Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Halmaz típus

TÁMOP-4.2.3.-12/1/KONV

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

Értékhalmaz: az alaphalmaz (amely az *Elemtípus* által van meghatározva) iteráltja („mely elemek lehetnek benne a halmazban"). Az *Elemtípus* általában valamely véges diszkrét típus lehet, legtöbbször még az elemszámát is korlátozzák (≤256). Ha nyelvi elemként nem létezik, akkor a megvalósításunkban lehet nagyobb elemszámú is.

Műveletek: *metszet*, *unió*, *különbség*, *komplemens*, *eleme* (egy elem benne van-e a halmazban), *része* (egyik halmaz részhalmaza-e a másiknak), *Halmazba* (elem hozzávétele egy halmazhoz), *Halmazból* (elem elhagyása egy halmazból), *Beolvasás* (halmaz beolvasása), *Kiírás* (halmaz kiírása), *Üres* (üres halmaz létrehozás eljárás), vagy *Üres'Halmaztípus* előre definiált konstans, *Üres*? (logikai értékű függvény).

Relációk: = , < , ≤ , ≥ , > , ≠ (parciális rendezés: a tartalmazás alapján).

Például:

```
Típus Nap    =0 .. 23
      Foglalt=Halmaz(Nap)

Változó ma,holnap: Foglalt

Konstans munkanap: Foglalt(7..12,14..20)

Üres(ma)  [ma "szabad"]
Ha Üres?(holnap) akkor ma:=munkanap
```

Halmazok ábrázolása többféleképpen is megoldható. Közülük tekintsük át a legfontosabbakat:

Elemek felsorolása

Halmaz(Elemtípus)=**Rekord**(db: Egész,elem: **Tömb**(1..MaxDb:Elemtípus))

Egy felsorolásként adjuk meg a halmazt, annyi elemű tömbben, ahány elemű éppen a halmaz (pontosabban az első Db darab elemében).

Elemek felsorolása esetén a halmazműveleteket az alábbi módon valósíthatjuk meg:

```
Eljárás Beolvasás(Változó a: Halmaz(Elemtípus)):
  Be: a.db [a halmaz elemszáma]
  Ciklus i=1-től a.db-ig
    Be: a.elem[i]
  Ciklus vége
Eljárás vége.
```

Műveletigény számítása: a ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a **halmaz elemszámával arányos**.

```
Eljárás Kiírás(Konstans a: Halmaz(Elemtípus)):
  Ki: a.db [a halmaz elemszáma]
  Ciklus i=1-től a.db-ig
    Ki: a.elem[i]
  Ciklus vége
Eljárás vége.
```

Műveletigény számítása: a ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a **halmaz elemszámával arányos**.

```
Eljárás Üres(Változó a: Halmaz(Elemtípus)):
  a.db:=0
Eljárás vége.
```

Műveletigény számítása: nem függ a **halmaz elemszámától**.

```
Függvény Üres?(Konstans a: Halmaz(Elemtípus)):
  Üres?:=(a.db=0)
Függvény vége.
```

Műveletigény számítása: nem függ a **halmaz elemszámától**.

```
Eljárás Halmazba(Változó a: Halmaz(Elemtípus),
                           Konstans e: elemtípus):
  a.db:=a.db+1
  a.elem[a.db]:=e
Eljárás vége.
```

Műveletigény számítása: nem függ a **halmaz elemszámától**.

```
Eljárás Halmazból(Változó a: Halmaz(Elemtípus),
                           Konstans e: Elemtípus):
  i:=1
  Ciklus amíg i≤a.db és a.elem[i]≠e
    i:=i+1
  Ciklus vége
  Ha i≤a.db akkor a.elem[i]:=a.elem[a.db]; a.db:=a.db-1
Eljárás vége.
```

Műveletigény számítása: a ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a **halmaz elemszámával arányos**.

```
Függvény eleme(Konstans e:Elemtípus, a: Halmaz(Elemtípus)):
                                       Logikai
  i:=1
  Ciklus amíg i≤a.db és e≠a.elem[i]
    i:=i+1
  Ciklus vége
  eleme:=i≤a.db
Függvény vége.
```

