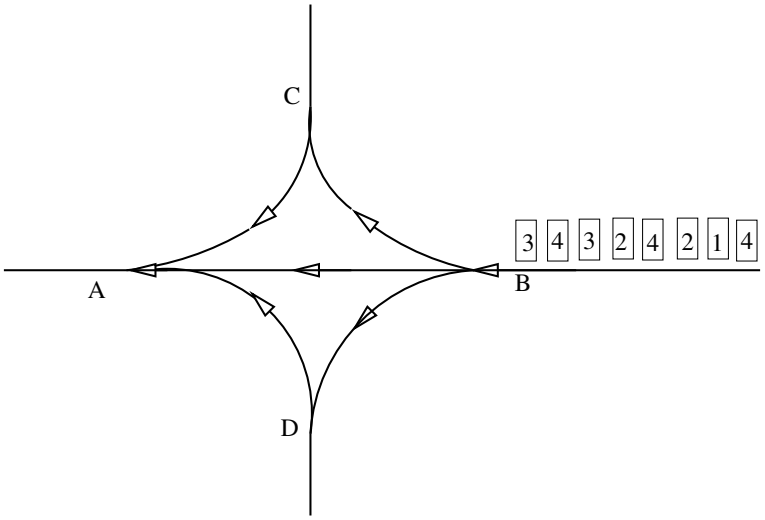
Vasúti kocsik rendezése

Duplaverem város vasútállomásán sok gondot okoz a vasúti kocsik rendezése. Az állomásról továbbítandó szerelvényeket úgy kell kialakítani, hogy amikor az megérkezik a célállomásra, a szerelvény végéről mindig lekapcsolható legyen az oda továbbított kocsisor. Minden továbbítandó szerelvény négy állomást érint, ezért a rendezés előtt minden kocsit megjelölnek az 1, 2, 3 vagy 4 számmal. A szerelvény kocsijait át kell rendezni úgy, hogy a szerelvény elején legyenek az 1-essel, aztán a 2-essel, majd a 3-assal, végül a 4-essel megjelöltek. Kezdetben a kocsik az ábrán látható `B` pályaszakaszon vannak. A vasúti váltók működése csak a következő műveleteket teszi lehetővé. Az átrendezendő kocsisorból balról az első kocsit át lehet mozgatni vagy az `A` szakaszba a már ott lévő kocsik mögé, vagy a `C` vagy `D` szakaszba a már ott lévő kocsik elé. Továbbá, a `C` vagy `D` szakaszon lévő első kocsit át lehet mozgatni az `A` szakaszon kialakítandó rendezett kocsisor végére.



1. ábra.

Feladat

Ijunk olyan programot, amely a megadott bemenetek mindegyikére meghatározza, hogy az adott kocsisor rendezhető-e!!

Bemenet

A standard bemenet első sora egy egész számot tartalmaz, a rendezésre váró kocsisorok n számát ($1 \leq n \leq 1000$). A további n sor mindegyike egy rendezésre váró kocsisor leírását tartalmazza, a sort a 0 szám zárja (ami természetesen nem része a bemenetnek). A sorban az utolsó 0 kivételével minden szám értéke 1,2,3 vagy 4 lehet. A sor legfeljebb 1 000 000 számot tartalmaz egy-egy szóközzel elválasztva.

Kimenet

A standard kimenetre pontosan n sort kell kiírni. Az i-edik sorba az `Igen` szöveget kell kiírni, ha a bemenet $i+1$-edik sorában megadott kocsisor rendezhető, egyébként pedig a `Nem` szöveget.

Példa

Bemenet	Kimenet

`2`
`3 4 3 2 4 2 1 4 0`
`2 3 1 4 2 1 0`
	`Igen`
	`Nem`

Korlátok

Időlimit: 0.1 mp.
Memórilimit: 32 MiB
A tesztek 40%-ában $n < 100$ és a kocsisor legfeljebb 1 000 elemű

Megoldás

A megoldás megadható bizonyos minták előfordulása alapján. Azt mondjuk, hogy az S sorozatban előfordul az

`a-b-c` minta, ha S felbontható úgy, hogy $S = \alpha, a, \beta, b, \gamma, c, \delta$.
Ha a K bemeneti számsorozatban az 1,2,3 és 4 számok közül csak három fordul elő, akkor nyilván rendezhető. A továbbiakban feltesszük, hogy mind a négy szám előfordul. Bontsuk fel a K sorozatot az 1 szám utolsó előfordulása szerint $K = E, 1, U$ alakban, ahol tehát az 1 szám nem szerepel az U sorozatban.
Ha az$E$ sorozat tartalmaz `2-3-4` mintát, akkor biztosan nem rendezhető, mivel az 1-es előtti nem 1-es számok mindegyikét be kell rakni valamelyik verembe, hogy az 1-est ki tudjuk vinni a helyére. Azonban verembe nem rakható kisebbre nagyobb. Másképpen fogalmazva, az E sorozatot fel kell tudni bontani két monotos csökkenő részsorozatra.

