5-3 | 标签系统的核心实现--位运算

计算机中的二进制

计算机以二进制表示数据，以表示电路中的正反。在二进制下，一个位只有 0 和 1。逢二进一位。类似十进制下，一个位只有 0~9。逢十进一位。

二进制常用运算

位运算

位运算是直接对二进制数进行操作的运算，包括与（&）、或（|）、异或（^）、取反（~）等。

1. 与（&）（两位同时为 1，结果才为 1，否则为 0）

```
YAML
  1 0 0 1 1
& 1 1 0 0 1
-----------------------------
  1 0 0 0 1
```

1. 或（|）（参加运算的两个位只要有一个为 1，其值为 1 ）

```
YAML
  1 0 0 1 1
| 1 1 0 0 1
-----------------------------
  1 1 0 1 1
```

1. 异或（^）（参加运算的两个位只要不相同则为 1）

```
YAML
  1 0 0 1 1
^ 1 1 0 0 1
-----------------------------
  0 1 0 1 0
```

1. 取反（~）（0 则变为 1，1 则变为 0）

```
YAML
~ 1 0 0 1 1
```

```
----------------------------
  0 1 1 0 0
```

相关实战代码案例：

```YAML
int a = 5; //二进制表示为 101
int b = 3; //二进制表示为 011
int c = a & b; //与运算，结果为 001，即 1
int d = a | b; //或运算，结果为 111，即 7
int e = a ^ b; //异或运算，结果为 110，即 6
int f = ~a; //取反运算，结果为 11111111111111111111111111111010，即 -6
```

移位运算

移位运算包括左移（<<）、右移（>>）以及无符号右移（>>>，无符号右移就是右移之后，无论该数为正还是为负，右移之后左边都是补上 0），它们分别将一个二进制数左移或右移一定的位数。

代码案例：

```YAML
int a = 5; //二进制表示为 101
int b = a << 1; //左移一位，结果为 1010，即 10
int c = a >> 1; //右移一位，结果为 10，即 2
int d = a >>> 1; //无符号右移一位，结果为 10，即 2
```

位运算赋值运算符

位运算赋值运算符（&=、|=、^=、<<=、>>=、>>>=）是将位运算结果赋值给左操作数的运算。

```YAML
int a = 5; //二进制表示为 101
int b = 3; //二进制表示为 011
a &= b; //相当于 a = a & b
a |= b; //相当于 a = a | b
a ^= b; //相当于 a = a ^ b
a <<= 1; //相当于 a = a << 1
a >>= 1; //相当于 a = a >> 1
```

```
a >>>= 1; //相当于 a = a >>> 1
```

标签记录的实现原理

基于（或）|，与+取反（&~）去实现：

假设我们的标签是一个数字 16，转换为二进制就是 10000。

设置标签

1. 或（|）（参加运算的两个位只要有一个为 1，其值为 1 ）

```
YAML
  0 0 0 0 1（用户原先就有记录一个标签）
| 1 0 0 0 0
------------------------------
  1 0 0 0 1（用户记录上标签后，存入数据库）
```

取消标签

1. 与 + 取反（&~）（两位同时为 1，结果才为 1，否则为 0）

```
YAML
   1 1 0 0 1 （用户原先就有记录两个标签）
   1 0 0 0 0
~  0 1 1 1 1
&  0 1 1 1 1
------------------------------
   0 1 0 0 1 （取消了用户的数字为 16 的那个标签）
```