Algoritmusok és adatstruktúrák

Vizsgaminta (v_04.pdf) feladatok megoldásai

1. ELSŐ RÉSZ

Készítsen adatszerkezeti táblázatot és algoritmust (struktúra diagram vagy pszeudokód) az alábbi feladatokra!

1.1. Első feladat

Input adatként adott egy N sorból (N<=10) és M oszlopból (M<=20) álló, egész számokat tartalmazó számtáblázat, ahol N és M értéke input adat. Kérjük be az adatokat és írjuk ki minden sorra, a benne található elemek alapján a megfelelő szöveget! (3p)

- Növő: ha az elemek (szigorúan véve) nőnek;
- Csökkenő: ha az elemek (szigorúan véve) csökkennek;
- Azonos: ha az elemek azonosak;
- Egyéb: ha a fentiek közül egyik sem teljesül.

1.2. Második feladat

Ha kitöröljük a táblázat „Egyéb" tulajdonságú sorait. (2p)

1.3. Megoldás

A két feladat összefügg, ezért a megoldásokat egyben adjuk meg. A táblázat (A) és a sorok száma (N) azért output jellegű is, mert ha töröltünk sort, akkor ezek megváltoznak. Az egyéb sorok jelzésére használt logikai tömb (EGYEB) azért kell, hogy tudjuk, melyik sorokat kell kitörölni. Amikor sort törlünk a táblázatból, akkor ebből a tömbből is ki kell törölnünk a megfelelő elemet.

A sorok minősítésére logikai változókat használunk, amelyeket minden sor vizsgálatakor igaz kezdőértékűre állítunk. Megvizsgálva az adott sor elemeit, amint találunk egy olyan szomszédos elempárt, ami megsérti az adott tulajdonságot, akkor hamisra állítjuk a megfelelő változó értékét. Az egyéb tulajdonságot nem figyeljük, mert ha a másik esetek egyike sem teljesül, akkor lesz ilyen minősítésű az adott sor.

A sorok törlésére használt legkülső ciklus while ciklus (és nem for), mert a sorok száma megváltozik, ha sort törlünk. A while feltétele ugyanis mindig az aktuális N értékét használja a hasonlításhoz, míg a for ciklus, a ciklus kezdetén meghatározott N értékhez hasonlít. Egy sor törlését az utána jövő sorok „előrébb léptetésével" végezzük, hogy a megmaradó sorok sorrendje ne változzon meg. A megoldás végén lévő táblázatkiírás nem volt része a feladatnak.

Funkció	Azonosító	Típus	Jelleg
A táblázat	A	Kétdimenziós egész tömb[10,20]	I,M,O
Sorok száma	N	Egész	I,O
Oszlopok száma	M	Egész	I
Növő elemek	NOVO	Logikai	M,O
Csökkenő elemek	CSOKKENO	Logikai	M,O
Azonos elemek	AZONOS	Logikai	M,O
Az egyéb sorok	EGYEB	Egydimenziós logikai tömb[10]	M,O
Segédváltozók	I,J,K	Egész	M

```
/* Adatok bekérése */
Be:N,M
for I<-1,N
  for J<-1,M
    Be:A[I,J]

/* Sorok vizsgálata és minősítése */
for I<-1,N
  NOVO<-CSOKKENO<-AZONOS<-igaz
  for J<-1,M-1
    if A[I,J]>=A[I,J+1]
      NOVO<-hamis
    if A[I,J]<=A[I,J+1]
      CSOKKENO<-hamis
    if A[I,J]<>A[I,J+1]
      AZONOS<-hamis
  EGYEB[I]<-hamis
  if NOVO
    Ki:I,". sor elemei növekvőek"
  else if CSOKKENO
    Ki:I,". sor elemei csökkenőek"
  else if AZONOS
    Ki:I,". sor elemei azonosak"
  else
    Ki:I,". sor elemei egyéb minősítésűek"
    EGYEB[I]<-igaz

/* Az egyéb minősítésű sorok törlése */
I<-1
while I<=N
  if EGYEB[I]
    /* Az I. sor törlése a mögöttes sorok előrébb másolásával */
    for J<-I+1,N
      for K<-1,M
        A[J-1,K]<-A[J,K]
      /* Az EGYEB tömbből is töröljük az I. elemet */
      EGYEB[J-1]<-EGYEB[J]
    /* A sorok száma csökken eggyel */
    N<-N-1
  else
    /* Vehetjük a következő sort */
    I<-I+1

/* Az eredmény táblázat kiírása */
for I<-1,N
  for J<-1,M
    Ki:A[I,J]
```