Műveletigény számítása: a külső ciklus az A halmaz elemszámaszor fut le, azaz a futási idő a **halmaz elemszámával arányos**.

```
Függvény része(Konstans a,b: Halmaz(Elemtípus)):
                                          Logikai
  i:=1
  Ciklus amíg i≤a.db és eleme(a.elem[i],b)
    i:=i+1
  Ciklus vége
  része:=i≤a.db
Függvény vége.
```

Műveletigény számítása: a ciklus az A halmaz elemszámaszor fut le, az elem függvény pedig a B halmaz elemszámaszor, azaz a futási idő a **két halmaz elemszámának szorzatával arányos**.

```
Operátor unió(Konstans a,b: Halmaz(Elemtípus)):
                                          Halmaz(Elemtípus)
              Másnéven: +
  Változó c: Halmaz(Elemtípus)
  c:=a
  Ciklus i=1-től b.db-ig
    Ha nem eleme(b.elem[i],a) akkor Halmazba(c,b.elem[i])
  Ciklus vége
  unió:=c
Operátor vége.
```

Műveletigény számítása: a külső ciklus a B halmaz elemszámaszor fut le, a belső legroszszabb esetben az A halmaz elemszámaszor, azaz a futási idő a **két halmaz elemszámának szorzatával arányos**.

```
Operátor metszet(Konstans a,b: Halmaz(Elemtípus)):
                                          Halmaz(Elemtípus)
                Másnéven: *
  Változó c: Halmaz(Elemtípus)
  c.db:=0
  Ciklus i=1-től a.db-ig
    Ha eleme(a.elem[i],b) akkor Halmazba(c,a.elem[i])
  Ciklus vége
  metszet:=c
Operátor vége.
```

Műveletigény számítása: a külső ciklus az A halmaz elemszámaszor fut le, a belső legroszszabb esetben a B halmaz elemszámaszor, azaz a futási idő a **két halmaz elemszámának szorzatával arányos**.

```
Operátor különbség(Konstans a,b: Halmaz(Elemtípus)):
                                          Halmaz(Elemtípus)
                    Másnéven: -
  Változó c: Halmaz(Elemtípus)
  c.db:=0
  Ciklus i=1-től a.db-ig
    Ha nem eleme(a.elem[i],b) akkor Halmazba(c,a.elem[i])
  Ciklus vége
  különbség:=c
Operátor vége.
```

Műveletigény számítása: a külső ciklus az A halmaz elemszámaszor fut le, a belső legrosszabb esetben a B halmaz elemszámaszor, azaz a futási idő a **két halmaz elemszámának szorzatával arányos.**

A megoldás alapvető problémája, hogy sehol sem ellenőrizhető, hogy a halmazban valóban csak a benne előfordulható elemek vannak.

Az így ábrázolt halmazok elemtípusára semmilyen megkötést nem kell tennünk, hiszen egy tömbben bármilyen elem elhelyezhető.

Még arra sincs korlátozás, hogy mekkora lehet az alaphalmaz elemszáma, amiből a halmaz elemei származnak. Csak annyi a megkötésünk, hogy a konkrét halmazok elemszámát korlátozzuk. Emiatt itt nem lehet definiálni a halmaz komplemens műveletet sem. Olyan művelet, amely azon lehetséges halmazelemeket adja meg, amelyek az adott halmazban nincsenek benne (például, ha a hét napjai a halmaztípus lehetséges elemei, egy konkrét halmaz pedig a hét munkanapjai, akkor a halmaz komplemense a hét ünnepnapjai).

Bittérkép

Vegyünk fel egy annyi bitből álló sorozatot, amennyi a halmaz lehetséges elemeinek száma. Képezzük le a halmaz elemtípusát az 1..Max'Elemszám típusra! Legyen az i. bit 1-es értékű, ha az i. lehetséges elem benne van a halmazban, s 0, ha nincs benne. Ha egy programozási nyelv rendelkezik Halmaz típussal, akkor általában ezt az ábrázolást választják.

