

PROGRAMOZÁSMÓDSZERTAN

3. ELŐADÁS'2005

(VÁZLAT)

1. A TÖMB TÍPUSKONSTRUKCIÓ

1.1. A tömb algebrai specifikációja

Típus **Tömb**(Index,Elem) :

Asszociált műveletek:

Létrehoz:Tömb

Lerombol (Tömb)

ElemSzám(Tömb) :Egész

ElemÉrték(Tömb, Index) :Elem \cup {NemDef}

ElemMódosít(Tömb, Index, Elem) :Tömb

Axiómák:

- t : Tömb(Index,Elem) = tetszőleges tömbtípusú objektum
- i, j : Index = tetszőleges indextípusú objektum
- e : Elem = tetszőleges elemtípusú objektum

1° A Létrehozás után a tömb minden eleme olyan értékű, amilyent a bázistípus Létrehoz_e művelete létrehoz¹. Pontosán annyi eleme lesz, amennyit az Index-típus meghatároz.

$ElemÉrték(Létrehoz, i) = Létrehoz_e \wedge$
 $ElemSzám(Létrehoz) = Számosság(Index)$

Állítás:

Egy elemének módosítása nem változtatja meg a tömb hosszát, azaz

$ElemSzám(ElemMódosít(t, i, e)) = ElemSzám(t)$

Biz.:

1° $\Rightarrow Számosság(Index) = Konstans$

$\Rightarrow t' = ElemMódosít(t, i, e) \wedge ElemSzám(t') = Számosság(Index)$

$\Rightarrow ElemSzám(t') = ElemSzám(t)$

2° Lerombolás után nincs értelme a műveletek (a Létrehoz kivételével) alkalmazásának. (Ez egy nyilvánvalóságot megfogalmazó axióma, l. a szignatúrákat!)

$ElemSzám(Lerombol(t)) = NemDef \wedge \dots$

3° A tömb elemének az az értéke, amit az utolsó rávonatkozó ElemMódosít művelet adott neki.

$ElemÉrték(ElemMódosít(t, i, e), i) = e$

¹ Esetleg: NemDef, ha az Elem típusban nem létezik a konstrukciós Létrehoz_e művelet.

4° Az elemek értéküket módosításig megőrzik. (Ez egy szokatlan axióma, amely azt mondja meg, hogy az *ElemMódosítás* művelet mire **nincs** hatással; a későbbi sorozatfélékhez képest jellemző eltérés.)

$$i \neq j \Rightarrow \text{ElemÉrték}(\text{ElemMódosít}(t, i, e), j) = \text{ElemÉrték}(t, j)$$

Állítás:

a tömb létrehozása után az egyes elemei az első *ElemMódosít*-ig kezdőértékükkel rendelkeznek.

Biz.:

a 3° és az 4° folyamánya.

Megjegyzések:

- az egyetlen kezelési anomáliáról, az *indextúllépés*ről nem kell rendelkezni az axiómákban, hiszen ez az egyik bázistípust érintő sértés lévén már korábban –az *Index-típushoz tartozó axiómák megsértése* miatt– hibás, azaz nem definiált eredményre vezetne.
- a 2° axióma egy *szintaktikai képtelenséget* fogalmaz meg: a *Lerombol* függvény „eredményére” lehetetlen más függvényt alkalmazni, hisz nincs mire, nincs értékészlete! Ez persze nem a tömb típuskonstrukciós eszköz specifikus problémája, *minden lerombol függvényé*. Ezért nem is jeleztük külön a többi művelet értékészleténél a NemDef értéket. S ezt a további típuskonstrukciónál már nem is említjük.
- ehhez hasonlóan lehetne a több-indexes tömbökkel szembeni elvárásunkat megfogalmazni, amitől eltekintünk.

3.2. A tömb algoritmikus specifikációja

Az *eltérő paraméterezési szokások* miatt külön-külön fogalmazzuk meg az 1- és a 2-indexes tömbök, azaz a vektorok és a mátrixok típuskonstrukcióját. (A többieket specifikálni ezek alapján már gyerekjáték.)

