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。

```cpp
LL p[70];
inline bool insert(LL x){
    for(int i = 62; i >= 0; i--){
        if((x >> i) == 0)   continue;
        if(!p[i]){ p[i] = x; return true; } // insert successfully
        else    x ^= p[i];
    }
    return false; // fail to insert
}
inline void norm(){ // normalization
    for(int i = 62; i >= 0; i--)
        if(p[i])
            for(int j = 62; j > i; j--)
                if((p[j] >> i) & 1)
                    p[j] ^= p[i];
}
inline bool exist(LL x){
    for(int i = 62; i >= 0; i--){
        if((x >> i) == 0)   continue;
        if(!p[i])   return false;
        else    x ^= p[i];
    }
    return true;
}
inline LL xorMax(){
    LL res = 0;
    for(int i = 62; i >= 0; i--)
        res = max(res, res ^ p[i]);
    return res;
}
inline LL xorMin(){
    for(int i = 0; i <= 62; i++)
        if(p[i])    return p[i];
    return 0;
}
inline LL kthMin(LL k){ // kth minimum number (excluding 0)
    // be sure normalized beforehand
    LL res = 0;
    vector<LL> tmp;
    for(int i = 0; i <= 62; i++)
        if(p[i])    tmp.push_back(p[i]);
    if(k >= (1ll << tmp.size()))    return -1;
    for(int i = tmp.size() - 1; i >= 0; i--)
```

```
44          if((k >> i) & 1)    res ^= tmp[i];
45      return res;
46  }
```