

## [4]位运算、基础数论

XY\_cpp  
20250804

# 课前须知

- 上课的时候专心听讲解，不要跟着老师抄代码，下课后独立完成。
- 不直接使用 AI 做题，AI 会做不等于自己会。  
    • 看完题解后，关闭题解独立练习。  
    • 练习中途遇到问题，应当分析题目及自己的思路，而非回忆题解或再次参考题解。
- 做过的题在课后需要重新独立完成，不参考老师的课件、代码，不参考自己以前的代码。

# 位运算

# 位运算

计算机内部使用二进制来表示数据。我们不仅希望只有计算机能够暗地里使用二进制操作，还希望我们自己也能使用二进制操作。

直接对二进制进行运算的操作被称为位运算。



就好比我们平时是在调度一个个班级（一个变量），而位运算可以对其到个人（每个二进制位）进行操作，这就大大提高了我们的工作效率。

# C++进制表示

C++内部支持直接定义如下四种进制的变量：

十进制: int x = 1234;

二进制: int x = 0b10011010010; *Binary*

八进制: int x = 02322; *Octal*

十六进制: int x = 0x4D2; *hexa decimal*

## 位运算

88 逻辑与

$$\begin{array}{r}
 1010 \\
 1100 \\
 \hline
 0110
 \end{array}$$



符号	描述	运算规则	示例
&	按位与	二进制 两个位都为 1 时才为 1	$(0b1010 \& 0b1100) == 0b1000$
	按位或	两个位一个为 1 就为 1	$(0b1010   0b1100) == 0b1110$
$\wedge$	按位异或	两个位不同时为 1	$(0b1010 \wedge 0b1100) == 0b0110$
$\sim$	按位取反	对每一位取反	$\sim 0b1010 == 0b0101$
$<<$	左移	所有位向左移, 末尾补零	$0b1010 << 1 == 0b10100$
$>>$	右移	所有位向右移, 末尾除去	$0b1010 >> 2 == 0b10$