3.2.1. A tömb exportmoduljai

Először a *vektor* exportmodulját adjuk meg:

```
ExportModul Tömb(Típus TIndex: Típus TElem):
  [Ef: TIndex: diszkrét, sőt véges típus (így van neki „Sorszám”, „Következő”
  ... függvénye)2,
  TElem típushoz asszociáltunk „Létrehoz”, „=”, „Be”, „Ki”... műveleteket,
  az Ef-Uf-ben a t-t mint e-k sorozatát tekintjük, elemeit 1-től indexelve]
  Eljárás Létrehoz (Változó t:Tömb)
    Ef: ¬∃t∈Tömb
    Uf: ∀i∈TIndex: ∃e∈TElem: t(Sorszám(i)+1)=e ∧
        Létrehoz(e).Uf
  Eljárás Lerombol (Változó t:Tömb)
    Ef: ∃t∈Tömb
    Uf: ¬∃t∈Tömb
```

² Szokásos elvárásokkal; pl. Sorszám(Min' TElem)=0.

```

Függvény ElemSzám(Konstans t:Tömb):Egész
  Ef:  $\exists t \in \text{Tömb}$ 
  Uf: ElemSzám(t) = Számosság'TIndex

Operátor ElemÉrték(Konstans t:Tömb, i:TIndex):TElem
  Másnéven t(i)
  Ef:  $\exists t \in \text{Tömb}$ 
  Uf: ElemÉrték(t, i) = t(Sorszám(i)+1)  $\wedge$ 
        t(Sorszám(i)+1) = t'(Sorszám(i)+1)

Operátor ElemMódosít(Változó t:Tömb,
                       Konstans i:TIndex, e:TElem)
  Másnéven t(i) := e
  Ef:  $\exists t \in \text{Tömb}$ 
  Uf: t'(Sorszám(i)+1) = e  $\wedge$ 
         $\forall j \neq i \in \text{TIndex}: t(\text{Sorszám}(j)+1) = t'(\text{Sorszám}(j)+1)$ 

Infix Operátor Egyenlő(Konstans t, tt:Tömb):Logikai
  Másnéven t=tt
  Ef:  $\exists t, tt \in \text{Tömb}$ 
  Uf: Egyenlő(t, tt) =
        (  $\forall i \in \text{TIndex}: t(\text{Sorszám}(i)+1) = tt(\text{Sorszám}(i)+1)$  )

Infix Operátor LegyenEgyenlő(Változó t:Tömb
                                Konstans tt:Tömb)
  Másnéven t:=tt
  Ef:  $\exists t, tt \in \text{Tömb}$ 
  Uf:  $\forall i \in \text{TIndex}: t'(\text{Sorszám}(i)+1) = tt(\text{Sorszám}(i)+1)$   $\wedge$ 
         $\forall i \in \text{TIndex}: tt(\text{Sorszám}(i)+1) = t'(\text{Sorszám}(i)+1)$ 

Operátor Be(Változó t:Tömb)
  Másnéven Be:t
  Ef:  $\exists t \in \text{Tömb} \wedge \forall i \in \text{TIndex}: \text{Be}(t(\text{Sorszám}(i)+1)) . \text{Ef}$ 
  Uf:  $\forall i \in \text{TIndex}: \text{Be}(t'(\text{Sorszám}(i)+1)) . \text{Uf}$ 

Operátor Ki(Konstans t:Tömb)
  Másnéven Ki:t
  Ef:  $\exists t \in \text{Tömb}$ 
  Uf:  $\forall i \in \text{TIndex}: \text{Ki}(t(\text{Sorszám}(i)+1)) . \text{Uf} \wedge$ 
         $\forall i \in \text{TIndex}: t(\text{Sorszám}(i)+1) = t'(\text{Sorszám}(i)+1)$ 

Függvény Hibás?(Változó t:Tömb):Logikai
  Ef:  $\exists t \in \text{Tömb}$ 
  Uf: Hibás?(t) = volt-e elkövetett s nem letesztelt hiba

Modul vége.

```

Megjegyzések:

- Jól látszik, hogy az elő- és utófeltételekben *visszanyúltunk* a *TIndex* és *TElem* típusokhoz. Az „=”, az „.Ef” és az „.Uf” a megfelelő típus megfelelő műveletére, illetőleg annak elő-utófeltételére utal.
- A feltételekben sokszor fordul elő a **t(Sorszám(i)+1)** kifejezés. Itt a **t(x)** a **t**-nek, mint *sorozatnak az x. elemét jelenti*, tehát nem keverendő össze a „hagyományos” t-

tömb elemindexeléssel. A **Sorszám(i)+1** konverzióval képezzük az i index értékét 1..**Számosság**'TIndex közé, azaz szokásos *sorozatindex*szé.