Logikai vektor

A bittérkép általánosításaként a halmazt {igaz,hamis} elemekből álló vektorként is értelmezhetjük, ahol indexként használjuk az elem típusú értéket.

Halmaz(Elemtípus)=**Tömb**(Min'Elemtípus..Max'Elemtípus:Logikai)

Ebben az esetben a halmazműveleteket logikai műveletekre visszavezetve valósítjuk meg:

```
Eljárás Beolvasás(Változó a: Halmaz(Elemtípus)):
  Be: N [a halmaz elemszáma]
  Üres(a)
  Ciklus i=1-től N-ig
    Be: e; a[e]:=igaz
  Ciklus vége
Eljárás vége.
```

Műveletigény számítása: a ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a **halmaz elemszámával arányos**.

```
Eljárás Kiírás(Konstans a: Halmaz(Elemtípus)):
  Ciklus i=Min'Elemtípus-tól Max'Elemtípus-ig
    Ha a[i] akkor Ki: i
  Ciklus vége
Eljárás vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos**.

```
Eljárás Üres(Változó a: Halmaz(Elemtípus)):
  Ciklus i=Min'Elemtípus-tól Max'Elemtípus-ig
    a[i]:=hamis
  Ciklus vége
Eljárás vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos**.

```
Függvény Üres?(Konstans a: Halmaz(Elemtípus)):
  i:=Min'Elemtípus
  Ciklus amíg i≤Max'Elemtípus és nem eleme(i,a))
    i:=i+1
  Ciklus vége
  Üres?:=(i>Max'Elemtípus)
Függvény vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos**.

```
Eljárás Halmazba(Változó a: Halmaz(Elemtípus),
                           Konstans e: elemtípus):
  a[e]:=igaz
Eljárás vége.
```

Műveletigény számítása: nem függ a **halmaz elemszámától**.

```
Eljárás Halmazból(Változó a: Halmaz(Elemtípus),
                           Konstans e: elemtípus):
  a[e]:=hamis
Eljárás vége.
```

Műveletigény számítása: nem függ a **halmaz elemszámától**.

```
Függvény eleme(Konstans e:Elemtípus, a: Halmaz(Elemtípus)):
                                                    Logikai
  eleme:=a[e]
Függvény vége.
```

Műveletigény számítása: nem függ a **halmaz elemszámától**.

```
Függvény része(Konstans a,b: Halmaz(Elemtípus)):
                                                    Logikai
  i:=Min'Elemtípus
  Ciklus amíg i≤Max'Elemtípus és (nem a[i] vagy b[i])
    i:=i+1
  Ciklus vége
  része:=i>Max'Elemtípus
Függvény vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos**.

```
Operátor metszet(Konstans a,b: Halmaz(Elemtípus)):
                                                    Halmaz(Elemtípus)
  Változó c: Halmaz(Elemtípus)
  Ciklus i=Min'Elemtípus-tól Max'Elemtípus-ig
    c[i]:=a[i] és b[i]
  Ciklus vége
  metszet:=c
Operátor vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos**.

```
Operátor unió(Konstans a,b: Halmaz(Elemtípus)):
                                                    Halmaz(Elemtípus)
  Változó c: Halmaz(Elemtípus)
  Ciklus i=Min'Elemtípus-tól Max'Elemtípus-ig
    c[i]:=a[i] vagy b[i]
  Ciklus vége
  unió:=c
Operátor vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos**.

```
Operátor különbség(Konstans a,b: Halmaz(Elemtípus)):
                                                    Halmaz(Elemtípus)
  Változó c: Halmaz(Elemtípus)
  Ciklus i=Min'Elemtípus-tól Max'Elemtípus-ig
    c[i]:=a[i] és nem b[i]
  Ciklus vége
  különbség:=c
Operátor vége.
```