Tegyük fel, hogy az E sorozatban előfordül a 2,3 és 4 számok mindegyike, egyébként nyilván rendezhető a bemenet. Tekintsük E-nek az $E = \alpha, 3, \beta, 4, \gamma$ felbontását, ahol γ nem tartalmaz 4-est, β pedig nem tartalmaz 3-ast. Ekkor α nem tartalmazhat 2-est, mert akkor lenne `2-3-4` minta E-ben. Megmutatjuk, hogy ha α nem tartalmaz 2-est, és vagy β nem tartalmaz 2-est, vagy *gamma* nem tartalmaz 3-ast, akkor bármilyen is az utolsó 1-es utáni U sorozat, a teljes bemenet rendezhető. Az E részsorozatból minden 1-est kiviszünk a kimeneti sorozat végére, a nem 1-eseket pedig betesszük valamelyik verembe az alábbiak szerint.

- β nem tartalmaz 2-est: Ekkor minden 4-est és 2-est az alsó verembe rakunk és minden 3-ast a felső verembe. Ezt követően az alsó veremből minden 2-est kiviszünk a kimeneti sorozat végére.
- γ nem tartalmaz 3-ast: Ekkor minden 4-es az alsó verembe, minden 3-as és 2-es pedig a felső verembe megy. Majd a felső veremből minden 2-est kiviszünk a kimeneti sorozat végére.

Ekkor az egyik verem csak 3-ast, a másik csak 4-est tartalmaz, tehát az U sorozat 2-eseit kiviszünk a kimeneti sorozat végére (U-ban már nincs 1-es), minden 4-est az alsó verembe, minden 3-st a felső verembe teszünk. Majd kirakjuk a veremből 3-asokat, aztán a 4-eseket.
Ha β tartalmaz 2-est és γ 3-st, akkor csak olyan felbontását tudjuk adni az E sorozatnak, hogy mindkét veremben lesz 3-as, ezért az U sorozat nem tartalmazhat 4-est.

Megvalósítás C++ nyelven

```
1  #include <iostream>
2  #define maxN 100001
3  using namespace std;
4
5  int main(){
6     int m,n,x,u1,u3,u4;
7     int K[maxN];
8     cin>>m;
9     for(int t=0;t<m;t++){
10       n=0; u1=0;
11       for(;;){
12          cin>>x;
13          if (x==0) break;
14          K[++n]=x;
15          if(x==1) u1=n;
16       }
17       if (u1==0){
18          cout<<"Igen"<<endl;
19          continue;
```

```
20          }
21          u4=0; u3=0;
22          bool van2=false,van3=false,van4=false;
23          bool van43=false, van32=false, van23=false;
24          for(int i=u1-1;i>0; i--){
25             switch (K[i]){
26                case 1:  break;
27                case 2:  van2=true;
28                         if(u3>0) van23=true;
29                         if(u3==0 && u4>0) van32=true;
30                         break;
31                case 3:  van3=true;
32                         if(u4==0) van43=true;
33                         if (u4>0 && u3==0) u3=i;
34                         break;
35                case 4:  van4=true;
36                         if(u4==0) u4=i;
37                         break;
38             }
39          }
40          if(!van2 || !van3 || !van4){
41             cout<<"Igen"<<endl;
42             continue;
43          }
44          if(van23){//van 2-3-4 minta
45             cout<<"Nem"<<endl;
46             continue;
47          }
48          if(van32 && van43){//van 3-2-4-3 minta
49             van4=false;
50             for(int l=u1+1;l<=n;l++)
51                if(K[l]==4) van4=true;
52             if (van4)
53                cout<<"Nem"<<endl;
54             else
55                cout<<"Igen"<<endl;
56          }else{
57             cout<<"Igen"<<endl;
58          }
59       }//for m
60       return 0;
61    }
```