Következzék a *mátrix* exportmodulja, amelyet rövidítve (ef-uf nélkül) mellékelünk. Értelem-szerű módosításokkal megkapható a precíz változat is.

```

ExportModul Tömb (Típus TIndex1, TIndex2: Típus TElem) :
  Eljárás Létrehoz (Változó t:Tömb)
  Eljárás Lerombol (Változó t:Tömb)
  Függvény ElemSzám (Konstans t:Tömb) :Egész
  Operátor ElemÉrték (Konstans t:Tömb,
                      i:TIndex1, j:TIndex2) :TElem
  Másnéven t(i, j)
  Operátor ElemMódosít (Változó t:Tömb,
                       Konstans i:TIndex1, j:TIndex2,
                       e:TElem)
  Másnéven t(i, j) := e
  Infix Operátor Egyenlő (Konstans t, tt:Tömb) :Logikai
  Másnéven t=tt
  Infix Operátor LegyenEgyenlő (Változó t:Tömb
                                Konstans tt:Tömb)
  Másnéven t:=tt
  Operátor Be (Változó t:Tömb)
  Másnéven Be:t
  Operátor Ki (Konstans t:Tömb)
  Másnéven Ki:t
  Függvény Hibás? (Változó t:Tömb) :Logikai
Modul vége.

```

3.2.2. A tömb reprezentációs-implemetációs modulja

Kétféle ábrázolást veszünk szemügyre. Az első az ún. *folytonos ábrázolás*, amelyben a tömb elemei egymás után, szorosan foglalnak helyet, a másikban a rákövetkezési kapcsolatot *lán-colással* (mutatókkal) biztosítjuk.

Az alábbi néhány fogalmat sűrűn használjuk.

```

Operátor Számosság (Típus t) :Egész
  Másnéven Számosság' t
  [a t típus konstansainak száma]
Operátor Méret (Típus t) :Egész
  Másnéven Méret' t
  [a t típus folytonos ábrázolásához szükséges memória mérete]
Függvény Sorszám (Konstans x:Típus) :0..MaxEgész
  [az x Típus-beli belsőábrázolású kódja, azaz 0-tól induló sorszáma]

```

A folytonos ábrázolás

Egy „naiv”, értsd leegyszerűsített *memóriamodellre* építjük a leírást. Az alábbiakban összefoglaljuk azokat a fogalmakat, amelyeket alapfogalmaknak tekintünk, s ezekre építjük föl a

folytonos ábrázolást. (Az egész mögé egy, az alábbi operátorok által kezelt *byte-sorozat*ot kell képelnünk, amelyek byte-jait *természetes számokkal indexelhetünk*.)

Konstans

MaxMem:Egész(???)

Típus

MemóriaCím=0..MaxMem [C Egész,
az Egész műveleteit használni fogjuk]

Konstans

Sehova:MemóriaCím(0)

Eljárás Lefoglal(**Változó** tmut:MemóriaCím,
Konstans db:Egész)

[a memóriában db-nyi helyet foglal,
a kezdőcímet tmut-ba adja vissza,
ha nem sikerült, akkor Sehova-t]

Eljárás Felszabadít(**Változó** tmut:MemóriaCím,
Konstans db:Egész)

[a tmut-nál kezdődő db-nyi hosszú
memóriatartományt felszabadítja,
tmut-ba Sehova-t tesz]

Eljárás Értékmásolás(**Konstans** db:Egész,
c₁:MemóriaCím,
c₂:MemóriaCím)

[a c₁ címre másol db darabnyi byte-ot
a c₂ címtől kezdődően]

Először a *vektor* modulját adjuk meg:

Modul Tömb(**Típus** TIndex: **Típus** TElem):

Reprezentáció**Változó**

kcím:MemóriaCím
hiba:Logikai

Implementáció

Eljárás Létrehoz(**Változó** v:Tömb):

Ef: $kcím=Sehova^3$ [$\neg\exists v\in Tömb$]

Uf: $\forall i\in TIndex: \exists e\in TElem: v(Sorszám(i)+1)=e \wedge$
Létrehoz(e).Uf

Változó i:TIndex
e:TElem
kc:MemóriaCím

Lefoglal(kcím, Számosság' TIndex*Méret' TElem)

