ÚJ SZÉCHENYI TERV

Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Összetett programozási tételek 2

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 több bemenő sorozat alapján egy sorozatot kell előállítanunk.

Metszet

A következő három feladattípus újabb feladattípus osztályba tartozik, ezekben **több sorozathoz kell egyet rendelni**.

F1. Adjuk meg két természetes szám osztói ismeretében az összes közös osztójukat!

F2. Nyáron és télen is végeztünk madármegfigyeléseket a Balatonon. Ismerjük, hogy nyáron, illetve télen mely madárfajok fordultak elő. Állapítsuk meg ezek alapján, hogy melyek a nem költöző madarak!

F3. Négy ember heti szabad estéi ismeretében állapítsuk meg, hogy a héten melyik este mehetnek el együtt moziba!

Közös jellemzőjük e feladatoknak, hogy valahány – alapesetben itt is kettő – halmaz elemei közül azokat kell *kiválogat*nunk, amelyek mindegyikben előfordulnak. E halmazok elemeit – a többi programozási tétel tárgyalásmódjához hasonlóan – egy sorozatban (tömbben) felsoroljuk.

Kérdés: Miért nem a sorozatszámítás tételt alkalmazzuk e feladat megoldására?

Válasz: A sorozatszámítás tételben halmazokkal végezhettünk műveleteket, itt pedig olyan sorozatokkal, amelyek halmazelemeket tartalmaznak felsorolva. Emiatt a sorozatszámításban említett, két halmaz közötti metszetműveletet így lehet végrehajtani, ha a halmazokat sorozatként ábrázoljuk.

A megoldásban válogassuk ki X olyan elemeit, amelyek benne vannak Y-ban! Az algoritmus tehát egy kiválogatás, amely a belsejében egy eldöntést tartalmaz.

Az algoritmus:

```
Változók:
    N,M: Egész          (a feldolgozandó sorozat elemei száma)
    X,Y: Tömb(1..N:Elemtípus)(feldolgozandó sorozatok elemei)
    Db: Egész                         (a közös elemek száma)
    Z: Tömb(1..min(N,M):Egész)              (a közös elemek)
```

```
Metszet(N,X,M,Y,Db,Z):
  Db:=0
  Ciklus i=1-től N-ig                                  (kiválogatás)
    j:=1
    Ciklus amíg j≤M és X(i)≠Y(j)                          (eldöntés)
      j:=j+1
    Ciklus vége
    Ha j≤M akkor Db:=Db+1; Z(Db):=X(i)
  Ciklus vége
Eljárás vége.
```

E feladattípus hasonlítható több, az elemi programozási tételek között szerepelt feladattípushoz, ha a feladatot némileg módosítjuk.

Eldöntés: van-e a két halmaznak közös eleme?

Kiválasztás: adjuk meg a két sorozat egyik közös elemét (ha tudjuk, hogy biztosan van ilyen)!

Keresés: ha van, akkor adjuk meg a két sorozat egyik közös elemét!

Megszámolás: hány közös eleme van a két sorozatnak?

Nézzük meg ezek közül például a *keresés*t!

A megoldásban keressünk X-ben egy olyan elemeit, amely benne van Y-ban! Az algoritmus tehát egy keresés, amely a belsejében egy eldöntést tartalmaz.

```
Változók:
    N,M: Egész          (a feldolgozandó sorozat elemei száma)
    X,Y: Tömb(1..N:Elemtípus)(feldolgozandó sorozatok elemei)
    Van: Logikai                            (van-e közös elem)
    E: Elemtípus                              (egy közös elem)
Metszetbeli elem(N,X,M,Y,Van,E):
  i:=1; Van:=hamis
  Ciklus amíg i≤N és nem VAN
    j:=1
    Ciklus amíg j≤M és X(i)≠Y(j)
      j:=j+1
    Ciklus vége
    Ha j≤M akkor Van:=igaz; E:=X(i) különben i:=i+1
  Ciklus vége
Eljárás vége.
```

Egyesítés (unió)

Ha halmazokról van szó, akkor a metszet mellett természetesen meg kell jelennie az uniónak is.

