Programozás
6. előadás

Tartalom

- Összetett adatszerkezetek halmozása
  - Mátrixok – vektorok vektora
  - Rekordok vektora
- Halmazzá alakítás
- Halmaz típus elemek felsorolásával
- Halmaz típus logikai vektorral

Mátrixok

Feladat:

Egy N×M-es raszterképet nagyítsunk a kétszeresére pontsokszorozással: minden régi pont helyébe 2×2 azonos színű pontot rajzolunk a nagyított képen.

→

Mátrixok

Problémák/válaszok:

- Hogyan ábrázoljunk egy képet?

  A kép rendezett pontokból áll, azaz biztosan valamilyen **sorozat**ként adható meg.

- Nehézkes lenne azonban a pontokra egy sorszámozást adni.

  Kézenfekvőbb azt megmondani, hogy egy képpont a kép hányadik **sor**ában, illetve **oszlop**ában található, azaz alkalmazzunk **dupla indexelés**t!

  A kétindexes tömböket hívjuk **mátrix**nak.

Mátrixok

Feladat:

Egy N*M-es raszterképet nagyítsunk a kétszeresére *pontsokszorozás*sal: minden régi pont helyébe 2*2 azonos színű pontot rajzolunk a nagyított képen.

Specifikáció:

- Bemenet: $N,M \in \mathbb{N}$, $K_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Kimenet: $NK_{1..2*N,1..2*M} \in \mathbb{N}^{2*N \times 2*M}$
- Előfeltétel: –
- Utófeltétel: $\forall i(1 \le i \le N)$: $\forall j(1 \le j \le M)$:

  $NK_{2*i,2*j}=K_{i,j}$ és

  $NK_{2*i-1,2*j}=K_{i,j}$ és

  $NK_{2*i,2*j-1}=K_{i,j}$ és

  $NK_{2*i-1,2*j-1}=K_{i,j}$

$:=(\mathbb{N}^{M})^{N}$

N – a sorok,

M – az oszlopok száma

Ez a **másolás** tétel egy variációja, csak egy elemből négy elem keletkezik.

Mátrixok

Algoritmus – adatleírás:

Specifikáció:

- Bemenet: $N,M \in \mathbb{N}$, $K_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Kimenet: $NK_{1..2*N,1..2*M} \in \mathbb{N}^{2 \cdot N \times 2 \cdot M}$
- Előfeltétel: –
- Utófeltétel: $\forall i(1 \le i \le N)$: $\forall j(1 \le j \le M)$:
  $NK_{2 \cdot i,2 \cdot j}=K_{i,j}$ és
  $NK_{2 \cdot i-1,2 \cdot j}=K_{i,j}$ és
  $NK_{2 \cdot i,2 \cdot j-1}=K_{i,j}$ és
  $NK_{2 \cdot i-1,2 \cdot j-1}=K_{i,j}$

Konstans

MaxN:**Egész**(???)
MaxM:**Egész**(???)

Típus

TMátrix=**Tömb**[1..MaxN,1..MaxM:**Egész**]

Változó

N,M:**Egész**
K,NK:TMátrix

Megjegyzés: programozási nyelvekben a mátrix indexelésére más jelölés is lehet. Pl. C++ esetén:

typedef int TMatrix[MaxN][MaxM].

Mátrixok

Algoritmus:

Specifikáció:

- Bemenet: $N,M \in \mathbb{N}$, $K_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Kimenet: $NK_{1..2*N,1..2*M} \in \mathbb{N}^{2 \cdot N \times 2 \cdot M}$
- Előfeltétel: –
- Utófeltétel: $\forall i(1 \leq i \leq N)$: $\forall j(1 \leq j \leq M)$:
  $NK_{2 \cdot i,2 \cdot j} = K_{i,j}$ és
  $NK_{2 \cdot i-1,2 \cdot j} = K_{i,j}$ és
  $NK_{2 \cdot i,2 \cdot j-1} = K_{i,j}$ és
  $NK_{2 \cdot i-1,2 \cdot j-1} = K_{i,j}$

Változó
i,j:**Egész**

```
i=1..N
  j=1..M
    NK[2*i,2*j]:=K[i,j]
    NK[2*i–1,2*j]:=K[i,j]
    NK[2*i,2*j–1]:=K[i,j]
    NK[2*i–1,2*j–1]:=K[i,j]
```

