高维前缀和

简介

常用高维前缀和 求关于一个集合子集(或超集)的状态的和

对于一个t维的前缀和(每维n个数），用容斥原理来做，容斥的复杂度达到2^t,总复杂度为$n^t * 2^t$。

如果用这种思路：先做完1维前缀和，再做2维，再做3维.....直到t维，可以将复杂度降到$t * 2^t$。

```
//用上述思路实现3维前缀和的做法
for(int i=1;i<=n;i++)
    for(int j=1;j<=m;j++)
        for(int k=1;k<=p;k++)
        a[i][j][k]+=a[i-1][j][k];
for(int i=1;i<=n;i++)
    for(int j=1;j<=m;j++)
        for(int k=1;k<=p;k++)
        a[i][j][k]+=a[i][j-1][k];
for(int i=1;i<=n;i++)
    for(int j=1;j<=m;j++)
        for(int k=1;k<=p;k++)
        a[i][j][k]+=a[i][j][k-1];
```

模板

复杂度 $O(n * 2^n)$

```
#include <bits/stdc++.h>
using namespace std;
const int maxn=(1<<20)+5;
int w, dp[maxn];
//w表示最高维度
void solve(){
    //求子集
    for(int i = 0; i<w; ++i){
        for(int j = 0; j<(1<<i); ++j){
```

```
            if(j&(1<<i)) dp[j] += dp[j^(1<<i)];
        }
    }
    //求超集
    for(int i = 0; i<w; ++i){
        for(int j = 0; j<(1<<i); ++j){
            if( !(j&(1<<(i))) ) dp[j] +=  dp[j|(1<<(i))];
        }
    }
}
```

例题

1. EOJ-3300 奇数统计

题意

给n个数，求在n个数中选两个数(可重复)，使得这两个数的组合数是奇数，求总共有多少种取法。

思路

组合数 C_m^n 奇偶性判断： n & m == m 成立则组合数为奇数，n&m = m说明m是n的子集。

代码

```
#include<bits/stdc++.h>
using namespace std;
const int maxn = 1e6 + 10;
typedef long long ll;
ll a[maxn], sum[maxn];
int main()
{
    int T, n, x;
    cin >> T;
    while(T--)
    {
        scanf("%d", &n);
        memset(a, 0, sizeof(a));
        memset(sum, 0, sizeof(sum));
        for(int i = 0; i < n; i++)
        {
            scanf("%d", &a[i]);
            sum[a[i]]++;
        }
```

```
        int m = 17;
        for(int i = 0; i < m; i++)
        {
            for(int j = 0; j < (1<<m); j++)
            {
                if(j & (1<<i))
                    sum[j] += sum[j ^ (1<<i)];
            }
        }
        ll ans = 0;
        for(int i = 0; i < n; i++)
            ans += sum[a[i]];
        cout<<ans<<endl;
    }
    return 0;
}
```