F4. Két szám prímosztóinak ismeretében adjuk meg legkisebb közös többszörösük prímosztóit!

F5. Egy iskola két földrajztanára órarendjének ismeretében adjuk meg azokat az órákat, amelyben valamelyikük tud egy órát helyettesíteni!

Ebben a feladattípusban tehát azon elemekre vagyunk kíváncsiak, amelyek két halmaz közül legalább az egyikben előfordulnak.

A megoldásban másoljuk le X elemeit Z-be, majd válogassuk ki Y olyan elemeit, amelyek nincsenek benne X-ben! Az algoritmus tehát egy másolás, majd egy kiválogatás, amely a belsejében egy eldöntést tartalmaz.

```
Változók:
    N,M: Egész          (a feldolgozandó sorozat elemei száma)
    X,Y: Tömb(1..N:Elemtípus)(feldolgozandó sorozatok elemei)
    Db: Egész                           (a közös elemek száma)
    Z: Tömb(1..N+M:Egész)                     (a közös elemek)

Egyesítés(N,X,M,Y,Db,Z):
  Z:=X; Db:=N                                    (másolás)
  Ciklus j=1-től M-ig                        (kiválogatás)
    i:=1
    Ciklus amíg i≤N és X(i)≠Y(j)                (eldöntés)
      i:=i+1
    Ciklus vége
    Ha i>N akkor Db:=Db+1; Z(Db):=X(i)
  Ciklus vége
Eljárás vége.
```

A példák megoldása helyett itt is egy alkalmazást nézzünk: egy sorozatból készítsünk halmazfelsorolást! A feladat tehát az, hogy másoljuk le egy sorozat elemeit, de az azonos értékű elemek az eredményben csak egyszer szerepeljenek.

Ez egy furcsa egyesítés, $Z=Z\cup X$ képlettel írhatnánk le, azaz azokat az elemeket vegyük bele X-ből az eredménybe, amelyek még nem szerepelnek benne.

```
Halmazfelsorolás_készítés(N,X,Db,Z):
  Db:=0
  Ciklus i=1-től N-ig
    j:=1
    Ciklus amíg j≤Db és X(i)≠Z(j)
      j:=j+1
    Ciklus vége
    Ha j>Db akkor Db:=Db+1; Z(Db):=X(i)
  Ciklus vége
Eljárás vége.
```

Összefuttatás (rendezettek uniója)

Bizonyos esetekben az unió feladattípus megoldása az előző fejezetbelinél hatékonyabb lehet. Ezt is konkrét feladatokon keresztül vizsgáljuk.

F6. Egy osztály lány-, illetve fiútanulóinak névsora alapján állítsuk elő az osztálynévsort!

F7. Egy iskolában négy szakkörre járnak tanulók (van, aki többre is). A szakkörnévsorok alapján állítsuk elő a szakkörre járó tanulók névsorát!

Megállapíthatjuk, hogy az általános egyesítéshez képest itt az a specialitás, hogy mindegyik sorozatként ábrázolt halmaz rendezett, s az eredménynek is rendezettnek kell lenni.

Ha nem két sorozatról van szó, akkor az a korábbiaknak megfelelően visszavezethető két sorozat feldolgozására.

A megoldásban haladjunk párhuzamosan a két sorozatban! Az eredmény első eleme vagy X_1, vagy Y_1 lesz. Amelyik kisebb, azt az eredménysorozatba tesszük, abban a sorozatban kell továbblépni egy elemmel, s újra egy-egy elemet hasonlítani. Ha egyenlők voltak, akkor az egyiket másoljuk az eredménybe, majd mindkét sorozatban továbblépünk. Ha az egyiknek a végére értünk, akkor a másikat minden változtatás nélkül az eredménybe másoljuk.

```
Változók:
    N,M: Egész          (a feldolgozandó sorozat elemei száma)
    X,Y: Tömb(1..N:Elemtípus)(feldolgozandó sorozatok elemei)
    Db: Egész                          (a közös elemek száma)
    Z: Tömb(1..N+M:Egész)                    (a közös elemek)
Összefuttatás(N,X,M,Y,Db,Z):
  i:=1; j:=1; Db:=0
  Ciklus amíg i≤N és j≤M
    Db:=Db+1
    Elágazás
      X(i)<Y(j) esetén Z(Db):=X(i); i:=i+1
      X(i)=Y(j) esetén Z(Db):=X(i); i:=i+1; j:=j+1
      X(i)>Y(j) esetén Z(Db):=Y(j); j:=j+1
    Elágazás vége
  Ciklus vége
  Ciklus amíg i≤N
    Db:=Db+1; Z(Db):=X(i); i:=i+1
  Ciklus vége
  Ciklus amíg j≤M
    Db:=Db+1; Z(Db):=Y(j); j:=j+1
  Ciklus vége
Eljárás vége.
```