Műveletigény számítása: a ciklus a halmaztípus lehetséges elemeinek számaszor fut le, azaz a futási idő a **halmaztípus elemszámával arányos.**

Itt már van értelme egy halmaz komplemenséről beszélni:

```
Operátor komplemens(Konstans a: Halmaz(Elemtípus)):
                                                Halmaz(Elemtípus)
  Változó c: Halmaz(Elemtípus)
  Ciklus i=Min'Elemtípus-tól Max'Elemtípus-ig
    c[i]:= nem a[i]
  Ciklus vége
  komplemens:=c
Operátor vége.
```

Ennél az ábrázolásnál szigorú megkötés az, hogy a lehetséges elemei indexként használhatók legyenek! Emiatt így nem definiálható síkbeli pontok halmaza, szavak halmaza, …

A kétféle ábrázolás közötti döntésnél (ha az előző bekezdés szerint mindkét ábrázolás használható) érdemes megfontolni az egyes eljárások műveletigényét (ami a futási időt befolyásolja). Ami biztos: az *eleme?* művelet mindenképpen a második ábrázolásnál hatékonyabb. A többi műveletnél az első ábrázolás futási ideje a két halmaz konkrét elemszámának szorzatával arányos, a másik ábrázolásé pedig a lehetséges elemek számával.

Ebből azt a következtetést vonhatjuk le, hogy ha a halmazok átlagos méretének négyzete kisebb, mint a lehetséges elemek száma, akkor az első ábrázolás a hatékonyabb, különben pedig a második. Ez azt jelenti, hogy ha döntési helyzetbe kerülünk, akkor a döntést csak a halmazokról szóló előzetes tudás alapján hozhatjuk meg.

Ami a beolvasás és a kiírás eljárásokból látszik, a felhasználó a halmazokat mindig az első elképzelés szerint, azaz az elemei felsorolásával látja.

A Halmaz típus problémái általában a definíciójából származnak, az elemeknek itt nincs egy természetes sorrendje. Emiatt nehézkes egyes műveletek megvalósítása.

Feladatatok halmazokra 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≤N≤100), a tantárgyak száma (1≤M≤100) és egy tanár sorszáma van (1≤T≤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. Állítsuk elő minden napra az aznap órát tartó tanárok halmazát (NAP) – a megoldás e halmazok elemszáma!

B. Állítsuk elő minden tárgyra az azt tanító tanárok halmazát (TÁRGY)– a megoldás a legnagyobb elemszámú halmaz elemszáma!

C. NEM HALMAZOS FELADAT

D. Állítsuk elő minden tanárra az órái halmazát (ÓRÁK) – a megoldás egy olyan halmaz sorszáma, aminek a T tanár halmazával nincs közös eleme – azaz soha nincs egyszerre órájuk!

```
Beolvasás:
  Ciklus amíg nem vége(f)
    Olvas(f,i,j,k,l)
    Halmazba(nap[k],i)
    Halmazba(tárgy[j],i)
    Halmazba(órák[i],[k,l])
  Ciklus vége
Eljárás vége.
```

```
A:
  Ciklus i=1-től 5-ig
    adb[i]:=0
    Ciklus j=1-től n-ig
      Ha eleme(j,nap[i]) akkor adb[i]:=adb[i]+1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

Megjegyzés: az első ábrázolás esetén tároljuk a halmaz elemszámát, azaz egyszerűbb megoldást kaphatunk:

```
A:
  Ciklus i=1-től 5-ig
    adb[i]:=nap[i].db
  Ciklus vége
Eljárás vége.
```

```
B:
  tantárgy:=0; maxdb:=0
  Ciklus j=1-től m-ig
    db:=0
    Ciklus i=1-től n-ig
      Ha eleme(i,tárgy[j]) akkor db:=db+1
    Ciklus vége
    Ha db>maxdb akkor maxdb:=db; tantárgy:=j
  Ciklus vége
Eljárás vége.
```

Megjegyzés: az első ábrázolás esetén tároljuk a halmaz elemszámát, azaz egyszerűbb megoldást kaphatunk:

```
B:
  tantárgy:=0; maxdb:=0
  Ciklus j=1-től m-ig
    db:=tárgy[j].db
    Ha db>maxdb akkor maxdb:=db; tantárgy:=j
  Ciklus vége
Eljárás vége.

