Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Programozási tételek összeépítése

TÁMOP-4.2.3.-12/1/KONV

Gyakran előfordul, hogy programozási tételeket egymás után kell használnunk. Ezen egymásutániságnál azonban sok esetben a két megoldó algoritmus egybeépítése egyszerűbb, rövidebb, hatékonyabb megoldást eredményez. Ebben a részben ezekkel foglalkozunk. Az egymásra építés mindig két programozási tétel összefogását jelenti, a fejezeteket a korábban alkalmazandó tétel szerint fogalmaztuk meg.

Kiválogatással összeépítés

Elsőként a *kiválogatás*t a *sorozatszámítás*sal építjük egybe. Olyan feladatoknál alkalmazható ez, amikor a számítást egy sorozat T tulajdonságú elemeire kell csak elvégeznünk.

Az algoritmus:

Függvény:

```
T: Elemtípus → Logikai
```

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]
```

A megoldásban vegyük a sorozatszámítás algoritmusát, de a műveletet ne minden esetben végezzük el, hanem csak akkor, ha a megfelelő elem T tulajdonságú! Sokszor nincs az eredményhez szükség a Db-re, a mostani megoldásban ezt is képezzük.

```
Kiválogatás_sorozatszámítás(N,X,S,Db):
  S:=F0:  Db:=0
  Ciklus i=1-től N-ig
    Ha T(X(i)) akkor Db:=Db+1; S:=f(S,X(i))
  Ciklus vége
Eljárás vége.
```

Második példánkban a *kiválogatás*t a *maximumkiválasztás*sal építjük egybe. Tipikusan ilyen feladatok azok, amelyekben meg kell határozni egy sorozat T tulajdonságú elemeinek maximumát, minimumát.

Az algoritmus:

Függvény:

```
T: Elemtípus → Logikai
```

```
Változók:
     N  : Egész           [a feldolgozandó sorozat elemei száma]
     X  : Tömb(1..N:Elemtípus) [a feldolgozandó sorozat elemei]
     Van: Logikai         [van-e maximális T tulajdonságú elem]
     Maxért: Elemtípus    [a maximális értékű elem értéke]
     Max: Egész           [a maximális értékű elem sorszáma]
```

Vegyük a megoldáshoz a maximumkiválasztás algoritmusát! Ne akkor jegyezzük fel az új elemet maximumnak, amikor az nagyobb az addigi maximumnál, hanem ennek még az is legyen a feltétele, hogy az új elem T tulajdonságú! Csupán egyetlen probléma van: semmi nem garantálja, hogy az első elem T tulajdonságú, sőt hogy egyáltalán ilyen elem lesz (amit pedig a maximumkiválasztás kihasznált).

Egyik lehetséges megoldás lenne, ha megkeresnénk az első T tulajdonságú elemet, s innen indítanánk a maximumkiválasztást. A másik – s mi ezt követjük – egy fiktív kezdőértékkel indít, s elölről vizsgálja a sorozatot. (Ha e fiktív érték marad a Maxért-ben az azt jelenti, hogy nincs T tulajdonságú elem a sorozatban.)

```
Kiválogatás_maximumkiválasztás(N,X,Van,Max,Maxért):
  Maxért:=-∞
  Ciklus i=1-től N-ig
    Ha T(X(i)) és X(i)>Maxért akkor Maxért:=X(i); Max:=i
  Ciklus vége
  Van:=(Maxért≠-∞)
Eljárás vége.
```

Egy másik megoldásban felveszünk egy 0. elemet, ami minimális értékű és T tulajdonságú. Ekkor a maximális értékű elem indexét határozzuk meg.

```
Kiválogatás_maximumkiválasztás(N,X,Van,Max):
  Max:=0; X(0):=-∞
  Ciklus i=1-től N-ig
    Ha T(X(i)) és X(i)>X(Max) akkor Max:=i
  Ciklus vége
  Van:=(Max≠0)
Eljárás vége.
```

A harmadik példában a *kiválogatás*t a *másolás*sal építjük egybe. Ezt olyan feladatoknál kell alkalmazni, ahol egy sorozat T tulajdonságú elemeit kell lemásolni, rajtuk egy függvény kiszámításával.

Az algoritmus:

```
Függvény:
     T: H_Elemtípus → Logikai
     f: H_elemtípus → G_elemtípus
```

```
Változók:
     N : Egész            [a feldolgozandó sorozat elemei száma]
     X : Tömb(1..N:H_elemtípus)    [feldolgozandó elemek]
     Db: Egész                 [a megfelelő elemek száma]
     Z : Tömb(1..N:G_elemtípus)    [feldolgozott elemek]
```

A megoldásban vegyük a kiválogatás kigyűjtéses algoritmusát, de ne a megfelelő elemek sorszámait gyűjtsük ki, hanem az ezen elemeken kiszámított függvényértéket!

