ÚJ SZÉCHENYI TERV

Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Elemi programozási tételek 1

TÁMOP-4.2.3.-12/1/KONV

MAGYARORSZÁG MEGÚJUL

A projektek az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósulnak meg.

Feladataink egy jelentős csoportjában egyetlen bemenő sorozat alapján kell meghatároznunk egy értéket eredményként.

1. Sorozatszámítás

Kezdjük a legelső témakört néhány feladattal!

F1. Egy osztály N db tanuló osztályzatának ismeretében adjuk meg az osztály átlagát!

F2. Egy M elemű betűsorozat betűit fűzzük össze egyetlen szöveg típusú változóba!

F3. Készítsünk algoritmust, amely egy autóversenyző körönkénti ideje alapján meghatározza a versenyző egy kör megtételéhez szükséges átlagidejét!

F4. A Balaton mentén K db madarász végzett megfigyeléseket. Mindegyik megadta, hogy milyen madarakat látott. Készítsünk algoritmust, amely megadja a megfigyelések alapján a Balatonon előforduló madárfajokat!

F5. Adjuk meg az első N természetes szám szorzatát (N faktoriális)!

Vizsgáljuk meg, mi a közös a fenti öt feladatban! Mindegyikben adatok valamilyen sorozatával kell foglalkoznunk, e sorozathoz kell hozzárendelnünk egyetlen értéket. Ezt az értéket egy, az egész sorozaton értelmezett függvény adja (N szám összege, M betű egymásutánírása, K halmaz uniója, N szám szorzata). Ezt a függvényt azonban felbonthatjuk értékpárokon kiszámított függvények sorozatára (2 „valami” összegére, egymásután írására, uniójára, szorzatára).

Az algoritmus:

```
Változók:
 N : Egész            [a feldolgozandó sorozat elemei száma]
 X : Tömb(1..N:Elemtípus)  [a feldolgozandó sorozat elemei]
 F0: Elemtípus                        [a művelet nulleleme]
 S : Elemtípus                                [az eredmény]
```

Így tehát minden olyan művelet szerepelhet e feladattípusban, amelyre a matematika valamilyen „egységes” jelölést használ: összeg, szorzat, unió, metszet, logikai műveletek, konkatenáció, ... Mindegyik művelet visszavezethető egy bináris műveletre, s megadható mindegyikhez egy semleges elem (nullelem) is, amelyre egy tetszőleges elemmel és vele elvégzett 2 operandusú művelet az adott elemet adja.

Variációk: $F = \Sigma, \Pi, \cup, \cap, \wedge, \vee$, & (konkatenáció).

F_0=0, 1, {}, alaphalmaz, Igaz, Hamis, "".

A megoldás a nullelemre, valamint a 2 operandusú műveletre épül, a matematikában is szokásos módszer, az indukció alapján:

- 0 elemre tudjuk az eredményt: F_0,

- ha i-1 elemre tudjuk az eredményt (F^{i-1}), akkor i elemre (F^i) a következőképpen kell kiszámítani: $F^i=f(F^{i-1},X_i)$.

  Az F-re vonatkozó alábbi rekurzív definícióból:

$$F^i:=F(X_1,...,X_i):=\begin{cases} F_0 & , \text{ ha } i = 0 \\ f(F^{i-1}, X_i)), & \text{különben} \end{cases}$$

  Végezetül az algoritmikus nyelvünkben a feladatot megoldó algoritmus:

```
Sorozatszámítás(N,X,S):
 S:=F0
 Ciklus I=1-től N-ig
   S:=f(S,X(I))
 Ciklus vége
Eljárás vége.
```

  Alkalmazzuk az általános megoldást a fejezet elején kitűzött egyes feladatokra!

