ÚJ SZÉCHENYI TERV

Ég és Föld vonzásában – a természet titkai

Informatikai tehetséggondozás:

Rendezések

TÁMOP-4.2.3.-12/1/KONV

MAGYARORSZÁG MEGÚJUL

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

Az alapfeladat egy N elemű sorozat nagyság szerinti sorba rendezése. A sorozat elemei olyanok, amelyekre a <,≤ relációk léteznek. Feltesszük a feladatok mindegyikében, hogy a sorozathoz létezik indexelés művelet, s ezt a megoldásban ki is használjuk. Az eredmény a módszerek jelentős részében helyben keletkezik, az eredeti sorrend elvész.

Konkrét feladatok ismertetése és megoldása helyett most egy számpéldát fogunk végignézni az egyes módszereknél. E számsorozat - egy kivétellel - a következő lesz: 5, 3, 9, 1, 7.

```
Változók:
    N  : Egész          [a feldolgozandó sorozat elemei száma]
    X  : Tömb(1..N:H_Elemtípus)     [a feldolgozandó sorozat]
```

Az egyes módszereket összehasonlítjuk tárigény (a rendezésben résztvevő tároló helyek száma), valamint végrehajtási idő (hasonlítások száma, mozgatások száma) szerint. A helyfoglalásba nem számítjuk bele a ciklusváltozókat, a végrehajtási időbe pedig ezek növelését, vizsgálatát, hiszen ezek mérete és ideje nem függ a rendezendő elemek típusától.

A hasonlítások és a mozgatások száma néha nem függ a rendezendő elemek értékétől, a legtöbb esetben azonban igen, így csak minimális és maximális számukról beszélhetünk.

1. Minimum-kiválasztásos rendezés

Az első módszer a következő ötletre épül. Hasonlítsuk össze az első elemet a sorozat összes többi mögötte levő elemével, s cseréljük meg közülük a legkisebbel! Ezzel elérhetjük, hogy a sorozat első helyére a legkisebb elem kerül. Folytassuk ugyanezen elven a sorozat második elemével, utoljára pedig az utolsóelőttivel.

```
Rendezés(N,X):
  Ciklus I=1-től N-1-ig
    MIN:=I
    Ciklus J=I+1-től N-ig
      Ha X(MIN)>X(J) akkor MIN:=J
    Ciklus vége
    Csere(X(I),X(MIN))
  Ciklus vége
Eljárás vége.
```

Itt a számpéldát a külső ciklus ciklusfeltételének kiértékelése előtt vizsgáljuk:

I=1 ⇒ 5, 3, 9, 1, 7
I=2 ⇒ 1, 3, 9, 5, 7
I=3 ⇒ 1, 3, 9, 5, 7
I=4 ⇒ 1, 3, 5, 9, 7

Helyfoglalás:	N+1
Hasonlítások száma:	N*(N-1)/2
Mozgatások száma:	3*(N-1)

2. Buborékos rendezés

Most egy másik rendezési alapelvet vizsgálunk meg: hasonlítsuk egymással a szomszédos elemeket, s ha a sorrendjük nem jó, akkor cseréljük meg őket!

Ezzel a módszerrel egy cikluslépés lefutása alatt a legnagyobb elem biztosan a sorozat végére kerül, s ezen kívül a nagyobb értékű elemek hátrafelé, a kisebbek pedig előrefelé mozdulnak el (innen van a buborékmódszer elnevezés).