注: 标红处的二进制只有4位, 并非C++中的实际情况。



$$\begin{aligned}
 x &\rightarrow x \cdot 2^y \\
 x &<< y
 \end{aligned}$$

## 异或的性质

$$\begin{array}{r}
 101101 \\
 000000 \\
 \hline
 110110
 \end{array}$$

1. 自反和维持原值

~~$$\begin{array}{l}
 (x \wedge 0) == x \\
 (x \wedge x) == 0
 \end{array}$$~~

2. 自反复性

$$(x \wedge (y \wedge y)) == x$$

$$x \wedge \overbrace{y \wedge y \wedge y \wedge y}^{\text{=0}} = x$$

3. 不进位的加法

$$(x \wedge y) \leq (x + y)$$

$$\begin{array}{r}
 110110 \\
 000101 \\
 \hline
 1
 \end{array}$$

$$\begin{cases}
 1+0=1 \\
 0+0=0 \\
 1+1=0
 \end{cases}$$

## 位运算的应用

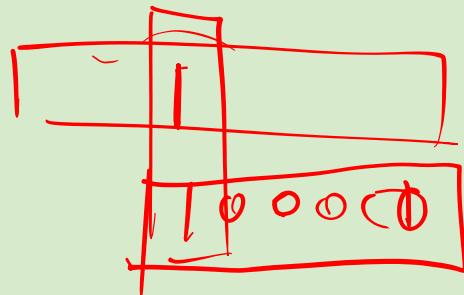
32 < 15

1. 判断一个数是否为2的次幂

$(x \& (x - 1)) == 0$

2. 清除最低位的1

$x \&= (x - 1)$



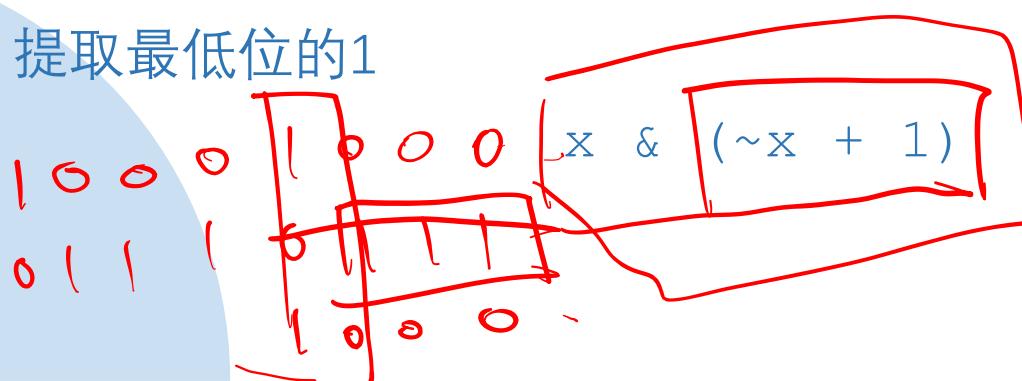
3. 判断第n位是否为1

$(x \& (1 << n)) != 0$

4. 指定第n位为1

$x |= (1 << n)$

5. 提取最低位的1

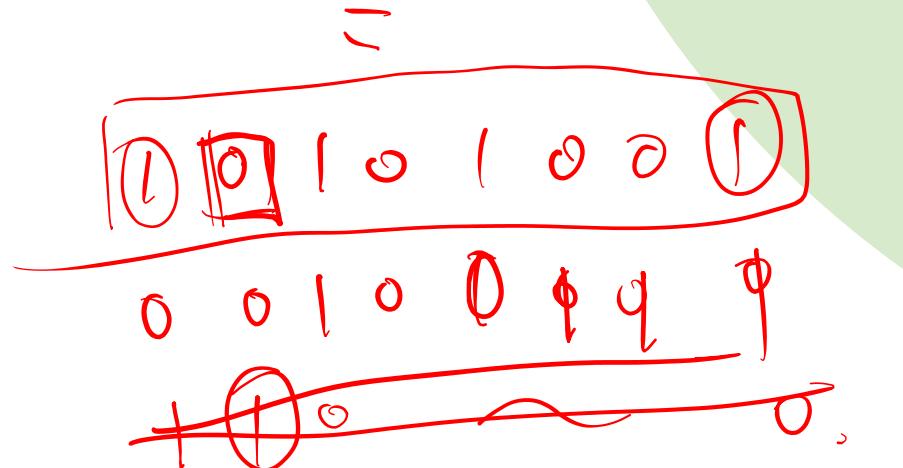


$x \& -x$

# 位运算的应用

## 7. 降序遍历 $m$ 的非空子集

```
for (int s = m; s; s = (s - 1) & m)
```



## 位运算笔试题

$$\begin{array}{r}
 1001 \quad 1101 \\
 10100 \quad 8+4+1 \\
 \hline
 8+1 \quad 8+4 \\
 \hline
 13
 \end{array}$$

1. (判断题) C++语言中, 表达式  $9 | 12$  的结果类型为  $\text{int}$ 、值为  $13$ 。  $\checkmark$

2. 如果a和b均为int类型的变量, 下列表达式不能正确判断“a等于b”的是  $\text{B}$ 。

- A.  $((a \geq b) \&\& (a \leq b))$
- B.  $((a >> 1) == (b >> 1))$
- C.  $((a + b) == (a + a))$
- D.  $((a ^ b) == 0)$

$$\begin{array}{r}
 101 \\
 10100 \quad 5 >> 1 = 2, 1100 \\
 10100 \quad 10100 \quad 8+4 \\
 \hline
 8+4 = 12, 1214
 \end{array}$$

3. 下面表达式的结果是布尔值true的是  $\text{B}$ 。

- A.  $4 \& (1 == 0)$  int
- B.  $(4 | 014) == 12$
- C.  $5 \&\& (3 == 1)$  false
- D.  $3 ^ (3 == 3)$

$$\begin{array}{r}
 101 \\
 10100 \quad 5 & 1 = 2, 1100 \\
 10100 \quad 10100 \quad 8+4 \\
 \hline
 8+4 = 12, 1214 \\
 \hline
 12
 \end{array}$$