Ha kcím=Sehova **akkor**
hiba:=Igaz

³ L. az [inicializálásnál](#)!

```

különben
  hiba:=Hamis
  Létrehoz(e) [visszavezetés a Létrehoz(TElem)-re,
              és nem foglalkozunk az e-hibákkal!]
  kc:=ElemCím(v,Min'TIndex) [=kcím; tömb-kezdőcím]
  Ciklus i=1-től Számosság'TIndex-ig
    Értékmásolás(Méret'TElem, kc, e)
    kc:+Méret'TElem
  Ciklus vége
  Elágazás vége
Eljárás vége.
Eljárás Lerombol(Változó v:Tömb):
  Ef: kcím≠Sehova [∃v∈Tömb]
  Uf: kcím=Sehova [¬∃v∈Tömb]
  Felszabadít(kcím, Számosság'TIndex*Méret'TElem)
Eljárás vége.
Függvény ElemSzám(Változó v:Tömb):Egész
  Ef: kcím≠Sehova [∃v∈Tömb]
  Uf: ElemSzám(v)=Számosság'TIndex
  ElemSzám:=Számosság'TIndex
Függvény vége.
Operátor ElemÉrték(Konstans v:Tömb, i:TIndex):TElem
  Másnéven v(i)
  Ef: kcím≠Sehova [∃v∈Tömb]
  Uf: ElemÉrték(v, i)=v(Sorszám(i)+1) ∧
      v(Sorszám(i)+1)=v'(Sorszám(i)+1)
  Értékmásolás(Méret'TElem,
                ElemÉrték,
                ElemCím(v, i))
  [ElemÉrték:=TElem(ElemCím(v, i))]
Operátor vége.
Operátor ElemMódosít(Változó v:Tömb,
                    Konstans i:TIndex, e:TElem):
  Másnéven v(i):=e
  Ef: kcím≠Sehova [∃v∈Tömb]
  Uf: v'(Sorszám(i)+1)=e ∧
      ∀j≠i∈TIndex: v(Sorszám(j)+1)=v'(Sorszám(j)+1)
  Értékmásolás(Méret'TElem,
                ElemCím(v, i),
                ElemÉrték)
  [TElem(ElemCím(v, i)):=e]
Operátor vége.
Infix Operátor Egyenlő(Konstans v,w:Tömb):Logikai
  Másnéven v=w
  Ef: v.kcím≠Sehova ∧ w.kcím≠Sehova [∃v,w∈Tömb]
  Uf: Egyenlő(v,w)=
      ( ∀i∈TIndex: v(Sorszám(i)+1)=w(Sorszám(i)+1) )

```

```

[hf.]
Infix Operátor LegyenEgyenlő(Változó v:Tömb
                                Konstans w:Tömb)

Másnéven v:=w
Ef: v.kcím≠Sehova ∧ w.kcím≠Sehova [∃v,w∈Tömb]
Uf: ∀i∈TIndex: v'(Sorszám(i)+1)=w(Sorszám(i)+1) ∧
      ∀i∈TIndex: w(Sorszám(i)+1)=v'(Sorszám(i)+1)
[hf.]

Operátor Be(Változó v:Tömb)
Másnéven Be:v
Ef: kcím≠Sehova [∃v∈Tömb] ∧
      ∀i∈TIndex: Be(v(Sorszám(i)+1)).Ef
Uf: ∀i∈TIndex: Be(v'(Sorszám(i)+1)).Uf
[hf.]

Operátor Ki(Konstans v:Tömb)
Másnéven Ki:v
Ef: kcím≠Sehova [∃v∈Tömb]
Uf: ∀i∈TIndex: Ki(v(Sorszám(i)+1)).Uf ∧
      ∀i∈TIndex: v(Sorszám(i)+1)=v'(Sorszám(i)+1)
[hf.]

Függvény Hibás?(Változó v:Tömb):Logikai
[hf.]

[„lokális” függvények:]
Függvény ElemCím(Konstans v:Tömb, i:TIndex):MemóriaCím
  ElemCím:=kcím+RelatívCím(i)
Függvény vége.
Függvény RelatívCím(Konstans i:TIndex):Egész
  RelatívCím:=Sorszám(i)*Méret'TElem
Függvény vége.

Inicializálás
kcím:=Sehova; hiba:=Hamis
Modul vége.

