**题意**

给出 一个n 然后 给出 2\*n 个数

可以重新排列成两行 然后 相邻的两个数 加起来 不能被三整除

可以上下相邻 也可以 左右相邻

先对所有数字对3取模

两个0不能相邻

1和2不能相邻



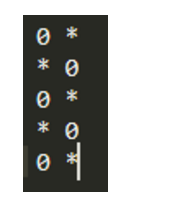
**思路**   
因为相加 根据同余定理 我们可以先把 每个数 模3

因为 可以重新排列 那么我们不妨 以最优的方式去排 看能不能得到 YES

很显然 ， 0 和 0 1 和 2 不能放在一起

也就是说

0 要摆放的话 肯定是错位摆放的 比如这样



星号 表示 空位 **那么很显然 0 的个数不能超过 n**

那么我们可以知道 因为 1 和 2 不能在一起

那么 可以是这样

用 两个 0 形成一条分界线 上面放1 下面放2



然后可以知道 如果 **只有两个 0 的话 那么很显然 只有当 1的个数 是奇数 并且 2 的个数 是偶数 才是成立的**

还有一个比较显然的是   
**假如 只有1 或者 只有2 的话 那么在满足 0的个数 < n的情况下 是恒成立的**

**那么很显然 如果0 的个数 <= 1 并且同时存在 1 和 2 的话 那么 肯定存在至少一对(1, 2) 相邻**

那只需要讨论 0 的个数 > 3 的情况了   
先 讨论 0 的个数 == 3 的情况 如果 0 的个数 == 3 那么 就可以多出一个0 去弥补 空位

也就是说 可以存在 奇数1 偶数2 或者 奇数2 偶数1 的情况   
那么也就是说 偶数1 偶数2 是不可行的

但实际上 不会存在这种情况 因为这种情况是 一奇两偶 相加是奇数 易知 数字总个数是2\*n 必然为偶数

所以这种情况 不用考虑 **也就是说 0的个数 == 3 并且 < n 的时候 是恒成立的**

很显然 当0的个数 >= 4的时候 也满足

#include <iostream>

#include <cstdio>

#include <cstring>

#include <cstring>

#include <cmath>

#include <algorithm>

using namespace std;

typedef long long ll;

int b[10];

int T,n,x;

bool check()

{

if(b[0]>n)return 0;

if(b[0]<2 && b[1] && b[2])return 0;

if(b[0]==2 && (b[1]&1)==0 && (b[2]&1)==0)return 0;

return 1;

}

int main()

{

scanf("%d",&T);

while(T--)

{

scanf("%d",&n);

for(int i=0;i<5;i++)

b[i]=0;

for(int i=1;i<=2;i++)

for(int j=1;j<=n;j++)

{

scanf("%d",&x);

b[x%3]++;

}

if(check())

puts("YES");

else puts("NO");

}

return 0;

}