Figyeljük tehát minden menetben a legutolsó csere helyét, s a következő menetben csak addig rendezzünk!

```
Rendezés(N,X):
  I:=N
  Ciklus amíg I≥2
    CS:=0
    Ciklus J=1-től I-1-ig
      Ha X(J)>X(J+1) akkor Csere(X(J),X(J+1)); CS:=J
    Ciklus vége
    I:=CS
  Ciklus vége
Eljárás vége.
```

Újra a belső ciklusnál nézzük a konkrét rendezési példát:

I=5 ⇒	5, 3, 9, 1, 7
	3, 5, 9, 1, 7
	3, 5, 9, 1, 7
	3, 5, 1, 9, 7
	3, 5, 1, 7, 9
I=4 ⇒	3, 5, 1, 7, 9
	3, 5, 1, 7, 9
	3, 1, 5, 7, 9
	3, 1, 5, 7, 9
I=2 ⇒	1, 3, 5, 7, 9
	1, 3, 5, 7, 9

Helyfoglalás:	N+1
Hasonlítások száma:	N-1 - N*(N-1)/2
Mozgatások száma:	0 - 3*N*(N-1)/2

A hasonlítások számában javulást látunk, az igazi javulás azonban nem a minimális és a maximális, hanem az átlagos végrehajtási időben van.

3. Beillesztéses rendezés

Újabb rendezési elvvel ismerkedünk meg. Az eddigi módszerek mindegyike olyan volt, hogy a sorozatot felosztotta egy már kész, rendezett szakaszra, s a rendezést a másik szakasz elemei között folytatta. Másik jellemzőjük volt, hogy a rendezés elkezdéséhez már az összes elemnek rendelkezésre kellett állnia.

Most a kártyakeveréshez hasonló elvből indulunk ki: egyetlen elem mindig rendezett; ha van egy rendezett részsorozatunk, akkor abba a nagyság szerinti helyére illesszük be a soron következő elemet! A beillesztendőt nem tesszük azonnal be a sorozatba, hanem csak a többieket tologatjuk hátra, s a beillesztendőt csak a végén tesszük a helyére

```
Rendezés(N,X):
  Ciklus I=2-től N-ig
    J:=I-1; Y:=X(I)
    Ciklus amíg J>0 és X(J)>Y
      X(J+1):=X(J); J:=J-1
    Ciklus vége
    X(J+1):=Y
  Ciklus vége
Eljárás vége.
```

Az elemek mozgatása miatt e példában a kivett elem visszahelyezéséig egyes elemek kétszer szerepelnek.

I=2 ⇒	5, 3, 9, 1, 7
	5, 5, 9, 1, 7
I=3 ⇒	3, 5, 9, 1, 7
I=4 ⇒	3, 5, 9, 1, 7
	3, 5, 1, 9, 7
	3, 1, 5, 9, 7
	3, 3, 5, 9, 7
I=5 ⇒	1, 3, 5, 9, 7
	1, 3, 5, 9, 9

Helyfoglalás:	N+1
Hasonlítások száma:	N-1 - N*(N-1)/2
Mozgatások száma:	2*(N-1) - 2*(N-1)+N*(N-1)/2

4. Szétosztó rendezés

A három alaprendezés, s azok javításai után speciális rendezésekkel foglalkozunk, amelyekben előfeltételként speciális megszorításaink lesznek.

A legelsőben feltesszük, hogy a rendezendő elemek olyan rekordok, amelyek kulcsmezője (vagy egy abból kiszámolt számérték) 1 és N közötti természetes szám lehet, s nincs két azonos kulcsú rekord.

A kulcsmező itt egyértelműen megadja azt a helyet, ahova az elemet tenni kell, így semmi hasonlításra nincs szükség. A módszerhez azonban egy újabb tömbre van szükség, ahova az eredményt elhelyezzük.

```
Változók:
     N  : Egész          [a feldolgozandó sorozat elemei száma]
     X,Y: Tömb(1..N:H_Elemtípus)             [a két sorozat]
```

```
Rendezés(N,X,Y):
  Ciklus I=1-től N-ig
    Y(X(I).kulcs):=X(I)
  Ciklus vége
Eljárás vége.
```

Itt a speciális feltétel miatt meg kell változtatnunk a példasorozatot: 3, 2, 5, 1, 4 (most csak a kulcsmező értékét tároljuk). A feladat másik specialitása miatt pedig két sorozatot kell adnunk, a bemenetet és az eredményt. Az eredmény még nem kitöltött tagjait jelöli.