```

A mátrix modulja, rövidítve:

```

Modul Tömb(Típus TIndex1,TIndex2: Típus TElem):
Reprezentáció
Változó
  kcím:MemóriaCím
  hiba:Logikai
Implementáció
Eljárás Létrehoz(Változó m:Tömb):
  Lefoglal(kcím,
           Számosság'TIndex1*
           Számosság'TIndex2*
           Méret'TElem)

```

```

Ha kcím=Sehova akkor
    hiba:=Igaz
különben
    hiba:=Hamis
    [hf.]
Elágazás vége
Eljárás vége.
Eljárás Lerombol (Változó m:Tömb) :
    Felszabadít (kcím, Számosság' TIndex1*
                Számosság' TIndex2*
                Méret' Elem)

Eljárás vége.
Függvény ElemSzám (Változó v:Tömb) :Egész
    ElemSzám:=Számosság' TIndex1*Számosság' TIndex2
Függvény vége.
Operátor ElemÉrték (Konstans m:Tömb,
                    i1:TIndex1, i2:TIndex2) :TElem
    Értékmásolás (Méret' TElem,
                  ElemÉrték,
                  ElemCím(m, i1, i2))
    [ElemÉrték:=TElem (ElemCím(m, i1, i2))]
Operátor vége.
Operátor ElemMódosít (Változó m:Tömb,
                       Konstans i1:TIndex1, i2:TIndex2,
                       e:TElem) :
    Értékmásolás (Méret' TElem,
                  ElemCím(m, i1, i2),
                  ElemÉrték)
    [TElem (ElemCím(m, i1, i2)) :=e]
Operátor vége.
...
[„lokális” függvények:]
Függvény ElemCím (Konstans m:Tömb,
                    i1:TIndex1,
                    i2:TIndex2) :MemóriaCím
    ElemCím:=kcím+RelatívCím(i1, i2)
Függvény vége.
Függvény RelatívCím (Konstans i1:TIndex1,
                       i2:TIndex2) :Egész
    RelatívCím:=(Sorszám(i1)*Számosság' TIndex1+
                Sorszám(i2))*Méret' TElem
Függvény vége.
Inicializálás
    kcím:=Sehova; hiba:=Hamis
Modul vége.

```

Továbbiakban, ha *folytonos ábrázolás*ról beszélünk, mindig az itt definiált *tömbös ábrázolás*-ra gondolunk (nem a „naiv” memóriamodellre).

A láncolt ábrázolás

Most is az ábrázolás alapmodelljét tisztázzuk először:

```

Típus
  MemóriaCím=Típ' Mutató [alapfogalom: minden típushoz a megfelelő]

Konstans
  Sehova: MemóriaCím(???)

Függvény Típ(Konstans m:MemóriaCím):Típ
  [konstrukciós függvény, amely az m-től kezdődően
  „egybefüggően” jelenti az adott típusú adatot]

Operátor Típ(Változó m:MemóriaCím, Konstans e:Típ)
  Másként Típ(m) :=e
  [értékadás operátor, amely az m-től kezdődően
  az adott típusú adatba az e értéket másolja]

Eljárás Lefoglal(Változó tmut:Típ' Mutató)
  [a memóriában „Típ”-nyi helyet foglal,
  és Típ-nek megfelelő kezdőértékre állít,
  a kezdőcímet tmut-ba adja vissza,
  ha nem sikerült, akkor Sehova-t]

Eljárás Lefoglal(Változó tmut:Típ' Mutató,
  Konstans kért:Típ)
  [a memóriában „Típ”-nyi helyet foglal,
  majd a kért kezdőértékkel föltölti,
  a kezdőcímet tmut-ba adja vissza,
  ha nem sikerült, akkor Sehova-t]

Eljárás Felszabadít(Változó tmut:Típ' Mutató)
  [a tmut-nál kezdődő „Típ”-nyi hosszú
  memóriatartományt felszabadítja,
  tmut-ba Sehova-t tesz]

```

A *vektor* modulja:

```

Modul Tömb(Típus TIndex: Típus TElem):
  Reprezentáció
  Típus
    TVektorElem=Rekord(ért:TElem, köv:TVektorElem' Mutató)
    [„lokális” típus!]
  Változó
    kcím:TVektorElem' Mutató
    hiba:Logikai
  Implementáció
  Eljárás Létrehoz(Változó v:Tömb):
    Változó
      i:Egész
      hol:TVektorElem' Mutató