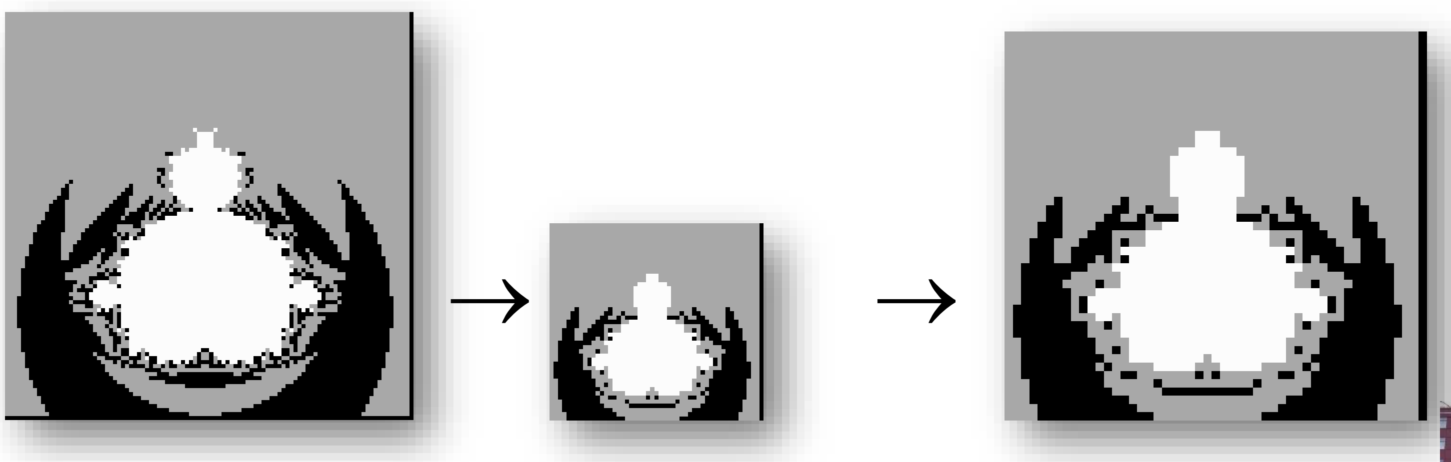
Megjegyzés: programozási nyelvekben a mátrix elemének elérésére más jelölés is lehet. Pl. C++ esetén K[i][j].

Mátrixok

Feladat:

Egy N×M-es raszterképet kicsinyítsünk a felére (N/2×M/2 méretűre) pontátlagolással: a kicsinyített kép minden pontja az eredeti kép 2×2 pontjának „átlaga” legyen!

„átlag”: színkódok átlaga



Mátrixok

Specifikáció: (másolás)

Feladat:
Egy N*M-es raszterképet kicsinyítsünk a felére (N/2*M/2 méretűre) *pontátlagolás*sal: a kicsinyített kép minden pontja az eredeti kép 2*2 pontjának „átlaga" legyen!

- Bemenet: $N,M \in \mathbb{N}$, $K_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Kimenet: $KK_{1..N/2,1..M/2} \in \mathbb{N}^{N/2 \times M/2}$
- Előfeltétel: PárosE(N) és PárosE(M)
- Utófeltétel: $\forall i(1 \le i \le N/2)$: $\forall j(1 \le j \le M/2)$:
  $KK_{i,j}=(K_{2*i,2*j}+K_{2*i-1,2*j}+K_{2*i,2*j-1}+K_{2*i-1,2*j-1})$ Div 4
- Definíció: PárosE: $\mathbb{N} \to \mathbb{L}$
  PárosE(x):=(x Mod 2)=0

Mátrixok

Algoritmus:

➢ Utófeltétel: $\forall i(1 \leq i \leq N/2): \forall j(1 \leq j \leq M/2):$
$KK_{i,j}=(K_{2*i,2*j}+K_{2*i-1,2*j}+K_{2*i,2*j-1}+K_{2*i-1,2*j-1})$ Div 4

Változó
i,j:**Egész**

i=1..N/2
j=1..M/2
KK[i,j]:=(K[2*i,2*j]+K[2*i–1,2*j]+
K[2*i,2*j–1]+K[2*i–1,2*j–1]) Div 4

Megjegyzés:

1) a színes képeknél az átlagolással baj lehet! Milyen szín egy piros és egy kék színű pont átlaga? (hamis színek)
2) **RGB** esetén a szín: **Rekord**(piros,zöld,kék:**Egész**); és az átlag? (komponensenkénti átlag)

Mátrixok

Feladat:

A Rák-köd képére alkalmazzunk egyféle Rank-szűrőt!
Minden pontot helyettesítsünk magának és a 8 szomszédjának maximumával!