Bemenet: 3, 2, 5, 1, 4

Kimenet:
I=1 $\Rightarrow$ $\otimes$, $\otimes$, $\otimes$, $\otimes$, $\otimes$
I=2 $\Rightarrow$ $\otimes$, $\otimes$, 3, $\otimes$, $\otimes$
I=3 $\Rightarrow$ $\otimes$, 2, 3, $\otimes$, $\otimes$
I=4 $\Rightarrow$ $\otimes$, 2, 3, $\otimes$, 5
I=5 $\Rightarrow$ 1, 2, 3, $\otimes$, 5
I=6 $\Rightarrow$ 1, 2, 3, 4, 5

Helyfoglalás: **2*N**

Hasonlítások száma: 0

Mozgatások száma: N

Kulcsmezőindexelés: N

5. Számlálva szétosztó rendezés

Egy kicsit kevesebbet követel előfeltételként a következő rendezés: itt az elemek kulcsmezője lehet az [1,N]-nél szélesebb intervallumban is és nem szükséges feltétel a kulcsok különbözősége.

Itt első lépésként számoljuk le minden lehetséges kulcsértékre, hogy hány ilyen értékű elem van! Mivel a kulcsértékekkel indexelhetünk, ezért a számlálás egyetlen indexeléssel elvégezhető minden elemre.

Másodjára minden lehetséges kulcsértékhez határozzuk meg a nála kisebb vagy egyenlő kulcsú rekordok számát!

Harmadik lépésként minden elemet közvetlenül a helyére helyezhetünk a fenti információ alapján.

```
Rendezés(N,X):
  DB(1..M):=(0,...,0)
  Ciklus I=1-től N-ig
    DB(X(I).kulcs):=DB(X(I).kulcs)+1
  Ciklus vége
  Első(1):=1
  Ciklus J=2-től M-ig
    Első(J):=Első(J-1)+DB(J)
  Ciklus vége
  Ciklus I=1-től N-ig
    Y(Első(X(I).kulcs)):=X(I)
    Első(X(I).kulcs):=Első(X(I).kulcs)+1
  Ciklus vége
Eljárás vége.
```

Itt a speciális rendezés miatt újabb példasorozatot vizsgálunk: 5, 3, 5, 1, 7. A bemenet mellett először a DB vektor értékeit közöljük, majd az eredményt.

Bemenet:	DB:
5, 3, 5, 1, 7	I=1 ⇒ 0, 0, 0, 0, 0, 0, 0 I=2 ⇒ 0, 0, 0, 0, 1, 0, 0 I=3 ⇒ 0, 0, 1, 0, 1, 0, 0 I=4 ⇒ 0, 0, 1, 0, 2, 0, 0 I=5 ⇒ 1, 0, 1, 0, 2, 0, 0 I=6 ⇒ 1, 0, 1, 0, 2, 0, 1

A második ciklusban:

DB:		Kimenet:
J=2 ⇒ 1, 0, 1, 0, 2, 0, 1 J=3 ⇒ 1, 1, 1, 0, 2, 0, 1 J=4 ⇒ 1, 1, 2, 0, 2, 0, 1 J=5 ⇒ 1, 1, 2, 2, 2, 0, 1 J=6 ⇒ 1, 1, 2, 2, 4, 0, 1 J=7 ⇒ 1, 1, 2, 2, 4, 4, 1 1, 1, 2, 2, 4, 4, 5		I=1 ⇒ ⊗, ⊗, ⊗, ⊗, ⊗ I=2 ⇒ ⊗, ⊗, ⊗, 5, ⊗ I=3 ⇒ ⊗, 3, ⊗, 5, ⊗ I=4 ⇒ ⊗, 3, 5, 5, ⊗ I=5 ⇒ 1, 3, 5, 5, ⊗ I=6 ⇒ 1, 3, 5, 5, 7

Helyfoglalás: 2*N+M*ε   **Hasonlítások száma**: 0

Mozgatások száma: N   **Kulcsmező-indexelés**: 5*N

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

1. feladat

Egy futóversenyen a versenyzőket (N db) rajtszám szerint (a rajtszám 1-től N-ig fut) percenként indítják. A célba érkezési listán időrendben megadják, hogy melyik rajtszámú versenyző mikor érkezett célba.