D:
  i:=1; Ha i=t akkor i:=i+1
  Ciklus amíg i≤n és nem üres?(metszet(órák[t],órák[i]))
    i:=i+1; Ha i=t akkor i:=i+1
  Ciklus vége
  Ha i≤n akkor DS:=i
Eljárás vége.
```

2. 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 (`tanár.pas`, …), amely megadja:

A. minden napra a szabadnapos tanárok számát;
B. azt a tanárt, akinek a legkevesebb lyukasórája van (lyukasóra: aznap előtte is van órája valamikor és utána is van órája valamikor);
C. az adott T tanárt egész héten helyettesíteni tudó tanárt (ha lehetséges, akkor úgy, hogy szabad napján senkit ne kelljen behívni az iskolába).
D. adott T tanárt a hét H. napján helyettesítő tanárokat úgy, hogy minden óráján szakos helyettesítés legyen;

A `tanár.be` szöveges állomány első sorában a tanárok száma ($1 \le N \le 100$), a tantárgyak száma ($1 \le M \le 100$), egy tanár sorszáma ($1 \le T \le N$) és egy nap sorszáma van ($1 \le H \le 5$), egy-egy szóközzel elválasztva. A következő sorok mindegyikében 4 egész szám van, egy-egy szóközzel elválasztva: tanár sorszáma, 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.

A `tanár.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, a legkisebb sorszámút kell megadni! Ha nincs megoldás (C és D részfeladatban), akkor -1-et kell kiírni! Az első sorban 5 szám szerepeljen, egy-egy szóközzel elválasztva! A negyedik sor első száma a helyettesített órák száma legyen, amit a helyettesítő tanárok sorszámai követnek, órák szerinti sorrendben, egy-egy szóközzel elválasztva!

Példa:

```
tanár.be    tanár.ki
```

```
3 4 1 1     1 0 2 3 3
1 1 1 6     2
1 1 2 2     3
1 2 1 3     2 2 2
2 1 2 3
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				2. óra	T4		
3. óra	T2			3. óra		T1		3. óra			
4. óra				4. óra				4. óra	T2		
5. óra				5. óra				5. óra			
6. óra	T1			6. óra				6. óra			

A. Állítsuk elő minden napra az aznap órát nem tartó tanárok halmazát (NAP) – a megoldás e halmazok elemszáma!

B. NEM HALMAZOS FELADAT

C. Állítsuk elő minden tanárra az órái halmazát (ÓRÁK) – a megoldás egy olyan halmaz I sorszáma, aminek a T tanár halmazával nincs közös eleme és a T tanárnak nincs órája az I tanár szabadnapjain (ehhez kell minden tanárra a szabadnapjai halmaza – SZABAD)!

D. Állítsuk elő minden tanárra az órái halmazát (ÓRÁK) – a megoldás egy olyan tanárhalmaz, akik órái halmazában a T tanár órái úgy szerepelnek, hogy az adott órához tartozó tantárgy benne van az illető tanár által tanított tárgyak halmazában (TANÍT)!

```
Beolvasás:
  nap[1..5]:=[1..n]; szabad[1..n]:=[1..5]
  Ciklus amíg nem vége(f)
    Olvas(f,i,j,k,l)
    Halmazból(nap[k],i)
    Halmazból(szabad[i],j)
    Halmazba(tárgy[j],i)
    Halmazba(órák[i],[j,k,l])
    Halmazba(tanít[i],j)
  Ciklus vége
Eljárás vége.
```

```
A:
  Ciklus i=1-től 5-ig
    adb[i]:=0
    Ciklus j=1-től n-ig
      Ha eleme(j,nap[i]) akkor adb[i]:=adb[i]+1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