## 例题-B4302

题目描述：

- $n$ 个数中只有一个数出现了奇数次
- 求这个数是多少
- $n \leq 10^5, x \leq 10^9$

思路：

- 首先考虑用map的做法
- 分析复杂度，思考有无更快的方法

$n \log n$

$map<int, int>$

$x^1 x^1 y^1 z^1 z^1$

## 例题-B4302

考虑使用异或。我们将所有数  
异或起来，便可以将出现偶数  
次的数字给消去，最终留下的  
就是出现奇数次的数。

```
#include <bits/stdc++.h>
using namespace std;
int main() {
    int n;
    cin >> n;
    int ans = 0;
    for (int i = 1; i <= n; i++) {
        int x;
        cin >> x;
        ans ^= x;
    }
    cout << ans << endl;
    return 0;
}
```

## 例题 - P1100

~~dp[i][j][k]~~ □

题目描述：

- 将一个整数  $x$  的高16位和低16位进行交换

- $0 \leq x \leq 2^{32} - 1$

$$2^{31} - 1$$

思路：



## 例题-P1100

```
high = x >> 16
```

```
low = x << 16
```

$0 \sim 2^{32} - 1$

在这道题的变量类型应该是什么？

0  
1

如果使用int，会发生什么事？

$-2^{31} \sim 2^{31} - 1$   $\rightarrow 2^{32}$

有没有其他写法？

unsigned

32位

# 数论

# 因数与倍数

一个整数  $a$  如果能被另一个整数  $b$  整除，那么我们称：

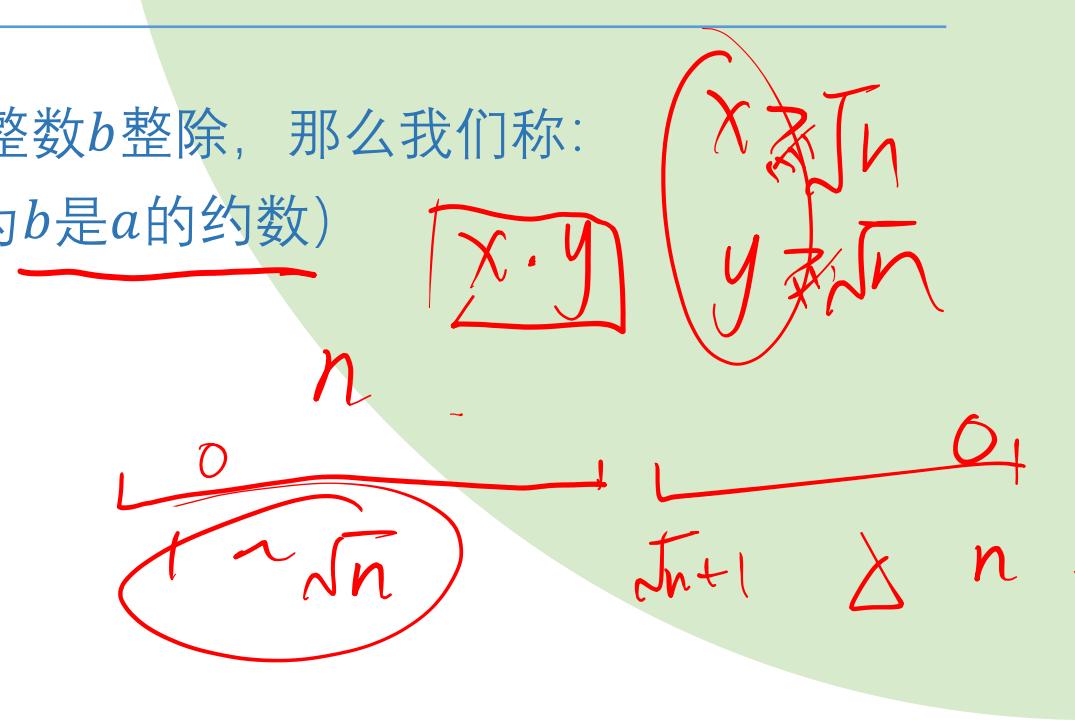
- $b$  是  $a$  的因数（也可以称为  $b$  是  $a$  的约数）
- $a$  是  $b$  的倍数