Írj programot (VERSENY.PAS, VERSENY.C,…), amely beolvassa a versenyzők számát (1≤N≤100), a célba érkezett versenyzők számát (1≤M≤N), majd M darab versenyző sorszámot (1≤sorszám≤N) és célba érkezési időt (0 és 10000 közötti egész számok, növekvő sorrendben). A program ezek alapján készítse el az eredménylistát, valamint adja meg, hogy kik nem érkeztek célba.

<u>Példa:</u>

```
Bemenet:                Kimenet:

N=6, M=4                1. helyezett: 5
sorszám: 2 idő: 10      2. helyezett: 2 4
sorszám: 4 idő: 12      4. helyezett: 1
sorszám: 5 idő: 12
sorszám: 1 idő: 15      Nem érkezett célba: 3 6
```

Jelölje `v(i)=igaz`, ha az i-edik versenyző célba ért! Kezdetben a `v` vektor minden elem legyen hamis, amit célba érésnél állítunk igazra! A célba ért versenyzők futási idejét a beérkezési időből és a rajtszámból számolhatjuk ki.

Ezután rendezni kell futási idő szerint, majd jöhet a kiírás, de figyelni kell a holtversenyre is!

```
Verseny(n,m):
  v(i):=(hamis, ..., hamis)
  Ciklus i=1-től m-ig
    Be: t(i).sor; v(t(i).sor):=igaz
    Be: j; t(i).idő:=j-t(i).sor
  Ciklus vége
  Ciklus i=1-től m-1-ig
    min:=i
    Ciklus j=i+1-től m-ig
      Ha t(j).idő<t(min).idő akkor min:=j
    Ciklus vége
    s:=t(i); t(i):=t(min); t(min):=s
  Ciklus vége
```

```
  j:=t(1).idő-1
  Ciklus i=1-től m-ig
    Ha j<t(i).idő akkor Ki: i,'. helyezett:'
    Ki: t(i).sor; j:=t(i).idő
  Ciklus vége
  Ha m<n akkor Ki: 'Nem érkezett célba:'
               Ciklus i=1-től n-ig
                 Ha nem v(i) akkor Ki: i
               Ciklus vége
Eljárás vége.
```

2. feladat

Egy pontozásos versenyen N (1≤N≤100) résztvevő indul. A versenyzőket M (3≤M≤10) pontozó pontozza. A pontozók azonban befolyásolhatnák a verseny végeredményét (a saját országbeli versenyzőknek nagyobb, a más országbeli versenyzőknek pedig kisebb pontot adva, ezért a verseny szervezői úgy döntöttek, hogy az M pontszám közül a legkisebbet és a legnagyobbat egy versenyzőnél sem veszik figyelembe, azaz mindenkit M-2 pontszám összegével értékelnek.

Készíts programot (`verseny.pas`, …), amely

A. kiírja minden versenyzőre, hogy mely pontozók pontszámát hagyják ki;

B. kiszámítja és kiírja minden versenyző pontszámát;

C. kiírja a verseny végeredményét (a versenyzőket pontszám szerint csökkenő sorrendben, holtverseny esetén azonos helyezési számmal)!