  F1. N szám összege

```
elemek száma: N
osztályzatok: X (N db elem)
F, f, F0    : S, +, 0
```

```
Összegzés(N,X,S):
  S:=0
  Ciklus I=1-től N-ig
    S:=S+X(I)
  Ciklus vége
Eljárás vége.
```

  F2. M betű egymásutánírása

```
elemek száma: M
betűk       : X (M db elem)
F, f, F0    : &, +, "",
```

```
Egymásután(M,X,SZ):
  SZ:="" (üres szöveg)
  Ciklus I=1-től M-ig
    SZ:=SZ+X(I)
  Ciklus vége
Eljárás vége.
```

  F4. K halmaz uniója

```
elemek száma : K
megfigyelések: X (N db elem)
F, f, F0     : ∪, ∪, {}
```

```
Unió(K,X,H):
  H:={} (üres halmaz)
  Ciklus I=1-től K-ig
    H:=H∪X(I)
  Ciklus vége
Eljárás vége.
```

2. Megszámolás

Az előző feladatokban előfordulhatott, hogy több elem is rendelkezik a vizsgált tulajdonsággal. Ekkor érdekes lehet annak megvizsgálása, hogy hány ilyen elem van.

F6. Családok létszámának, illetve jövedelmének alapján állapítsuk meg, hogy hány család él a létminimum alatt!

F7. Egy futóverseny végeredménye határozzuk meg, hogy a versenyzők hány százaléka teljesítette az olimpiai induláshoz szükséges szintet!

F8. Adjuk meg egy szöveg magánhangzóinak számát!

A közös jellemző itt tehát a számlálás. Vegyük észre, hogy a feladatot sorozatszámításként is felfoghatjuk, amelyben 1-eseket kell összeadnunk, de bizonyos feltételtől függően.

Az algoritmus:

```
Függvény:
 T: Elemtípus → Logikai
Változók:
 N : Egész           [a feldolgozandó sorozat elemei száma]
 X : Tömb(1..N:Elemtípus) [a feldolgozandó sorozat elemei]
 DB: Egész         [az eredmény - a megfelelő elemek száma]
```

A feladat sorozatszámítás (sőt összegzés), tehát egy ciklust kell alkalmazni a megoldáshoz. A ciklus belsejében egy unió típusú (χ függvény értéke) adatot kell feldolgozni, ezt egy elágazással tehetjük meg.

```
Megszámolás(N,X,DB):
 DB:=0
 Ciklus I=1-től N-ig
   Ha T(X(I)) akkor DB:=DB+1
 Ciklus vége
Eljárás vége.
```

Nézzük meg két feladat megoldását!

F16. Létminimum alattiak száma.

Minden családi létszámhoz megadható az a jövedelem, amely a létminimumhoz kell, a megoldásban ezt használjuk fel.

```
elemek száma: N
létszámok: L (N db elem)
jövedelmek: J (N db elem)
létminimumok: MIN sorozat
T(L-j): j-MIN(L)≥0
```

```
Megszámolás(N,J,L,DB):
  DB:=0
  Ciklus I=1-től N-ig
    Ha J(I)≤MIN(L(I))
          akkor DB:=DB+1
  Ciklus vége
Eljárás vége.
```

F6. Olimpiai indulási szintet hány százalék teljesítette?

A feladat megoldása egy megszámolás, majd az eredményből és a résztvevők számából százalékot számolunk.

```
elemek száma: N
idők: ID (N db elem)
T(id): id≤SZINT
```

```
Megszámolás(N,ID,SZAZ):
 DB:=0
 Ciklus I=1-től N-ig
   Ha ID(I)≤SZINT akkor DB:=DB+1
 Ciklus vége
 SZAZ:=Kerekít(100*DB/N)
Eljárás vége.
```

3. Maximumkiválasztás

A sorozatszámítás egy újabb speciális esetével ismerkedünk meg a következő feladattípusban, először néhány feladaton keresztül.

F9. Egy kórházban megmérték minden beteg lázát, adjuk meg, hogy ki a leglázasabb!

F10. Egy család havi bevételei és kiadásai alapján adjuk meg, hogy melyik hónapban tudtak a legtöbbet megtakarítani!

F11. Egy osztály tanulói nevei alapján adjuk meg a névsorban legelső tanulót!