```
Kiválogatás_másolás(N,X,Db,Z):
  Db:=0
  Ciklus i=1-től N-ig
    Ha T(X(i)) akkor Db:=Db+1: Z(Db):=f(X(i))
  Ciklus vége
Eljárás vége.
```

E fejezet algoritmusai nagyon hasonlóan alkalmazhatók *szétválogatás*sal, *metszet*tel, *unió*val való egymásra építésre. A szükséges módosításokat a *kiválogatás* ciklusa helyett a megfelelő programozási tétel ciklusában kell elvégezni.

Szétválogatással összeépítés

A szétválogatással összeépítés ugyanazon az elven megy, mint a kiválogatással összeépítés, csak egyszerűen duplázni kell a tennivalókat.

Nézzük például a szétválogatás maximumkiválasztással összeépítését!

Ne akkor jegyezzük fel az új elemet maximumnak, amikor az nagyobb az addigi maximumnál, hanem ennek még az is legyen a feltétele, hogy az új elem T tulajdonságú, illetve nem T tulajdonságú!

```
Szétválogatás_maximumkiválasztás(N,X,Van1,Max1,Van2,Max2):
  Maxért1:=-∞; Maxért2:=-∞
  Ciklus i=1-től N-ig
    Ha T(X(i)) akkor Ha X(i)>Maxért1 akkor Maxért1:=X(i)
                                           Max1:=i
            különben Ha X(i)>Maxért2 akkor Maxért2:=X(i)
                                           Max2:=i
  Ciklus vége
  Van1:=(Maxért1≠-∞); Van2:=(Maxért2≠-∞)
Eljárás vége.
```

A *szétválogatás* specialitása lehet, hogy az eredményeként szereplő két sorozatra különböző programozási tételeket is lehetne alkalmazni (például lehetne feladat a T tulajdonságú elemek maximumának és a nem T tulajdonságú elemek átlagának meghatározása a feladat).

```
Szétválogatás_maximum_átlag(N,X,Van,Max,Átlag):
  Maxért:=-∞; S:=0; Db:=0;
  Ciklus i=1-től N-ig
    Ha T(X(i)) akkor Ha X(i)>Maxért akkor Maxért:=X(i)
                                          Max:=i
            különben S:=S+X(i); Db:=Db+1
  Ciklus vége
  Van:=(Maxért≠-∞)
  Ha Db>0 akkor Átlag:=S/Db különben Átlag:=0
Eljárás vége.
```

Sok esetben csak egy belső részfeladat a szétválogatás. Nézzük például azt a feladatot, amiben arra vagyunk kíváncsiak, hogy a T vagy a nem T tulajdonságú elemek összege-e a nagyobb?

```
Szétválogatás_nagyobb(N,X,TöbbT):
  S1:=0; S2:=0
  Ciklus i=1-től N-ig
    Ha T(X(i)) akkor S1:=S1+X(i)
            különben S2:=S2+X(i)
  Ciklus vége
  TöbbT:=(S1>S2)
Eljárás vége.
```

Metszettel összeépítés

A metszettel összeépítés nagyon egyszerű, hiszen a metszet lényegében egy kiválogatás. Emiatt a metszet maximumkiválasztással így alakul:

```
Metszet_maximum(N,X,M,Y,Van,Maxért):
  Maxért:=-∞
  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 Ha X(i)>Maxért akkor Maxért:=X(i)
  Ciklus vége
  Van:=Maxért>-∞
Eljárás vége.
```

Egyetlen fontos különbség: itt nincs értelme maximális indexnek (hiszen ugyanaz a metszetbeli érték mindkét sorozatban szerepel), csak maximális értéknek.

Különlegessége a metszetnek (és az uniónak is), hogy a kiválogatással is kombinálható, értelmes feladat lehet két sorozat közös T tulajdonságú elemeinek megadása.

Ehhez kétféleképpen is hozzáállhatunk. Az első változatban a metszethez építjük hozzá a kiválogatást:

```
Metszet_kiválogatás(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 Ha T(X(i)) akkor Db:=Db+1; Z(Db):=X(i)
  Ciklus vége
Eljárás vége.
```

A második változatban a kiválogatáshoz építjük a metszetet, ezzel elkerüljük a felesleges belső eldöntést:

```
Metszet_kiválogatás(N,X,M,Y,Db,Z):
  Db:=0
  Ciklus i=1-től N-ig                              (kiválogatás)
    Ha T(X(i)) akkor
       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)
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

Egy harmadik feladat lehet a metszet és a rendezés összeépítése:

```
Metszet_rendezés(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 Beilleszt(Db,Z,X(i)); Db:=Db+1
  Ciklus vége
Eljárás vége.
```

Itt praktikus a beillesztéses rendezést használni, ami akkor is elkezdhető, ha még nem ismerjük a rendezendő sorozat összes elemét.

```
Beilleszt(Db,Z,E):
  i:=Db
  Ciklus amíg i>0 és Z(i)>E
    Z(i+1):=Z(i); i:=i-1
  Ciklus vége
  Z(i+1):=E
Eljárás vége.
```

Unióval összeépítés

Az unióval összeépítés is nagyon egyszerű, hiszen az unió lényegében egy kiválogatás. Van azonban olyan, amit érdemes a mechanikus alkalmazás helyett átgondolni, ilyen például az unió összeépítése a maximumkiválasztással.