→

Mátrixok

Specifikáció: (másolás+maximum-kiválasztás)

- Bemenet: $N,M \in \mathbb{N}$, $K_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Kimenet: $RK_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Előfeltétel: –
- Utófeltétel: $\forall i(1<i<N): \forall j(1<j<M):$

$$RK_{i,j} = \max_{p=i-1}^{i+1} \max_{q=j-1}^{j+1} K_{p,q} \text{ és}$$

$$\forall j(1 \leq j \leq M): RK_{1,j}=K_{1,j} \quad \text{és} \quad RK_{N,j}=K_{N,j}$$

$$\forall i(1 \leq i \leq N): RK_{i,1}=K_{i,1} \quad \text{és} \quad RK_{i,M}=K_{i,M}$$

Feladat:

A Rák-köd képére alkalmazzunk egyféle *Rank-szűrőt*! Minden pontot helyettesítsünk magának és a 8 szomszédjának maximumával!

Mátrixok

Algoritmus:

➢ Utófeltétel: ∀i(1<i<N): ∀j(1<j<M):

$RK_{i,j} = \max_{p=i-1}^{i+1} \max_{q=j-1}^{j+1} K_{p,q}$ és

∀i (1≤i≤N): ∀j (1≤j≤M):

$RK_{1,j}=K_{1,j}$ és $RK_{N,j}=K_{N,j}$

$RK_{i,1}=K_{i,1}$ és $RK_{i,M}=K_{i,M}$

Változó
max,
i,j:**Egész**

i=2..N–1

j=2..M–1

max:=0

p=i–1..i+1

q=j–1..j+1

K[p,q]>max

I | N

max:=K[p,q] | —

max:=…(i,j) körüli 3×3-as részmátrix maximuma…

RK[i,j]:=max

…

Mátrixok

Algoritmus:

- Utófeltétel: $\forall i(1<i<N)$: $\forall j(1<j<M)$:

  $RK_{i,j} = \max_{p=i-1}^{i+1} \max_{q=j-1}^{j+1} K_{p,q}$ és

  $\forall i\ (1 \le i \le N)$: $\forall j\ (1 \le j \le M)$:

  $RK_{1,j}=K_{1,j}$ és $RK_{N,j}=K_{N,j}$

  $RK_{i,1}=K_{i,1}$ és $RK_{i,M}=K_{i,M}$

Specifikáció:

- Bemenet: $N \in \mathbb{N}$, $X \in H^N$
- Kimenet: $Max \in \mathbb{N}$
- Előfeltétel: $N>0$
- Utófeltétel: $1 \le Max \le N$ és $\forall i\ (1 \le i \le N)$: $X_{Max} \ge X_i$

Max:=X[1]

i=2..N

X[i]>Max (I / N)

Max:=X[i] | —

Maximumérték-kiválasztás tétel.

Változó max, i,j:**Egész**

i=2..N–1

j=2..M–1

max:=0

p=i–1..i+1

q=j–1..j+1

K[p,q]>max (I / N)

max:=K[p,q] | —

RK[i,j]:=max

. . .

Mátrixok

Algoritmus (folytatás):

- Utófeltétel: $\forall i(1<i<N): \forall j(1<j<M):$
  $RK_{i,j}=\max_{p=i-1}^{i+1} \max_{q=j-1}^{j+1} K_{p,q}$ és
  $\forall j(1 \leq j \leq M):$
  $RK_{1,j}=K_{1,j}$ és $RK_{N,j}=K_{N,j}$
  $\forall i(1 \leq i \leq N):$
  $RK_{i,1}=K_{i,1}$ és $RK_{i,M}=K_{i,M}$

…

j=1..M

RK[1,j]:=K[1,j]

RK[N,j]:=K[N,j]

i=1..N

RK[i,1]:=K[i,1]

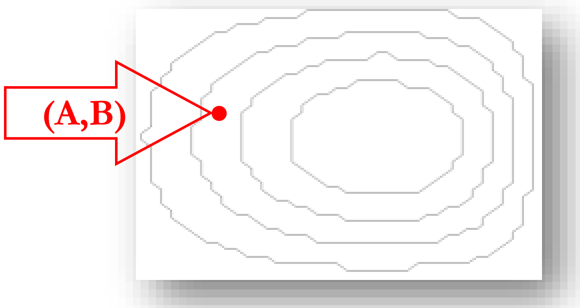
RK[i,M]:=K[i,M]

Változó
i,j:**Egész**

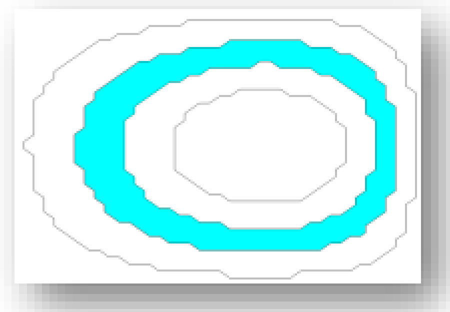
Mátrixok

Feladat:

Egy kép egy adott (fehér színű) tartományát egy (A,B) belső pontjából kiindulva fessük be világoskékre!