Közös jellemzőjük e feladatoknak, hogy minden esetben egy sorozat elemei közül kell kiválasztani a legnagyobbat (illetve a legkisebbet). Itt is meggondolandó - mint a *kiválasztás*nál -, hogy elem értéket vagy pedig elem sorszámot várunk eredményként. Az ottani gondolatmenethez hasonlóan most is a sorszám meghatározását választjuk.

Az algoritmus:

```
Változók:
 N  : Egész          [a feldolgozandó sorozat elemei száma]
 X  : Tömb(1..N:Elemtípus)[a feldolgozandó sorozat elemei]
 MAX: Egész              [a maximális értékű elem sorszáma]
```

A sorozatszámításnál használjuk **F** függvényként a $MAX(A_1,A_2,...A_N)$ függvényt! Ehhez az **f** függvényt a következőképpen használjuk:

$$f(x,y):=max(x,y).$$

Legyen ehhez $\mathbf{F}_0:=A_1$! Alakítsuk át úgy a sorozatszámítás megoldását, hogy ne értéket, hanem sorszámot kapjunk eredményül! (Annyi észrevenni valónk van csak az algoritmizáláskor, hogy most a sorozat a 2. elemmel kezdődik.)

```
Maximumkiválasztás(N,X,MAX):
  MAX:=1
  Ciklus I=2-től N-ig
    Ha X(MAX)<X(I) akkor MAX:=I
  Ciklus vége
Eljárás vége.
```

Sok esetben túlságosan sok időbe kerül a sorozat egy elemének meghatározása. Ekkor olyan megoldást készíthetünk, amely a maximális értéket határozza meg, vagy pedig a sorszámot is és az értéket is.

```
Maximumkiválasztás(N,X,MAX,MAXERT):
  MAX:=1; MAXERT:=X(1)
  Ciklus I=2-től N-ig
    Ha MAXERT<X(I) akkor MAX:=I; MAXERT:=X(I)
  Ciklus vége
Eljárás vége.
```

Ha a feladat minimumkiválasztás, akkor pedig nincs más teendőnk, mint az elágazás feltételében szereplő < reláció átírása >-ra. Erre példa az egyik kitűzött feladat megoldása.

F11. A névsorban legelső tanuló.

```
elemek száma: N
tanulók neve: X (N db elem)
```

```
Minimum(N,X,MIN,MEV):
  MIN:=1; NEV:=X(1)
  Ciklus I=2-től N-ig
    Ha NEV>X(I)
      akkor MIN:=I:NEV:=X(I)
  Ciklus vége
Eljárás vége.
```

Feladatatok programozási tételekre a Nemes Tihamér OITV-ről és az Informatika OKTV-ről

1. feladat

Egy iskola tanárairól tudjuk, hogy mikor milyen órát tartanak. A tanárokat, a tantárgyakat, a hét napjait, a napokon belüli órákat sorszámukkal azonosítjuk. Készíts programot (`iskola.pas`, …), amely megadja:

A. minden napra az aznap órát tartó tanárok számát;
B. azt a tantárgyat, amit a legtöbb tanár tanít;
C. azt a tanárt, akinek a legtöbb lyukasórája van (lyukasóra: aznap előtte is van órája valamikor és utána is van órája valamikor);
D. az adott T tanárt egész héten helyettesíteni tudó tanárt.

Az `iskola.be` szöveges állomány első sorában a tanárok száma ($1 \le N \le 100$), a tantárgyak száma ($1 \le M \le 100$) és egy tanár sorszáma van ($1 \le T \le N$), egy-egy szóközzel elválasztva. A következő sorok mindegyikében 4 egész szám van, egy-egy szóközzel elválasztva: tanár sorszám, tanított tantárgy sorszáma, nap (1 és 5 közötti egész szám), óra (0 és 8 közötti egész szám). Például 3 7 2 0 azt jelenti, hogy a harmadik tanár a hetedik tantárgyat a hét második napján a nulladik órában tanítja.