Példa:

```
Bemenet:                            Kimenet:

N=4, M=4                            Kihagyott pontozók:
1. versenyző pont: 8 2 6 6          1. versenyző: 1 2
2. versenyző pont: 5 5 5 5          2. versenyző: 1 2
3. versenyző pont: 4 5 6 7          3. versenyző: 1 4
4. versenyző pont: 5 4 6 5          4. versenyző: 2 3
                                    Pontszámok:
                                    1. versenyző: 12 pont
                                    2. versenyző: 10 pont
                                    3. versenyző: 11 pont
                                    4. versenyző: 10 pont
                                    Sorrend:
                                    1. helyezett: 1. versenyző 12 pont
                                    2. helyezett: 3. versenyző 11 pont
                                    3. helyezett: 2. versenyző 10 pont
                                    3. helyezett: 4. versenyző 10 pont
```

A. NEM RENDEZÉS, de a rendezendők kiszámításához szükséges;

B. NEM RENDEZÉS, de a rendezendők kiszámításához szükséges;

C. ez már egyszerű rendezés (bármely módszer jó lenne), holtverseny esetén azonos helyezési számkiírásával.

Figyelni kell arra, hogy rendezésnél az elemek elmozdulnak a helyükről, azaz a sorszámuk nem azonos az indexükkel!

```
Verseny:
  Ciklus i=1-től n-ig
    Ha pont(i,1)≥pont(i,2) akkor max(i):=1; min(i):=2
                        különben max(i):=2; min(i):=1
    Ciklus j=3-tól m-ig
      Ha pont(i,j)>pont(i,max(i)) akkor max(i):=j
      Ha pont(i,j)<pont(i,min(i)) akkor min(i):=j
    Ciklus vége
  Ciklus vége
  Ciklus i=1-től n-ig
    op(i):=-pont(i,max(i))-pont(i,min(i))
    Ciklus j=1-től m-ig
      op(i):=op(i)+pont(i,j)
    Ciklus vége
    s(i):=i
  Ciklus vége
  Ciklus i=1-től n-1-ig
     Ciklus j=i+1-től n-ig
       Ha op(i)<op(j) akkor Csere(op(i),op(j)
                            Csere(s(i),s(j))
    Ciklus vége
  Ciklus vége
  x:=1
  Ciklus i=1-től n-ig
    Ki: x,'. helyezett:',s(i),'. versenyző ',op(i),' pont'
    Ha op(i)>op(i+1) akkor x:=i+1
  Ciklus vége
Eljárás vége.
```

3. feladat

Az Olimpiai Játékokon M ország vesz részt (1≤M≤100), N versenyszámban versenyeznek a résztvevők (1≤N≤1000). Minden versenyszámban 1 arany-, 1 ezüst-, valamint 1 vagy 2 bronzérmet adnak ki (kieséses versenyek esetén a döntőbe jutásért küzdők közül mindkét vesztes bronzérmet kap). Az országokat 1 és M közötti sorszámukkal azonosítjuk.

Készíts programot (OLIMPIA.PAS, OLIMPIA.C, …), amely az eredmények ismeretében előállítja az olimpia éremtáblázatát! Az éremtáblázat aranyérmek száma szerint csökkenő sorrendű legyen. Azonos aranyérem szám esetén a több ezüst-, azonos ezüstérem szám esetén a több bronzérem döntsön! Ha mindhárom éremből ugyanannyi van, akkor a kisebb sorszámú legyen elöl!

Példa:

```
Bemenet:                                Kimenet:

M=5, N=3                                2. ország: 1 A, 2 E, 2 B
1. szám: A – 2, E – 3, B – 2            1. ország: 1 A
2. szám: A – 1, E – 2, B – 3            5. ország: 1 A
3. szám: A – 5, E – 2, B – 2,3          3. ország: 1 E, 2 B
```