找出 12 的因数和倍数：

- 因数：1, 2, 3, 4, 6, 12
- 倍数：12, 24, 46, 48 ...

因数是成对出现的， $1 \times 12 = 2 \times 6 = 3 \times 4$ ，因此寻找一个数的因数只需要枚举到  $\sqrt{n}$ 。

$$\begin{aligned}
 1, 2, 3 &\leq \sqrt{12} = 2\sqrt{3} = 3 \cdot 4 \\
 4, 6, 12 &\geq \sqrt{12} = 2\sqrt{3} = 3 \cdot 4
 \end{aligned}$$



## 合数与质数

60 10  $2 \times 3 \times 6$

质数（素数）是指大于1、只有1和它本身两个因数的正整数。2是最小的质数，也是唯一的偶质数，所有其他质数都是奇数。

合数是指大于1，除了1和它本身外还有其他因数的正整数。

特殊说明：1既不是质数也不是合数。

判断一个数「是否为质数」等价于判断一个数「是否只有两个因数」，那么也枚举到 $\sqrt{n}$ 即可。

```
for (2 ~  $\sqrt{n}$ ) {  
    if (n % i == 0) {
```

## 例题-B3840

题目描述：

- 询问在  $A$  和  $B$  之间有多少个素数
- 数据范围不超过 1000

思路：

- 参考之前的讲解，自行思考方法

## 例题-B3840

直接枚举A至B中的每一个数，  
判断每一个数是否为质数即可。

然而这题数据范围较小，每个数都判断是否为质数尚可接受。但如果数据范围增大，或者题目输入组数变多，盲目地判断每一个数是否为质数会造成很大的开销。

因此，需要一种批量筛选质数的方法。

$\sqrt{x}$

```
bool is_prime(int x) {  
    if (x == 2) return true;  
    for (int i = 2; i * i <= x; i++) {  
        if (x % i == 0) return false;  
    }  
    return true;  
}
```

## 例题-B3840

该方法被称为埃氏质数筛法：

```
void seive(int n) {  
    is_prime[0] = is_prime[1] = false;  
    for (int i = 2; i <= n; i++) is_prime[i] = true;  
    for (int i = 2; i <= n; i++) {  
        if (is_prime[i]) {  
            prime.push_back(i);  
            if ((long long)i * i > n) continue;  
            for (int j = i * i; j <= n; j += i) is_prime[j] = false;  
        }  
    }  
}
```

该算法的时间复杂度为 $O(n \log n)$ 。

# 例题 - B3969

题目描述：

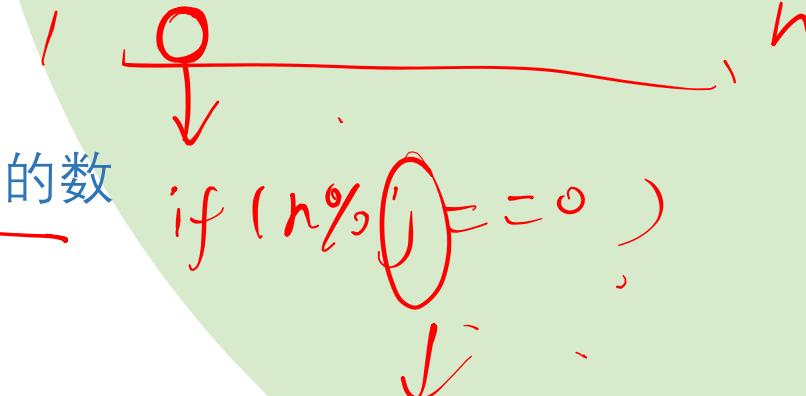
- 找出  $n$  以内最大质因子不超过  $B$  的数
- $1 \leq n, B \leq 10^6$