```

```

Lefoglal(kcím) [figyelem: ha létre tud jönni, akkor kap kezdőértéket!]
Ha kcím<>Sehova akkor
    hiba:=Hamis
    hol:=kcím [az első elem létrejött, az ő címe a tömb kezdőcíme]
    i:=1
    Ciklus amíg i<Számosság'TIndex és nem hiba
        Lefoglal(TVektorElem(hol).köv) [figyelem:kap kezdőértéket!]
        hol:=TVektorElem(hol).köv
        hiba:=hol=Sehova [nincs több hely]
    Ciklus vége [a további elemek is létrejöttek]
    Ha hiba akkor
        [hf.: kcím-től kezdve felszabadítjuk a lefoglalt helyet]
    különben
        TVektorElem(hol).köv:=Sehova [utolsó után nincs több elem]
        [ez elhagyható, mert automatikusan Sehova címmel jött létre4]
    Elágazás vége
különben
    hiba:=Igaz
    Elágazás vége
Eljárás vége.
Eljárás Lerombol(Változó v:Tömb):
    Változó
        i:TIndex
        hol:TVektorElem'Mutató

    hol:=TVektorElem(kcím).köv [a törlendő utánira mutat]
    Ciklus i=Min'TIndex-től Max'TIndex-1-ig
        Felszabadít(kcím) [az akt. elsőt töröltük]
        kcím:=hol [a törlendőre állítás]
        hol:=TVektorElem(kcím).köv [a törlendő utánira mutat]
    Ciklus vége
    Felszabadít(kcím) [az utolsót töröltük]
    [kcím=Sehova!]
Eljárás vége.
Operátor ElemÉrték(Konstans v:Tömb, i:TIndex):TElem
    ElemÉrték:=TElem(ElemCím(v,i))
Operátor vége.
Operátor ElemMódosít(Változó v:Tömb,
        Konstans i:TIndex, e:TElem):
    TElem(ElemCím(v,i)):=e
Operátor vége.
...

```

⁴ Ez sem igaz a Pascal esetében!

```

[„lokális” függvény:]
Függvény ElemCím(Konstans v:Tömb,
                    i:TIndex):VektorElem'Mutató

Változó
    j:Egész
    hol:TVektorElem'Mutató

    hol:=kcím
Ciklus j=1[!]-től Sorszám(i)-ig
    hol:=TVektorElem(kcím).köv
Ciklus vége
    ElemCím:=hol
Függvény vége.

Inicializálás
    kcím:=Sehova; hiba:=Hamis
Modul vége.

```

A *mátrix* modulja az alábbi. Ötlete: a mátrix elemeinek „kiegyenesítése”, azaz a mátrix-sorok egymásután fűzése (sorfolytonos láncolt ábrázolás):

```

Modul Tömb(Típus TIndex1,TIndex2: Típus TElem):
Reprezentáció
Típus
    TMátrixElem=Rekord(ért:TElem, köv:TMátrixElem'Mutató)
    [„lokális” típus!]
Változó
    kcím:TMátrixElem'Mutató
    hiba:Logikai
Implementáció
Eljárás Létrehoz(Változó m:Tömb):
Változó
    i:Egész
    hol:TMátrixElem'Mutató

    Lefoglal(kcím); hol:=kcím [az első elem létrejött]
Ciklus i=2-től Számosság'TIndex1*Számosság'TIndex2-ig
    Lefoglal(TMátrixElem(hol).köv)
    hol:=TMátrixElem(hol).köv
Ciklus vége [a további elemek is létrejöttek]
Eljárás vége.
Eljárás Lerombol(Változó m:Tömb):
Változó
    i:Egész
    hol:TMátrixElem'Mutató

```

```

hol:=TMátrixElem(kcím).köv [a törlendő utánira mutat]
Ciklus i=1-től Számosság'TIndex1*Számosság'TIndex2-1-ig
    Felszabadít(kcím) [az aktuális elsőt töröltük]
    kcím:=hol [kcím a törlendőre állítása]
    hol:=TMátrixElem(kcím).köv [a törlendő utánira mutat]
Ciklus vége
Felszabadít(kcím) [az utolsót töröltük]
[kcím=Sehova!]
Eljárás vége.
Operátor ElemÉrték(Konstans m:Tömb,
                    i1:TIndex1,
                    i2:TIndex2):TElem
    ElemÉrték:=TElem(ElemCím(v, i1, i2))
Operátor vége.
Operátor ElemMódosít(Változó m:Tömb,
                      Konstans i1:TIndex1,
                      i2:TIndex2, e:TElem):
    TElem(ElemCím(v, i1, i2)):=e
Operátor vége.
...
[„lokális” függvény:]
Függvény ElemCím(Konstans m:Tömb,
                  i1:TIndex1,
                  i2:TIndex2):TMátrixElem'Mutató

Változó
    j:Egész
    hol:TMátrixElem'Mutató
hol:=kcím
Ciklus j=1[!]-től Sorszám(i1)*Számosság'TIndex1+
                Sorszám(i2)-ig
    hol:=TMátrixElem(kcím).köv
Ciklus vége
ElemCím:=hol
Függvény vége.
Inicializálás
    kcím:=Sehova; hiba:=Hamis
Modul vége.