2. MÁSODIK RÉSZ

Tervezzen adatstruktúrát és készítsen feladatonként egy-egy teljesen paraméterezett szubrutint (adatszerkezeti táblázat és algoritmus (struktúradiagram vagy pszeudokód)) az alábbi feladatokra!

Síkbeli, egész koordinátájú pontokból halmazokat képezünk, ahol egy halmazba tetszőleges számú (természetesen különböző) pont tartozhat. A koordináták lehetséges értékei a [0,100] intervallumba esnek.

2.1. Első feladat

Annak eldöntése, hogy egy adott pont beleesik-e egy adott halmazba vagy sem. (1p)

A feladatok itt is összefüggnek, ezért a közös konstans és típus deklarációkat csak itt, az első feladat megoldásánál adjuk meg.

A megoldásokban egy ponthalmazt egy kétdimenziós 101*101-es logikai tömb segítségével tárolunk. Minden koordináta eggyel nagyobb indexnek felel meg, hogy ne kelljen 0-tól indexelnünk. Tehát pl. a (0, 0) pont megfelelője a mátrix [1, 1] eleme, a (10, 25) pontnak a [11, 26] elem, a (100,100) pontnak pedig a mátrix [101,101] indexű eleme felel meg. Egy adott mátrixelem akkor és csak akkor igaz, ha a hozzá tartozó pont a halmazban van.

Konstans
KOORDMAX 100
INDEXMAX KOORDMAX+1

Típus
PONT Rekord
    X Egész
    Y Egész
PONT_HALMAZ Kétdimenziós logikai tömb[INDEXMAX, INDEXMAX]

Funkció	Azonosító	Típus	Jelleg
A ponthalmaz	PH	PONT_HALMAZ	I
A pont	P	Egész	I

```
TARTALMAZAS(PH,P)
return PH[P.X+1,P.Y+1]
```

2.2. Második feladat

Egy halmaz elemeinek kiírása egy adott nevű szövegfájlba. (2p)

A pontokat növekvő X, azon belül növekvő Y koordináták szerint írjuk a fájlba. Minden sorba egy pont kerül az X és Y koordinátájával. A megoldásban „előállítunk" egy pontot is, hogy a tartalmazás (TARTALMAZAS) függvényt használhassuk, de megjegyezzük, hogy enélkül is megoldható a feladat (aminek elkészítését az olvasóra bízzuk).

Funkció	Azonosító	Típus	Jelleg
A ponthalmaz	PH	PONT_HALMAZ	I
A szövegfájl neve	FNEV	Sztring	I
A fájlváltozó	F	Szövegfájl	M
Egy pont	P	PONT	M
Segédváltozók	I,J	Egész	M

```
HALMAZ_KIIR(PH,FNEV)
NYIT(F,FNEV,"O")
for I<-1,INDEXMAX
  for J<-1,INDEXMAX
    P.X<-I-1
    P.Y<-J-1
    if TARTALMAZAS(PH,P)
      Ki F:P.X,P.Y
ZAR(F)
```

2.3. Harmadik feladat

Két halmaz metszete. (2p)

A megoldásban nem használtuk a tartalmazás vizsgálatot (TARTALMAZAS), de megjegyezzük, hogy azzal is megoldható a feladat (aminek elkészítését az olvasóra bízzuk).

Funkció	Azonosító	Típus	Jelleg
Az egyik halmaz	A	PONT_HALMAZ	I
A másik halmaz	B	PONT_HALMAZ	I
Az eredmény	C	PONT_HALMAZ	M,O
Segédváltozók	I,J	Egész	M

```
METSZET(A,B,C)
for I<-1,INDEXMAX
  for J<-1,INDEXMAX
    C[I,J]<-A[I,J] AND B[I,J]
```