```
C:
  i:=1; Ha i=t akkor i:=i+1
  Ciklus amíg i≤n és (nem üres?(metszet(órák[t],órák[i])) vagy
                      nem része(szabad[i],szabad[t])
    i:=i+1; Ha i=t akkor i:=i+1
  Ciklus vége
  Ha i≤n akkor CS:=i különben {előző feladat D része}
Eljárás vége.
```

```
D:
  DH:=[]
  Ciklus j=0-tól 7-ig
    Ha eleme([?,h,j],órák[t]) akkor
      x:=elemérték([?,h,j],órák[t])
      i:=1; Ha i=t akkor i:=i+1
      Ciklus amíg i≤n és (nem eleme(x,órák[i]))
                      vagy nem eleme(x,tanít[i])
        i:=i+1; Ha i=t akkor i:=i+1
      Ciklus vége
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Megjegyzés: az eleme és az elemérték műveletpár egyszerre is megvalósítható, ha ügyes ábrázolást választunk.

3. feladat

Egy közösségi portálra N ember jelentkezett be. Mindenki megadta, hogy kiket ismer, amit az ismerősük vissza is igazolt.

Készíts programot (*ISMER.PAS, ISMER.C*, ...), amely megadja azon párokat

A. akiknek minden ismerősük közös (de van legalább 1);

B. akiknek van közös ismerősük!

Az *ISMER.BE* szöveges állomány első sorában az emberek N száma ($1 \le N \le 100$) és az ismeretségek M száma ($0 \le M \le 5000$) van, egy szóközzel elválasztva. A következő M sor mind-

egyike két egymást ismerő ember sorszámát tartalmazza ($1 \le i \ne j \le N$), egy szóközzel elválasztva.

Az *ISMER.KI* szöveges állomány első sorába azon párok K számát kell írni, akiknek minden ismerősük közös, a következő K sorba pedig egy-egy ilyen pár sorszámát, a sorszámokat egy szóközzel elválasztva! A következő sorba azon párok L számát kell írni, akiknek van közös ismerősük, s az ezt követő L sorba pedig egy-egy ilyen pár sorszámát, a sorszámokat egy szóközzel elválasztva!

Példa:

ISMER.BE

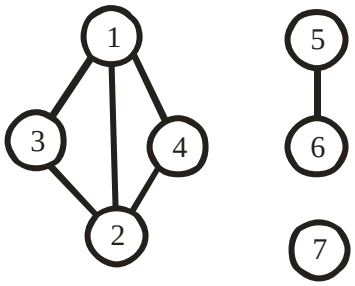
```
7 6
1 2
1 3
1 4
3 2
4 2
5 6
```

ISMER.KI

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

ismerőseik: 3,4

ismerőseik: 1,2



Számoljuk ki mindenkire az ismerősei halmazát! Az A részfeladat megoldásai azon (i,j) párok, amelyek ismerősei közösek (figyelve arra, hogy i-nek ismerőse lehet j, j-nek pedig i, de nem biztos) és legalább egy van belőlük. A B részfeladat megoldásai pedig azon (i,j) párok, amelyek ismerősei halmazának metszete nem üres.

```
Ismer(N,x,y,M,K,mind,L,van):
  ism:=({},…,{})
  Ciklus i=1-től M-ig
    Halmazba(ism(x[i]),y[i])
    Halmazba(ism(y[i]),x[i])
  Ciklus vége
  K:=0; L:=0
  Ciklus i=1-től N-1-ig
    Ciklus j=i+1-től N-ig
      Ha különbség(ism[i],{j})=különbség(ism[j],{i}) és
              nem üres(különbség(ism[i],{j})
         akkor K:=K+1; mind[K,1]:=i; mind[K,2]:=j
      Ha nem üres(metszet(ism[i],ism[j]))
         akkor L:=L+1; van[L,1]:=i; van[L,2]:=j
    Ciklus vége
  Ciklus vége
Eljárás vége.
```