思路：

- 未经优化的埃氏筛的本质是在用一个质数去剔除它的所有倍数
- 那么反过来想，一个数会被它的所有质因子筛过
- 埃氏筛可以解决这题的问题！

洛谷

$$O(n^{\frac{1}{2}} \sqrt{n} \cdot \sqrt[4]{n})$$
$$O(n^{\frac{1}{2} + \frac{1}{4} + \frac{1}{4}})$$



## 例题-B3969

我们用 $a[i]$ 来存储某个数 $i$ 的最大质因子。

模拟埃氏筛的过程，每找到一个质数，它一定是当前最大的质数，就用该质数去更新它的所有倍数。

一定要注意循环的上限。

```
int n, b;
cin >> n >> b;
a[1] = 1;
for (int i = 2; i <= n; i++) {
    if (!a[i]) {
        for (int j = i; j <= n; j += i) a[j] = i;
    }
}
for (int i = 1; i <= n; i++) {
    ans += (a[i] <= b);
}
cout << ans << endl;
```

# 唯一分解定理

算术基本定理，又称为正整数的唯一分解定理，即：每个大于1的自然数，要么本身就是素数，要么可以写为2个或以上的素数的乘积，而且这些素因子按大小排列之后，写法仅有一种方式。

例如：

$$6936 = 2^3 \times 3 \times 17^2$$

$$1200 = 2^4 \times 3 \times 5^2$$

$$5207 = 41 \times 127$$

以上的分解方法都是唯一的。

## 例题-B3871

题目描述：

- 分解一个正整数 $n$
- 输出类似 $2^2 * 5$ 的形式
- $1 \leq n \leq 10^{12}$

思路：

- 在 $\sqrt{n}$ 范围内找到因数 $x$ 并统计指数 $k$
- 最后可能还会剩下大于 $\sqrt{n}$ 质因数，也需要统计
- 将所有统计好的质因数一并输出

## 例题-B3871

注意点：

- 该题的long long较多，替换为ll以便于代码的编写
- 在判断 $i \leq \sqrt{n}$ 时，需要特别注意 $i$ 是否会超过int
- 在输出的时候，将“ \* ”与后面的表达式看作一个整体，那么只有第一个不需要输出

```
ll n;
cin >> n;
vector<pair<ll, int>> fac;
for (int i = 2; 1ll * i * i <= n; i++) {
    if (n % i == 0) {
        int cnt = 0;
        while (n % i == 0) {
            n /= i;
            cnt++;
        }
        fac.push_back(make_pair(i, cnt));
    }
}
if (n != 1) fac.push_back(make_pair(n, 1));
sort(fac.begin(), fac.end());
for (int i = 0; i < fac.size(); i++) {
    if (i != 0) printf(" * ");
    ll x = fac[i].first;
    int y = fac[i].second;
    printf("%lld", x);
    if (y > 1) printf("^%d", y);
}
```

## 例题-B4070

---

<https://www.luogu.com.cn/problem/B4070>

请同学们自行阅读题目，进行如下尝试：

1. 尝试用一句话概括题意
2. 思考一个正整数 $a$ 可以被拆分为多少互不相同的数字

## 例题-B4070

思路：

- 这题本质上是在求一个数的质因数分解，随后询问能将这些质因数最多拆成多少个互不相同的数
- 因此首先进行质因数分解，求出每一个质因数的指数 $y$ 。随后，我们需要将 $y$ 拆分为若干个不同数字的和。
- 显然只需要找到一个 $k$ ，满足：
$$1 + 2 + 3 + \cdots + k - 1 + k > y$$
- 那么 $k - 1$ 就是我们要的最大拆分数量
- 把所有求出的 $k - 1$ 加起来即可