Számoljuk ki az egyes országok éremszámait, majd az éremszám szerint rendezzük csökkenő sorrendbe őket!

```
olimpia(N,s,M,érem):
  érem(i,j):=(0,…,0)
  Ciklus i=1-től M-ig
    érem(i,0):=i
  Ciklus vége
  Ciklus i=1-től N-ig
    érem(s(i,1),1):=érem(s(i,1),1)+1
    érem(s(i,2),2):=érem(s(i,2),2)+1
    érem(s(i,3),3):=érem(s(i,3),3)+1
    Ha s(i,4)>0 akkor érem(s(i,4),3):=érem(s(i,4),3)+1
  Ciklus vége
  Ciklus i=1-től M-1-ig
    max:=i
    Ciklus j=i+1-től M-ig
      Ha nagyobb(j,max) akkor max:=j
    Ciklus vége
    Csere(érem(max),érem(i))
  Ciklus vége
Eljárás vége.
```

```
nagyobb(i,j):
  nagy:=hamis
  Ha érem(i,1)>érem(j,1) akkor nagy:=igaz
  különben ha érem(i,1)=érem(j,1) akkor
              Ha érem(i,2)>érem(j,2) akkor nagy:=igaz
              különben ha érem(i,2)=érem(j,2) akkor
                          Ha érem(i,3)>érem(j,3)
                             akkor nagy:=igaz
  nagyobb:=nagy
Függvény vége.
```

4. feladat

Egy piacon N egymást követő napon árulnak almát. Arra vagyunk kíváncsiak minden napon, hogy az addigi napok közül mely K napon lehetett a legolcsóbban almát venni!

Írj programot (ALMAK.PAS,ALMAK.C,ALMAK.BAS), amely beolvassa a napok N számát (1≤N≤100), majd K értékét (1≤K≤10), majd ezután egyesével olvassa az egyes napokon az alma árát, s minden beolvasás után kiírja azon K nap sorszámát növekvő sorrendben, amelyeken a legolcsóbban lehetett almát venni! Amíg nem volt K nap, addig a program ne írjon ki semmit!

Példa:

```
N=10, M=4

1. Be: 80
2. Be: 70
3. Be: 75
4. Be: 90      Ki: 1 2 3 4
5. Be: 100     Ki: 1 2 3 4
6. Be: 60      Ki: 1 2 3 6
7. Be: 77      Ki: 2 3 6 7
8. Be: 80      Ki: 2 3 6 7
9. Be: 77      Ki: 2 3 6 7, de a 2 3 6 9 is jó megoldás
10. Be: 90     Ki: 2 3 6 7, de a 2 3 6 9 is jó megoldás
```

Ha még nem volt K nap, akkor sorba rendezve gyűjtjük az árakat és a napsorszámokat (`T(i).ár`, illetve `t(i).sorszám`). Ha már volt K nap, akkor ha a K-adik legolcsóbbnál olcsóbb az alma, akkor beillesztjük a tároltak közé.

```
Almák(n,k):
  üres(h)
  Ciklus i=1-től n-ig
    Be: ár
    Ha i≤k akkor j:=i-1
                 Ciklus amíg j>0 és ár<t(j).ár
                   t(j+1):=t(j); j:=j-1
                 Ciklus vége
                 t(j+1).ár:=ár; t(j+1).sorszám:=i
                 Halmazba(h,i)
    különben ha ár<t(k).ár
           akkor j:=k-1; halmazból(h,t(k).sorszám)
                 Ciklus amíg j>0 és ár<t(j).ár
                   t(j+1):=t(j); j:=j-1
                 Ciklus vége
                 t(j+1).ár:=ár; t(j+1).sorszám:=i
                 Halmazba(h,i)
```

```
    Ha i≥k akkor Ciklus j=1-től i-ig
                   Ha eleme(j,h) akkor Ki: j
                 Ciklus vége
  Ciklus vége
Eljárás vége.
```

5. feladat

Egy piacon M árus N egymást követő napon árul almát. Az árusok különböző napokon kezdenek almát árulni, s ettől kezdve, amíg más árat nem adnak, ugyanazon az áron adják az almát. Arra vagyunk kíváncsiak minden napon, hogy aznap mely K árustól lehet a legolcsóbban almát venni!

Írj programot (ALMA.PAS, ALMA.C), amely megadja minden napra, hogy aznap mely K árustól lehet a legolcsóbban almát venni, ha van aznap egyáltalán K árus!