```

1.3. Speciális tömbök

1.3.0. A „specialitás” mibenlétéről

*Speciális tömb*nek most azt tekintjük, amely elemei rendelkeznek helycsökkentő tulajdonsággal:

1. van olyan elemérték, amely gyakoriságánál fogva „dominánsnak” tekinthető,
2. van néhány olyan eleme, amellyel a többiek generálhatók.

Általános ötlet a reprezentációra:

1. csak egyszer (vagy egyszer sem) tárolni az ismétlődőt,

2. csak a „báziselemeket” tárolni.

Két aleset az 1.-hez:

- szisztematikusan elhelyezkedő domináns elem,
- rendszertelenül, elszórtan elhelyezkedő ismétlődő elem.

1.3.1. Szisztematikusan elhelyezkedő elemek tömbökben

Ötlet: a domináns elemet egy példányban tárolva a nem ismétlődők előtt (vagy után), és a címkiszámító függvény olyan definiálása, hogy az adott indexhez a tárolt elem indexét adja.

Az ilyen fajta tömböknek érdekes alkalmazása a négyzetes szám-mátrixok ($TIndex_1 = TIndex_2$) között található:

- alsó-/felső-háromszög mátrix;
- szimmetrikus mátrix;
- tridiagonális vagy Jacobi-féle mátrix (amelyben csak a főátlóban és balról, jobbról közvetlen mellette álló elemek nem 0-k);
- Toeplitz-féle mátrix (amelynek minden, a főátlóval párhuzamos „átlói” mentén csupa azonos értéke van, a többi 0);
- Hänkel-féle mátrix (hasonlatos a Toeplitz-hez, de a mellékátló mentén vannak a nem 0 elemek).

Reprezentációhoz:

```
Típus TSziszVektIndex=0 [a domináns indexe] .. Helyszükséglet
      TSziszVekt=Tömb (TSziszVektIndex:TElem)
```

Implementációhoz:

```
Függvény Index (Konstans i, j: TIndex): TSziszVektIndex
      Index := a mátrix (i, j) elemének TSziszVekt-beli
              indexe
Függvény vége.
```

Például a MaxN*MaxN-es alsó-háromszög mátrix esetén:

```
Típus T3szögMIndex=0..MaxN*(MaxN+1) div 2
      T3szögMátrix=Tömb (T3szögMIndex:TElem)
Függvény Index (Konstans i, j: TIndex): T3szögMIndex
      Ha i>j akkor Index:=Sorszám(i)*(Sorszám(i)+1)
                      div 2 + Sorszám(j) + 1
      különben Index:=0
Függvény vége.
```

Gondolja meg a többi művelet mennyiben változik, ill. a többi speciális mátrix esetén az ábrázolás és az *index(transzformációs) függvény* hogyan definiálható.

1.3.2. Hiányosan kitöltött tömbök

Ötlet: mivel *nincs rendszer* a tárolandó elemek hollétében, ezért tárolni kell a *nem ismétlődő* elemek *indexét* (indexeit) is.

Egy ilyen „tömör” vektor ábrázolása és kezelésének alapjául szolgáló indextranszformációs függvénye:

```

Típus TTömörIndex=0 [a domináns indexe] ..Helyszükséglet
      TTömörElem=Rekord(érték:TElem,index:TIndex)
      TTömörVektor=Tömb(TTömörIndex:TTömörElem)
Függvény Index(Konstans v:TTömörVektor,
                  i:TIndex):TTömörIndex
Változó kerKi:Rekord(van:Logikai,melyik:TTömörIndex)
kerKi:=Keresés(v(k).index k∈TTömörIndex, =i?)
Ha kerKi.van akkor Index:=kerKi.melyik
      különben Index:=0
Függvény vége.