A mechanikus megoldás:

```
Egyesítés_maximum(N,X,M,Y,Maxért):
  Maxért:=X(1)
  Ciklus i=2-től N-ig
    Ha X(i)>Maxért akkor Maxért:=X(i)
  Ciklus vége
  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 Ha Y(j)>Maxért akkor Maxért:=Y(j)
  Ciklus vége
Eljárás vége.
```

A lényeg: felesleges Y elemeiből azok kiválogatása, amelyek nem elemei X-nek, a kiválogatást folytathatjuk Y összes elemével. A maximumnál ugyanis nem zavaró, ha ugyanazt az elemet többször is megvizsgáljuk.

```
Egyesítés_maximum(N,X,M,Y,Maxért):
  Maxért:=X(1)
  Ciklus i=2-től N-ig
    Ha X(i)>Maxért akkor Maxért:=X(i)
  Ciklus vége
  Ciklus j=1-től M-ig
    Ha Y(j)>Maxért akkor Maxért:=Y(j)
  Ciklus vége
Eljárás vége.
```

Mivel az uniónak mindig van eleme, ezért Maxért is mindig lesz, nem kell felkészülni arra az esetre, amikor nincs maximális elem.

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

1. feladat

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

Írj programot, amely megadja azt a csapatot, amely a kiesettek közül a legtöbbször győzött!

Első lépésként minden csapatra leszámoljuk, hogy hány győzelme és hány veresége van. Bár vereségből egy kieséses versenyen maximum egy lehet, de mégis egyszerűbb megszámolásként kezelni.

Ezután a feladat egy kiválogatás és egy maximumkiválasztás összeépítése.

Legyen ismert N csapat, M mérkőzés eredménye!

```
Számolás(N,M,Gy,V):
  Gy:=(0,…,0); V:=(0,…,0)
  Ciklus i=1-től M-ig
    Be: A,B    {A legyőzte B-t}
    Gy(A):=Gy(A)+1
    V(B):=V(B)+1    (lehetne V(B):=1 is)
  Ciklus vége
Eljárás vége.
```

```
Összeépítés(N,Gy,V,Max):
  Max:=0
  Ciklus i=1-től N-ig
    Ha V(i)>0 akkor Ha Gy(i)>Gy(Max) akkor Max:=i
  Ciklus vége
Eljárás vége.
```

2. feladat

Két különböző vizsgálták, hogy középiskolások milyen programozási nyelvet tanulnak. Add meg azt a nyelvet, amit a mindkét évben tanultak közül a legkevesebben, illetve legtöbben tanultak!

Bemenetként két sorozatot kapunk, a két évben tanult nyelvek sorozatát. Ezek nem halmazok, hiszen ugyanazt a nyelvet többen is tanulhatták. Két úton járhatunk el. Az egyik megoldásban először előállítjuk a tanult nyelvek két multihalmazát, majd a metszetet összeépítjük a maximumkiválasztással:

```
Multihalmaz_előállítás(N,A,NN,AA):
  NN:=0
  Ciklus i=1-től N-ig
    j:=1
    Ciklus amíg j≤NN és A(i)≠AA(j)
      j:=j+1
    Ciklus vége
    Ha j>NN akkor NN:=NN+1; AA(NN).elem:=X(i); AA(NN).db:=1
         különben AA(j).db:=AA(j).db+1
  Ciklus vége
Eljárás vége.

Metszet_maximum(N,X,M,Y,Van,Maxért):
  Multihalmaz_előállítás(N,X,NN,XX)
  Multihalmaz_előállítás(M,Y,MM,YY)
  Maxért:=-∞
  Ciklus i=1-től NN-ig                              (kiválogatás)
    j:=1
    Ciklus amíg j≤MM és XX(i).elem≠YY(j).elem      (eldöntés)
      j:=j+1
    Ciklus vége
    Ha j≤MM akkor Ha XX(i).db+YY(j).db>Maxért
                  akkor Maxért:=XX(i).db+YY(j).db
  Ciklus vége
  Van:=Maxért>-∞
Eljárás vége.
```

A másik megoldásban csak az egyik multihalmazt állítjuk elő, majd megszámoljuk, hogy az elemei hányszor szerepelnek a másik felsorolásban: Ezután a közös elemekre maximumot határozunk meg.

```
Metszet_maximum(N,X,M,Y,Van,Maxért):
  Multihalmaz_előállítás(N,X,NN,XX)
  Maxért:=-∞
  Ciklus i=1-től NN-ig                              (kiválogatás)
    Db:=0
    Ciklus j=1-től M-ig
      Ha XX(i).elem=Y(j) akkor Db:=Db+1
    Ciklus vége
    Ha Db>0 akkor Ha XX(i).db+Db>Maxért
                  akkor Maxért:=XX(i).db+Db
  Ciklus vége
  Van:=Maxért>-∞
Eljárás vége.
```