Az `iskola.ki` szöveges állományba négy sort kell írni! Az első sorba az A, a másodikba a B, a harmadikba a C, a negyedikbe pedig a D részfeladat eredményét. Ha több megoldás van, akkor az elsőt kell kiírni. Ha nincs megoldás (C és D részfeladatban), akkor -1-et kell kiírni! Az első sorban 5 szám szerepeljen, egy-egy szóközzel elválasztva!

Példa:

```
iskola.be   iskola.ki

3 4 1       2 3 1 0 0
1 1 1 6     2
1 1 2 2     1
1 2 1 3     3
2 1 2 2
2 2 3 1
3 4 1 2
3 2 1 4
3 3 2 1
```

A példában szereplő 3 tanár órarendje:

1. tanár	1. nap	2. nap	3. nap	2. tanár	1. nap	2. nap	3. nap	3. tanár	1. nap	2. nap	3. nap
0. óra				0. óra			T2	0. óra			
1. óra				1. óra				1. óra		T3	
2. óra		T1		2. óra		T1		2. óra	T4		
3. óra	T2			3. óra				3. óra			
4. óra				4. óra				4. óra	T2		
5. óra				5. óra				5. óra			
6. óra	T1			6. óra				6. óra			

A. NEM TÉTELES FELADAT!

B. NEM TÉTELES FELADAT!

C. Minden tanárra minden napra számoljuk meg, hogy az adott tanár aznap hány órát tart (DB), válasszuk ki az aznapi legelső (E) és legutolsó (U) óráját. A lyukasórák száma ekkor U-E+1-DB, ha DB>0.

D. NEM TÉTELES FELADAT!

```
Beolvasás:
  db():=(0,…,0); e():=(9,…,9); u():=(-1,…,-1)
  Ciklus amíg nem vége(f)
    Olvas(f,i,j,k,o)
    db(i,k):=db(i,k)+1
    Ha o<e(i,k) akkor e(i,k):=o
    Ha o>u(i,k) akkor u(i,k):=o
  Ciklus vége
Eljárás vége.

C:
  max:=0; maxért:=-1
  Ciklus i=1-től n-ig
    lyukas(i):=0
```

```
    Ciklus j=1-től 5-ig
      Ha db(i,j)>1
         akkor lyukas(i):=lyukas(i)+u(i,j)-e(i,j)+1-db(i,j)
    Ciklus vége
    Ha lyukas(i)>maxért akkor maxért:=lyukas(i); max:=i
  Ciklus vége
Eljárás vége.
```

2. feladat

Egy vizsgabizottságban egy nap feljegyezték, hogy az egyes vizsgázók mikor vizsgáztak (egyszerre egy vizsgázó lehet), mindenkiről 4 adatot tudunk: `kezdőóra, kezdőperc, végóra, végperc.` A vizsgázók adatait idő szerinti sorrendben kapjuk.

Készíts programot (`vizsga.pas`,…), amely beolvassa a vizsgázók számát ($1 \le N \le 100$), az egyes vizsgázók vizsgájának kezdetét (v[i,1]) és végét (v[i,2]), majd megadja

A. annak a 60 perces időszaknak a kezdetét, amikor a legtöbb vizsgázó végzett a vizsgájával (közülük az első pontosan ebben a percben végezzen);

B. a leghosszabb vizsgaszünet hosszát – óra, perc (amikor 2 vizsgázó között senki sem volt a vizsgabizottságnál);

C. a leghosszabb időtartamot, amikor a vizsgabizottság nem tarthatott szünetet!

Példa

```
Bemenet:
N=6
8 20 8 30
8 50 9 0
9 50 10 10
10 10 10 30
10 30 10 55
11 55 12 10
```

```
Kimenet:
10 10        (10 óra 10 perctől)
1 0          (1 óra, 0 perc)
1 5          (1 óra, 5 perc)
```

Az első fontos gondolat: beolvasáskor azonnal mindent számoljunk át percekbe, azzal könnyebb lesz dolgozni! Eredmény kiírásnál pedig számoljuk vissza órákra és percekre!

