A. Common Subsequence

如果有相同元素输出 1 个即可。

```
#include<bits/stdc++.h>
using namespace std;
int b[2005];
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        memset(b,0,sizeof(b));
        int n,m,a,f=0;
        scanf("%d%d",&n,&m);
        for(int i=1;i<=n;i++)scanf("%d",&a),b[a]=1;
        for(int i=1;i<=m;i++)
        {
            scanf("%d",&a);
            if(b[a])f=a;
        }
        if(!f)puts("NO");
        else printf("YES\n1 %d\n",f);
    }
    return 0;
}
```

B. Sequential Nim

只有一堆则先手必胜，从倒数第二堆开始逆推，如果进入下一堆先手必胜则 $a[i]=1$ 必败 $a[i]\neq1$ 必胜，若下一堆先手必败则肯定是必胜的。

```
#include<bits/stdc++.h>
using namespace std;
int a[100005];
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n,f=1;
```

```
        scanf("%d",&n);
        for(int i=1;i<=n;i++)scanf("%d",&a[i]);
        for(int i=n-1;i;i--)
        {
            if(f){if(a[i]>1)f=1;else f=0;}
            else f=1;
        }
        if(f)puts("First");
        else puts("Second");
    }
    return 0;
}
```

C. Prefix Flip

给两个 01 串□ $3n/2n$ 个操作内要把前者变成后者，一次操作可以选择一个前缀， 01 反转，且区间翻转。

从后往前扫某一位需要改变如果第一个字符和它相同则直接操作这个前缀，否则先操作 1，再操作这个前缀，可以控制在 $2n$ 次内□ $C1$ 可以 $O(n^2)$ 暴力翻转过□ $C2$ 要 $O(n\log n)$ 或线性（事实上写平衡树tle了）。于是可以记录 l,r 即区间 $[1,i]$ 在数组里的真实位置和当前的反转状态即可。

C1代码

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5+5;
char s[N],t[N];
vector<int>v;
void rev(int l)
{
    for(int i=0,j=l;i<j;i++,j--)swap(s[i],s[j]);
    for(int i=0;i<=l;i++)s[i]^=1;
}
int main()
{
    int T;
    scanf("%d",&T);
    while(T--)
    {
        int n,f=0;
        scanf("%d%s%s",&n,s,t);
        for(int i=0;s[i];i++)s[i]-='0',t[i]-='0';
        for(int i=n-1;i>=0;i--)
        {
```

```
            if(s[i]==t[i])continue;
            if(s[0]==s[i])v.push_back(i),rev(i);
            else
            {
                v.push_back(0),rev(0);
                v.push_back(i),rev(i);
            }
        }
        printf("%d\n",v.size());
        for(int x:v)printf("%d ",x+1);
        puts("");vector<int>().swap(v);
    }
    return 0;
}
```

C2代码

```
#include<bits/stdc++.h>
using namespace std;
const int N=1e5+5;
char s[N],t[N];
vector<int>v;
int main()
{
    int T;
    scanf("%d",&T);
    while(T--)
    {
        int n;
        scanf("%d%s%s",&n,s,t);
        for(int i=0;s[i];i++)s[i]-='0',t[i]-='0';
        int l=0,r=n-1,f=0,s0=s[0];
        for(int i=n-1;i;i--,(l<r)?r--:r++)
        {
            if(s[r]^f==t[i])continue;
            if(s0==s[r]^f)v.push_back(i),swap(l,r),s0^=1,f^=1;
            else
            {
                v.push_back(0);
                v.push_back(i),swap(l,r),f^=1;
            }
        }
        if(s0!=t[0])v.push_back(0);
        printf("%d ",v.size());
        for(int x:v)printf("%d ",x+1);
        puts("");vector<int>().swap(v);
    }
    return 0;
```

```
}
```

D. Unmerge

合并两个数组是归并排序的那个操作，给出一个长度 $2n$ 排列判定能不能由两个长度 n 的数组合并。

观察可以发现一个数字和后面连续的所有比它小的数字是必须在一起的，把它们打包为一组□dp看能不能选一些组长度和为 n

