdio.h>
25 const int N = 1e6 + 10;
26 int n;
27
28 int lowbit(int x) {
29     return x & (-x);
30 }
31
32 int main() {
33     scanf("%d", &n);
34     for(int i = 0; i < n; i++) {
35         int x, sum = 0;
36         scanf("%d", &x);
37         while(x) {
38             // 当x > 0 时，每一次减去x的最低位1
39             // 这样减的次数就是x二进制表示中1的个数
40             x -= lowbit(x);
41             sum ++;
42         }
43         printf("%d ", sum);
44     }
45
46     return 0;
47 }

```

这里x每一次直接定位一个1，不用再迭代到1