A. NEM ILYEN TÉTEL;

B. vizsgakezdetek és az előző vizsga végek különbségének maximuma;

C. az utolsó szünet végétől az adott vizsga végéig terjedő időtartamok maximuma!

```
B:
  idő:=0
  Ciklus i=1-től n-1-ig
    Ha v(i+1,1)-v(i,2)>idő akkor idő:=v(i+1,1)-v(i,2)
  Ciklus vége
Eljárás vége.
```

```
C:
  idő:=-1; k:=1
  Ciklus i:=2-től n-ig
    Ha v(i,1)≠v(i-1,2) akkor
      Ha v(i-1,2)-v(k,1)>idő akkor idő:=v(i-1,2)-v(k,1)
      k:=i
    Elágazás vége
  Ciklus vége
  Ha v(n,2)-v(k,1)>j akkor idő:=v(n,2)-v(k,1)
Eljárás vége.
```

3. feladat

Ismerjük N emberről, hogy ki szülője kinek ($1 \le N \le 100$). Az embereket az 1 és N közötti sorszámukkal azonosítjuk.

Készíts programot (`ember.pas`, …), amely megadja, hogy

A. kinek van a legtöbb gyereke;

B. kinek van a legtöbb unokája;

C. hány embernek nincs unokája!

A bemenet első sorában az N értéke szerepel, a következő N-1 sor pedig egy-egy szülő és gyerek sorszámát tartalmazza.

Példa:

Bemenet:		Kimenet:	Magyarázat:
N=12			
szülő:	gyerek:	A: 4	5,6,7,10 sorszámú
1	2	B: 1	4,8,9,11,12 sorszámú
1	3	C: 10	csak 1-nek és 2-nek van
2	4		
2	8		
2	9		
4	5		
4	6		
4	7		
4	10		
3	11		
3	12		

A. aki szülőként szerepel, annak a gyerekszámát növeljük eggyel – ha mindenkinek ismert a gyerekszáma, akkor azokra kell egy maximum;
B. aki szülőként szerepel és a gyereke valahányszor szülő, akkor az unokák számát növeljük annyival – ha mindenkinek ismert az unokaszáma, akkor azokra kell egy maximum;
C. a B részben kiszámolt unokaszámok között hány 0 van!

```
A:
  gy():=(0,…,0)
  Ciklus i=1-től m-ig
    gy(szülő(i)):=gy(szülő(i))+1
  Ciklus vége
  mgy:=1
  Ciklus i=2-től n-ig
    Ha gy(i)>gy(mgy) akkor mgy:=i
  Ciklus vége
Eljárás vége.

B:
  u():=(0,…,0)
  Ciklus i=1-től m-ig
    Ciklus j=1-től m-ig
      Ha gyerek(i)=szülő(j) akkor u(szülő(i)):=u(szülő(i))+1
    Ciklus vége
  Ciklus vége
  mu:=1
  Ciklus i=2-től n-ig
    Ha u(i)>u(mu) akkor mu:=i
  Ciklus vége
Eljárás vége.

C:
  db:=0
  Ciklus i=1-től n-ig
    Ha u(i)=0 akkor db:=db+1
  Ciklus vége
Eljárás vége.
```

4. feladat

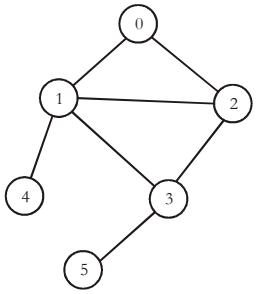
Egy állatkertben ismerjük a bejárható útvonalakat. A bejárat a 0 sorszámú pont. Az egyes állatokat az 1 és N közötti sorszámukkal azonosítjuk ($1 \le N \le 100$), az utakat pedig két olyan állat sorszámával (A,B), amelyek ketrece között vezetnek.

Készíts programot (`allat.pas`, `allat.c`, …), amely az állatkerti utak ismeretében megadja, hogy hány olyan állat van, amelyik zsákutca végén található, valamint azt, hogy melyik állathoz vezet a legtöbb út (ha több is van, bármelyik megadható)!