```
#include<bits/stdc++.h>
using namespace std;
const int N=2002;
int p[2*N],f[N];
vector<int>w;
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n;
        scanf("%d",&n);
        for(int i=1;i<=2*n;i++)scanf("%d",&p[i]);
        for(int i=1;i<=2*n;i++)
        {
            int maxn=p[i],t=i;
            while(i+1<=2*n&&p[i+1]<maxn)i++;
            w.push_back(i-t+1);
        }
        f[0]=1;
        for(int i=0;i<w.size();i++)
        {
            for(int j=n;j>=w[i];j--)
            f[j]=max(f[j],f[j-w[i]]);
        }
        if(f[n])puts("YES");
        else puts("NO");
        vector<int>().swap(w);
        for(int i=0;i<=n;i++)f[i]=0;
    }
    return 0;
}
```

E. Mastermind

给定长度 n 的数组 b ，可选的组成元素有 $n+1$ 种，问有没有数组 a 使得 $a[i]=b[i]$ 恰有 x 个位置，而数组重新排序后 $a[i]=b[i]$ 可以有 y 个（即构成有 y 个相同）。

明显与打乱后可以不匹配的位置个数相关的是出现最多的元素的次数 f ，越少越好），所以固定 x 个位置时用优先队列每次取出出现次数最多的。剩余 $n-x$ 个位置不match的下界将会是 $2f-(n-x)$ ，可以通过给数值排序后每个向后错 $\lfloor\frac{n-x}{2}\rfloor$ 位来实现。如果没有完全打乱不要紧，我们可以用一个没用过的数字 noUse 在填充 $n-y$ 个位置时补救一下，在那 $n-x$ 个中优先选取仍 $a[i]=b[i]$ 的。构造完毕后仍不满足条件即不能。

```
#include<bits/stdc++.h>
#define pii pair<int,int>
#define mp make_pair
#define fi first
#define se second
using namespace std;
const int N=1e5+10;
int b[N],num[N],a[N];
bool fix[N];
priority_queue<pii>q;
vector<pii>v;
int main()
{
    int t;
    scanf("%d",&t);
    while(t--)
    {
        int n,x,y,noUse;
        scanf("%d%d%d",&n,&x,&y);
        for(int i=1;i<=n;i++)scanf("%d",&b[i]),num[b[i]]++;
        for(int i=1;i<=n+1;i++)if(!num[i]){noUse=i;break;}
        for(int
i=1;i<=n;i++)if(num[b[i]])q.push(mp(num[b[i]],b[i])),num[b[i]]=0;
        int t=x;
        while(t--)
        {
            pii p=q.top();q.pop();
            if(p.fi-1)q.push(mp(p.fi-1,p.se));
            num[p.se]++;
        }
        for(int i=1;i<=n;i++)
        if(num[b[i]])num[b[i]]--,a[i]=b[i],fix[i]=1;else
v.push_back(mp(b[i],i));
        sort(v.begin(),v.end()),t=v.size();
        for(int i=0;i<v.size();i++)a[v[(i+t/2)%t].se]=v[i].fi;
        t=n-y;int i=1;
        for(int i=1;i<=n&&t;i++)if(!fix[i]&&a[i]==b[i])a[i]=noUse,t--;
```

```
        for(int i=1;i<=n&&t;i++)if(!fix[i]&&a[i]!=noUse)a[i]=noUse,t--;
        assert(t==0);
        int f=1;
        for(int i=1;i<=n;i++)if(!fix[i]&&a[i]==b[i])f=0;
        if(!f)puts("NO");
        else
        {
            puts("YES");
            for(int i=1;i<=n;i++)printf("%d ",a[i]);
            puts("");
        }
        priority_queue<pii>().swap(q);
        vector<pii>().swap(v);
        for(int i=0;i<=n+1;i++)num[i]=a[i]=b[i]=fix[i]=0;
    }
    return 0;
}
```