```

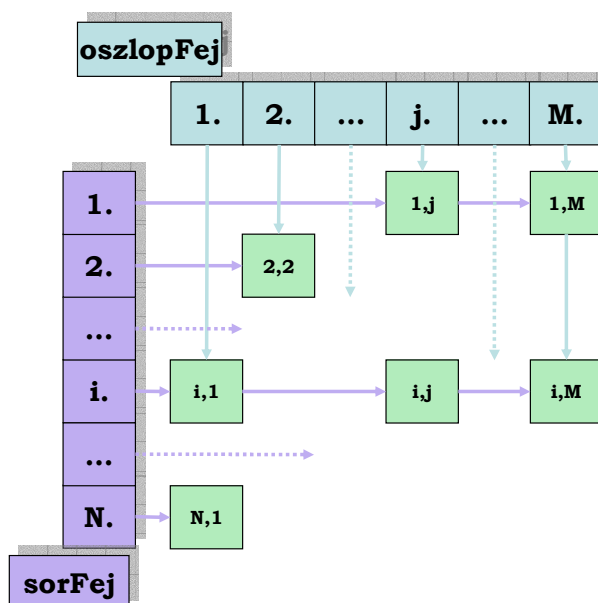
A mátrix esetében hasonlóan gondolkodhatunk:

```

Típus TTömörIndex=0..Helyszükséglet
      TTömörElem=Rekord(érték:TElem,sor,oszl:TIndex)
      TTömörMátrix=Tömb(TTömörIndex:TTömörElem)
Függvény Index(Konstans m:TTömörMátrix,
                  i,j:TIndex): TTömörIndex
Változó kerKi:Rekord(van:Logikai,melyik:TTömörIndex)
kerKi:=Keresés((m(k).sor,m(k).oszlop) k∈TTömör-
                Index, =(i,j)?)
Ha kerKi.van akkor Index:=kerKi.melyik
      különben Index:=0
Függvény vége.

```

Az elemek „elérése” így lineárisan hosszú lehet a mátrix jobb-alsó sarka felé haladva (feltéve az indexek szerinti rendezettséget). Ezen lehet segíteni az ún. *ortogonális ábrázolással* (vagyis egyfajta *láncolt* ábrázolással). Az alábbi rajz mutatja a lényegét:



Hiányos mátrix ortogonális ábrázolása

1.3.3. Generálható elemek tömbökben

Helytakarékossgot olyan tömbök esetében, amelyek elemei néhány kiemelt segítségével generálhatók nyilvánvalóan oly módon érhetünk el, hogy az ábrázolásnál csak a bázis elemekre szorítkozunk. Mivel a többi elemre az érték helyett a kiszámítás szabálya kell ismert legyen, akkor tudunk spórolni a hellyel, ha a szabályhozrendelést elemcsoportokra tudjuk „göngyöltve” kifejezni. Ez azt is jelenti, hogy valamiféle topologikus szisztematikusságnak is létezni kell. Ekkor a megvalósításhoz csak a bázis elemek tárolása és az indexek alapján működő generáló függvény szükséges.

Egy jellegzetes példa a Vandermonde-mátrix ($TIndex_1=1..N$, $TIndex_2=1..M$, $TElem=Valós$):

	1.	2.	3.	...	M.
1.:	1	1	1	...	1
2.:	a_1	a_2	a_3	...	a_M
3.:	a_1^2	a_2^2	a_3^2	...	a_M^2
...
N.:	a_1^{N-1}	a_2^{N-1}	a_3^{N-1}	...	a_M^{N-1}

Típus TVanderMátrix=**Tömb**($TIndex_2:Valós$)

[a generáló elemek a 2. sorban]

Függvény Vander(**Konstans** m:TVanderMátrix,
i:TIndex₁, j:TIndex₂):Valós

Ha i=1 **akkor** Vander:=1

különben Vander:=m(j)^{j-1}

Függvény vége.

TARTALOM

ProgramozásMódszertan 3. előadás'2005 (vázlat).....	1
1. A Tömb típuskonstrukció.....	1
1.1. A tömb algebrai specifikációja.....	1
3.2. A tömb algoritmikus specifikációja	2
3.2.1. A tömb exportmoduljai	2
3.2.2. A tömb reprezentációs-implementációs modulja.....	4
1.3. Speciális tömbök	12
1.3.1. Szisztematikusan elhelyezkedő elemek tömbökben.....	12
1.3.2. Hiányosan kitöltött tömbök.....	13
1.3.3. Generálható elemek tömbökben.....	14
Tartalom	16