Példa:

```
Bemenet:              Kimenet:

Állatok száma: 5      Állatok száma zsákutca végén: 2
Utak száma: 7         Legtöbb út: 1
1. út: 0 1
2. út: 1 4
3. út: 3 1
4. út: 3 5
5. út: 2 0
6. út: 2 3
7. út: 1 2
```



Első lépésként minden állatra számoljuk meg, hogy oda hány út vezet (D)! Vigyázat, a 0. „állat” nem állat, hanem a bejárat!

A. ahova 1 út vezet, azok a zsákutcák, ezek számát kell megadni;

B. ahova a legtöbb út vezet, azt kell megadni!

```
A–B:
  d():=(0,…,0)
  Ciklus i=1-től m-ig
    Ha a(i)>0 akkor d(a(i)):=d(a(i))+1
    Ha b(i)>0 akkor d(b(i)):=d(b(i))+1
  Ciklus vége
  db:=0; max:=1
  Ciklus i=1-től n-ig
    Ha d(i)=1 akkor db:=db+1
    Ha d(i)>d(max) akkor max:=i
  Ciklus vége
Eljárás vége.
```

5. feladat

Ismerjük N településre az M napos időjárás előrejelzést, ezek alapján keressük a legmelegebb települést.

Készíts programot (`elore.pas`, …), amely megadja négyféle értelmezés szerint a legmelegebb települést:

A. a legmelegebb település az, amelyre az előrejelzések maximuma a legnagyobb;

B. a legmelegebb település az, amelyre az előrejelzések átlaga a legnagyobb;

C. a legmelegebb település az, amelyben a leghosszabb időszakon belül várható folyamatosan K fok feletti hőmérséklet;

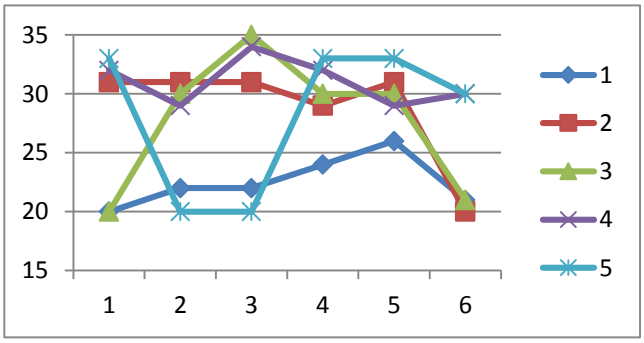
D. a legmelegebb település az, amelyre a legtöbb napon fordul elő, hogy a várt hőmérséklet nagyobb minden más arra a napra előre jelzett hőmérsékletnél.

Az `elore.be` szöveges állomány első sorában a települések száma (1≤N≤1000), a napok száma (1≤M≤1000) és a hőmérséklet korlát van (20≤K≤50), egy-egy szóközzel elválasztva. A következő N sor mindegyikében M egész szám van, egy-egy szóközzel elválasztva: az i-edik település j-edik napra várt hőmérséklete.

Az `elore.ki` szöveges állomány négy sorába egy-egy település sorszámát kell írni! Az első sorba az A, a másodikba a B, a harmadikba a C, a negyedikbe pedig a D szempont szerinti legmelegebb települést. Ha több megoldás van, bármelyik megadható. Ha nincs megoldás (C és D részfeladatban), akkor -1-et kell kiírni!

Példa:

```
elore.be                elore.ki
5 6 30                      3
20 22 22 24 26 21           4
31 31 31 29 31 20           2
20 30 35 30 30 21           5
32 29 34 32 29 30
33 20 20 33 33 30
```



A. egy mátrix azon sorát kel megtalálni, ahol a legnagyobb érték szerepel;

B. egy mátrix azon sorát kel megtalálni, ahol a sorösszeg a lehető legnagyobb;

C. egy mátrix azon sorát kel megtalálni, ahol a lehető leghosszabb az az intervallum, amikor folyamatosan K fok feletti hőmérséklet;