Észrevehetjük a megoldást elemezve, hogy ha olyan szerencsénk volt, hogy X(N)=Y(M), akkor az utolsó két ciklusra tulajdonképpen nincs is szükség, hiszen nem maradt másolnivaló. Az a baj, hogy ez a szerencsés eset viszonylag ritkán fordul elő.

A második megoldásváltozatban mi magunk idézzük elő e szerencsét: mindkét sorozat végére egy nagyon nagy (az adott típus legnagyobb értéke), de egyező elemet teszünk.

```
Összefuttatás(N,X,M,Y,Db,Z):
  i:=1; j:=1; Db:=0
  X(N+1):=+∞; Y(M+1):=+∞   (az elemtípus maximális értéke)
  Ciklus amíg i<N+1 vagy j<M+1
    Db:=Db+1
    Elágazás
      X(i)<Y(j) esetén Z(Db):=X(i); i:=i+1
      X(i)=Y(j) esetén Z(Db):=X(i); i:=i+1; j:=j+1
      X(i)>Y(j) esetén Z(Db):=Y(j); j:=j+1
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Ebben a megoldásban a hozzávett fiktív elem nem kerül be az eredménybe. Ha szükségünk lenne rá, a ciklus vége után még elhelyezhetnénk az eredmény végére.

A harmadik változatban kihasználjuk azt a – nem minden feladatban meglevő – specialitást, hogy a két sorozatban biztosan nincs közös elem. Ilyen például a fejezet elején szereplő F6 feladat. Ezt az altípust a megkülönböztetés érdekében **összefésülés**nek nevezzük.

```
Összefésülés(N,X,M,Y,Db,Z):
  i:=1; j:=1; Db:=0
  X(N+1):=+∞; Y(M+1):=+∞   (az elemtípus maximális értéke)
  Ciklus amíg i<N+1 vagy j<M+1
    Db:=Db+1
    Ha X(i)<Y(j) akkor Z(Db):=X(i); i:=i+1
              különben Z(Db):=Y(j); j:=j+1
  Ciklus vége
Eljárás vége.
```

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

1. feladat

Egy programozási versenyen minden versenyző választhat egy programozási nyelvet, amin dolgozni fog.

Készíts programot a következő feladat megoldására, A programod olvassa be a választható nyelvek számát (1≤M≤10) és a versenyen induló tanulók számát (1≤N≤100), majd a választható nyelveket, s legvégül az egyes tanulók által választott nyelveket! Ezután a program adja meg, hogy mely tanulók választottak illegális nyelvet (olyat, ami nem szerepelt a felsoroltak között), mely nyelveket nem választotta senki, s melyik választott nyelvet hányan választották!

Példa:

```
Nyelvek száma: 3          ⇒  Illegális nyelv: 3. versenyző
Versenyzők száma: 5          Nem választott nyelv: Logo
Választható nyelvek:         Választott nyelvek:
    Pascal                           Pascal: 3 versenyző
    Logo                             C++: 1 versenyző
    C++
Választott nyelvek:
    Pascal
    Pascal
    Delphi
    C++
    Pascal
```

A feladatban adott egy halmaz (a választható nyelvek) és egy multihalmaz (a választott nyelvek) – az utóbbi abban furcsa, hogy az elemek sorszáma is fontos, emiatt nem lehet akármilyen halmazábrázolást választani.

Az első részfeladat megoldása azon sorszámok a multihalmazból, amelyekhez a halmazban nem tartozik elem.

A második részfeladatban ki kell számítani a halmaz és a multihalmaz „különbségét".

A harmadik részfeladat pedig a multihalmaz tényleges előállítása.

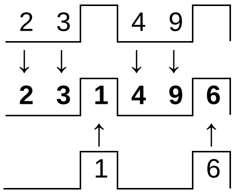
```
Nyelvek(N,t,M,v):
  Ciklus i=1-től N-ig
    Ha nincs(t(i),v,m) akkor Ki: 'Illegális nyelv: ',i,t(i)
  Ciklus vége
  Ciklus i=1-től M-ig
    Ha nincs(v(i),t,n) akkor Ki: 'Nem választott: ',v(i)
  Ciklus vége
  s:=(0,…,0)
  Ciklus i=1-től N-ig
    Keres(t(i),van,j); Ha van akkor s(j):=s(j)+1
  Ciklus vége
  Ciklus i=1-től M-ig
    Ha s(i)>0 akkor Ki: v(i),s(i)
  Ciklus vége
Eljárás vége.

Keres(s,van,i):
  i:=1
  Ciklus amíg i≤M és s≠v(i)
    i:=i+1
  Ciklus vége
  keres:=i
Eljárás vége.
```

```
nincs(s,v,db):
  i:=1
  Ciklus amíg i≤db és s≠v(i)
    i:=i+1
  Ciklus vége
  nincs:=(i>db)
Függvény vége.
```

2. feladat

Adott egy egész számokból álló sorozat.

Készíts programot, amely kiszámít két olyan monoton növekvő sorozatot, amelyekből fésűs egyesítéssel megkapható a bemeneti sorozat!



A *SOROZAT.BE* szöveges állomány első sorában a bemeneti sorozat elemeinek N száma (***0≤N≤1000***) van. A második sor pontosan ***N*** egész számot tartalmaz egy-egy szóközzel elválasztva, a bementi sorozatot. A sorozat minden ***x*** elemére teljesül, hogy (***0≤x≤30000***).

A *SOROZAT.KI* szöveges állomány első és egyetlen sora a ***0 0*** számpárt tartalmazza, ha a bemeneti sorozat nem állítható elő két monoton növekvő sorozat fésűs egyesítéseként. Egyébként az első sorban a fésűs egyesítéshez kiszámított első sorozat ***K*** elemszáma álljon. A második sor pontosan ***K*** számot tartalmazzon, az első sorozat elemeit, egy-egy szóközzel elválasztva. A harmadik sor tartalmazza a második sorozat ***L*** elemszámát. A negyedik sor pontosan ***L*** számot tartalmazzon, a második sorozat elemeit, egy-egy szóközzel elválasztva. A kiszámított sorozatok egyike üres is lehet, ekkor a ***0*** számot és üres sort kell kiírni. A bemeneti sorozat minden eleme pontosan egyik kiszámított sorozat eleme. Több megoldás esetén bármelyik megadható.

Példa:

SOROZAT.BE	*SOROZAT*.KI
6	4
2 3 1 4 9 6	2 3 4 9
	2
	1 6

Sorra vizsgáljuk a bemenet elemeit. Ha a következő elem egyik sorozatba sem tehető (mert mindkettő utolsójánál kisebb), akkor nincs megoldás. Ha az első sorozat utolsó eleme a kisebb, akkor először próbáljuk a másodikba tenni! Ha a másodiké a kisebb, akkor pedig az elsőbe!