Az ALMA.BE szöveges állomány első sorában az árusok M száma (1≤M≤100), a napok N száma (1≤N≤1000), és a K értéke (1≤K≤M) van egy-egy szóközzel elválasztva. Az állomány további sorai egy-egy árus érkezését írják le. Mindegyik sorban három szám van egy-egy szóközzel elválasztva: az érkezés napja (1≤nap≤N, a sorok eszerint növekvő sorrendben jönnek), az árus sorszáma (1≤sorszám≤M) és az általa árult alma ára attól a naptól kezdve (0<ár≤1000).

Az ALMA.KI szöveges állományba pontosan N sort kell írni, az i-edik sorba az i-edik napon legolcsóbb K árus **sorszámát**, növekvő sorrendben, egy-egy szóközzel elválasztva. Ha aznap nem árult almát K árus, akkor a sorba egyetlen 0-t kell kiírni.

Példa:

ALMA.BE	ALMA.KI	magyarázat
6 8 3	0	az első napon nincs 3 árus
1 1 100	2 3 6	
1 2 90	1 3 6	
2 6 80	1 3 6	a negyedik napon nem jött újabb árus
2 3 70	1 3 6	az ötödik napon nem jött újabb árus
2 5 120	1 3 6	a hatodik napon nem jött újabb árus
3 1 60	3 4 6	
3 4 100	3 4 6	a nyolcadik napon nem jött újabb árus
7 1 120		
7 4 75		

Naponta figyeljük a K legolcsóbb árust. Az input file-ban előreolvasunk egy rekordot, a következő árus érkezését. Azon a napon van teendőnk, amely napon jött árus (több is lehet egymás után).

Ekkor meg kell néznünk, hogy árult-e már korábban. Ha szerepelt, akkor el kell távolítani: ki kell szedni a legolcsóbb K halmazából (`h`), valamint az árak szerint rendezett tömbből is ki kell hagyni! Ha már több, mint K árusunk van, akkor a K+1-ediket kell bevennünk a legolcsóbbak közé!

Ezután az új árat be kell szúrnunk a rendezett tömbbe, s ha a legolcsóbb K közé kerül, akkor a h halmazba is fel kell venni. Ha a halmazban már volt K elem, akkor a legnagyobbat ki kell hagyni.

Minden nap végén meg kell nézni, hogy van-e K árus, s ha igen, akkor a h halmaz elemeit kell kiírni!

```
Almák(n,m,k):
  Olvas(BeF,nap,sorszám,ár); db:=0; üres(h)
  Ciklus ns=1-től n-ig
    Ciklus amíg nap=ns
      j:=1
      Ciklus amíg j≤db és t(j).sorszám≠sorszám
        j:=j+1
      Ciklus vége
      Ha j≤db akkor
        Ha j≤k akkor Halmazból(h,t(j).sorszám)
                     Ha db>k akkor Halmazba(h,t(k+1).sorszám)
        Ciklus amíg j≤db
          t(j):=t(j+1); j:=j+1
        Ciklus vége
        db:=db-1
      Elágazás vége
      j:=db
      Ciklus amíg j>0 és ár<t(j).ár
        t(j+1):=t(j); j:=j-1
      Ciklus vége
      t(j+1).ár:=ár; t(j+1).sorszám:=sorszám; db:=db+1
      Ha j+1≤k akkor Halmazba(h,sorszám)
                     Ha db>k akkor Halmazból(h,t(k+1).sorszám)
      Ha nem vége?(BeF) akkor Olvas(BeF,nap,sorszám,ár)
                     különben nap:=n+1
    Ciklus vége
    Ha db≥k akkor Ciklus i=1-től m-ig
                    Ha eleme(i,h) akkor Ír(KiF,i)
                  Ciklus vége
         különben Ír(KiF,0)
  Ciklus vége
Eljárás vége.
```