D. NEM ELEMI TÉTEL

```
A:
  max:=-maxint
  Ciklus i=1-től n-ig
    Ciklus j=1-től m-ig
      Ha h(i,j)>max akkor a:=i; max:=h(i,j)
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

```
B:
  max:=-maxint
  Ciklus i=1-től n-ig
    s:=0
    Ciklus j=1-től m-ig
      s:=s+h(i,j)
    Ciklus vége
    Ha s>max akkor b:=i; max:=s
  Ciklus vége
Eljárás vége.
```

```
C:
  max:=-1; c:=-1
  Ciklus i=1-től n-ig
    h(i,0):=-maxint; h(i,m+1):=-maxint; ck:=0; cv:=-1
    Ciklus j=1-től m-ig
      Ha h(i,j)>k és h(i,j-1)≤k akkor c1:=j
      Ha h(i,j)>k és h(i,j+1)≤k
         akkor Ha j-c1>cv-ck akkor ck:=c1; cv:=j
    Ciklus vége
    Ha cv-ck>max akkor c:=i; max:=cv-ck
  Ciklus vége
Eljárás vége.
```

6. feladat

Egy kieséses versenyben ismerjük a csapatok mérkőzéseit: ki kit győzött le.

Írj programot (`kieses.pas`, …), amely megadja:

A. a még versenyben levőket;

B. azokat a csapatokat, amelyek legalább egyszer győztek, de már kiestek;

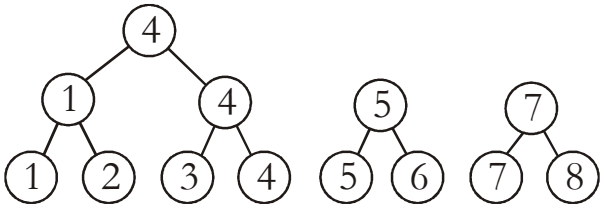
C. a legtöbb csapatot közvetlenül vagy közvetve legyőző csapatot!

A `kieses.be` szöveges állomány első sorában a csapatok száma ($2 \le N \le 1000$) és a mérkőzések száma van ($1 \le M < N$), egy szóközzel elválasztva. A következő M sor mindegyikében két csapat A és B sorszáma van ($1 \le A \ne B \le N$), ami azt jelenti, hogy az A-adik csapat legyőzte a B-edik csapatot.

A `kieses.ki` szöveges állomány első sorába a még versenyben levő csapatok darabszámát, majd a sorszámát kell írni (egy-egy szóközzel elválasztva, növekvő sorrendben), a második sorba azok darabszámát, majd a sorszámát, amelyek úgy estek ki, hogy legalább egyszer győztek (egy-egy szóközzel elválasztva, növekvő sorrendben), a harmadik sorba pedig azt a csapatot, amely a legtöbb más csapatot győzte le közvetve vagy közvetlenül! Ha több megoldás van, bármelyik kiírható.

Példa:

```
kieses.be          kieses.ki
8 5                3 4 5 7
1 2                1 1
4 3                4
4 1
7 8
5 6
```



Először számoljuk meg mindenkire, hogy hányszor győzött (GY), illetve hányszor veszített (V)! Utána a következőket kapjuk:

A. azok vannak még versenyben, akik nem veszítettek;

B. azokat a csapatokat, amelyek legalább egyszer győztek, de már kiestek, azaz veszítettek is;

C. NEM ELEMI TÉTEL

```
Előkészítés:
  Ciklus i=1-től m-ig
    gy(a(i)):=gy(a(i))+1; v(b(i)):=v(b(i))+1
  Ciklus vége
Eljárás vége.
```

```
Afeladat:
  a:=0
  Ciklus i=1-től n-ig
    Ha v(i)=0 akkor a:=a+1; ta(a):=i
  Ciklus vége
Eljárás vége.
```

```
Bfeladat:
  b:=0
  Ciklus i=1-től n-ig
    Ha gy(i)*v(i)>0 akkor b:=b+1; tb(b):=i
  Ciklus vége
Eljárás vége.
```