```
Sorozat(N,A,N1,A1,N2,A2,van):
  N1:=0; N2:=0; A1(0):=0; A2(0):=0
  van:=igaz; i:=1
  Ciklus amíg i≤N és van
    Ha A(i)<A1(N1) és A(i)<A2(N2) {A(i) egyikbe sem tehető}
       akkor N1:=0; N2:=0; van:=hamis
    különben ha A1(N1)<A2(N2) {az első végén van a kisebb}
                akkor ha A2(N2)≤A(i)
                         akkor N2:=N2+1; A2(N2):=A(i)
                      különben N1:=N1+1; A1(N1):=A(i)
                {a második végén van a kisebb}
    különben ha A1(N1)≤A(i) akkor N1:=N1+1; A1(N1):=A(i)
                         különben N2:=N2+1; A2(N2):=A(i)
  Ciklus vége
Eljárás vége.
```

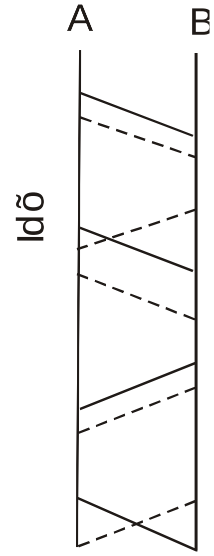
3. feladat

Egy vállalat két telephelye (A és B) között csomagok kézbesítésére két futárt alkalmaz. A futárok a távolságot mindig ***O*** perc alatt teszik meg. Ha éppen szemben haladnak egymással, akkor találkozhatnak.

Készíts programot, amely megadja, hogy a futárok hányszor, illetve hányadik kézbesítésükkor találkozhatnak egymással út közben!

A *FUTAR.BE* szöveges állomány első sorában az első és a második futár kézbesítéseinek száma (***1≤N≤1000, 1≤M≤1000***), továbbá a távolág megtételéhez szükséges idő (***1≤O≤100***) van, egy-egy szóközzel elválasztva. A következő N sorban az első, az azt követő M sorban pedig a második futár kézbesítéseit írtuk le, mindegyiket indulási idő szerint növekvő sorrendben. Minden kézbesítéshez tartozó sor egy betűvel (A vagy B) kezdődik – annak a telephelynek az azonosítójával, ahonnan a futárnak el kell indulnia. Ezt követi egy szóközzel elválasztva az indulás ideje (0≤idő≤20000). (Feltesszük, hogy a futár az indulás idejében a megfelelő telephelyen van.)

A *FUTAR.KI* szöveges állomány első sorába a találkozások ***K*** számát kell írni! A következő ***K*** sor mindegyikében két szám legyen egy szóközzel elválasztva: az első és a második futár kézbesítésének sorszáma, ami alatt találkozhatnak. Ezek a sorok a találkozási idő szerint növekvő sorrendben legyenek!



Példa: (az ábrán a második futár útját szaggatott vonallal jelöltük)

```
FUTAR.BE

4 5 10
A 10
A 40
B 70
A 100
A 15
```

```
FUTAR.KI

2
2 2
4 5
```

```
B 35
A 50
B 75
B 100
```

A két indulási idő sorozatot össze kell futtatni, s nézni, hogy a két futár szembe megy-e egymással. Ha szembe mennek, akkor meg kell nézni, hogy találkozhatnak-e (azaz az egyik beérkezési ideje előtt indul-e a második, vagy fordítva).

```
Futár(N,f1,M,f2,k,er):
  f1(n+1).mikor:=maxint; f2(m+1).mikor:=maxint
  i:=1; j:=1; k:=0
  Ciklus amíg i<n+1 vagy j<m+1
    Ha f1(i).mikor<f2(j).mikor
       akkor Ha f1(i).honnan≠f2(j).honnan
                akkor Ha f1(i).mikor+o>f2(j).mikor
                       akkor k:=k+1; er(k,1):=i; er(k,2):=j
             i:=i+1
    különben ha f1(i).mikor=f2(j).mikor
       akkor Ha f1(i).honnan≠f2(j).honnan
                akkor k:=k+1; er(k,1):=i; er(k,2):=j
             i:=i+1; j:=j+1
    különben {f1(i).mikor>f2(j).mikor}
             Ha f1(i).honnan≠f2(j).honnan
                akkor Ha f2(j).mikor+o>f1(i).mikor
                       akkor k:=k+1; er(k,1):=i; er(k,2):=j
       j:=j+1
  Ciklus vége
Eljárás vége.
```