

线性基

Linear Basis

Idea: 从二进制的角度看待每个数，每一位可以类比为向量空间的一维，每一维只有 0 和 1 两种取值。 n 个数就是 n 个 64 维的向量（long long 为例），它们构成了 64 维向量空间的一个子空间。这个子空间中允许异或运算，两个向量相异或即每一维相异或。

线性基是一组数，构成这个子空间的一个基底，也即满足以下性质：

- 原集合的任一元素可由线性基中的一些元素相异或得到；
- 线性基是满足上述条件的最小集合；
- 线性基没有异或为 0 的子集：否则违背第二条；
- 线性基的选取元素方案不同，异或值不同：否则存在子集异或为 0。

Code:

$p[]$ 是存储线性基中的数组， $p[i]$ 存储最高位为 i 的数字。

```
1  LL p[70];
2  inline bool insert(LL x){
3      for(int i = 62; i >= 0; i--){
4          if((x >> i) == 0) continue;
5          if(!p[i]){ p[i] = x; return true; } // insert successfully
6          else x ^= p[i];
7      }
8      return false; // fail to insert
9  }
10 inline void norm(){ // normalization
11     for(int i = 62; i >= 0; i--){
12         if(p[i])
13             for(int j = 62; j > i; j--){
14                 if((p[j] >> i) & 1)
15                     p[j] ^= p[i];
16             }
17     }
18     inline bool exist(LL x){
19         for(int i = 62; i >= 0; i--){
20             if((x >> i) == 0) continue;
21             if(!p[i]) return false;
22             else x ^= p[i];
23         }
24         return true;
25     }
26     inline LL xorMax(){
27         LL res = 0;
28         for(int i = 62; i >= 0; i--){
29             res = max(res, res ^ p[i]);
30         }
31         return res;
32     }
33     inline LL xorMin(){
34         for(int i = 0; i <= 62; i++){
35             if(p[i]) return p[i];
36         }
37         return 0;
38     }
39     inline LL kthMin(LL k){ // kth minimum number (excluding 0)
40         // be sure normalized beforehand
41         LL res = 0;
42         vector<LL> tmp;
43         for(int i = 0; i <= 62; i++){
44             if(p[i]) tmp.push_back(p[i]);
45         }
46         if(k >= (lll << tmp.size())) return -1;
47         for(int i = tmp.size() - 1; i >= 0; i--){
48             if((k >> i) & 1) res ^= tmp[i];
49         }
50         return res;
51     }
52 }
```