→



Festendők a „**belső pontok**”, ha Belső(i,j)=Igaz.

Ahol Belső:$\mathbb{N}\times\mathbb{N}\rightarrow\mathbb{L}$

Belső(i,j)=(i=A és j=B vagy
Fehér(i,j) és
(Belső(i–1,j) vagy Belső(i+1,j) vagy
Belső(i,j–1) vagy Belső(i,j+1)))

Mátrixok

Specifikáció:

- Bemenet: $N,M \in \mathbb{N}$, $K_{1..N,1..M} \in \mathbb{N}^{N \times M}$, $A,B \in \mathbb{N}$
- Kimenet: $KK_{1..N,1..M} \in \mathbb{N}^{N \times M}$
- Előfeltétel: $A \in [1..N]$ és $B \in [1..M]$ és $K_{A,B}$=fehér
- Utófeltétel: $\forall i(1 \leq i \leq N)$: $\forall j(1 \leq j \leq M)$:
  Belső(i,j) $\rightarrow KK_{i,j}$=világoskék és
  nem Belső(i,j) $\rightarrow KK_{i,j}=K_{i,j}$

Algoritmus:

KK:=K
Festés(A,B)

Mátrixok

Algoritmus:

- Utófeltétel: $\forall i(1 \le i \le N)$: $\forall j(1 \le j \le M)$:
  Belső(i,j) $\rightarrow KK_{i,j}$=világoskék és
  nem Belső(i,j) $\rightarrow KK_{i,j}=K_{i,j}$

- Definíció:
  Belső(i,j)=(i=A és j=B vagy
  Fehér(i,j) és
  (Belső(i–1,j) vagy Belső(i+1,j) vagy
  Belső(i,j–1) vagy Belső(i,j+1)))

Festés(i,j:**Egész**)

KK[i,j]:=világoskék	
I — KK[i–1,j]=fehér — N	
Festés(i–1,j)	—
I — KK[i+1,j]=fehér — N	
Festés(i+1,j)	—
I — KK[i,j–1]=fehér — N	
Festés(i,j–1)	—
I — KK[i,j+1]=fehér — N	
Festés(i,j+1)	—

Mátrixok

Algoritmus:

- Utófeltétel: $\forall i(1 \leq i \leq N)$: $\forall j(1 \leq j \leq M)$:
  Belső(i,j) $\rightarrow KK_{i,j}$=világoskék és
  nem Belső(i,j) $\rightarrow KK_{i,j}=K_{i,j}$

- Definíció:
  Belső(i,j)=(i=A és j=B vagy
  Fehér(i,j) és
  (Belső(i–1,j) vagy Belső(i+1,j) vagy
  Belső(i,j–1) vagy Belső(i,j+1)))

animáció

Festés(i,j:**Egész**)

KK[i,j]:=világoskék	
i>1 és KK[i–1,j]=fehér	
I	N
Festés(i–1,j)	—
i<N és KK[i+1,j]=fehér	
I	N
Festés(i+1,j)	—
j>1 és KK[i,j–1]=fehér	
I	N
Festés(i,j–1)	—
j<M és KK[i,j+1]=fehér	
I	N
Festés(i,j+1)	—

Rekordok vektora

Feladat:

Egy adott napon N-szer volt földrengés. Ismerjük az egyes rengések időpontját (időrendben). Mondjuk meg, hogy hány másodpercenként volt földrengés!

Megoldás felé:

- Definiálni kellene, mi az idő!
  Az időt megadhatjuk az (óra, perc, másodperc) hármassal, azaz az idő:

  $$\text{Idő}=\text{Ó}\times \text{P}\times \text{Mp},\quad \text{Ó},\text{P},\text{Mp}=\mathbb{N}$$

- Algoritmikus sablon: **Másolás** tétel!

Rekordok vektora

Feladat:

Egy adott napon N-szer volt földrengés. Ismerjük az egyes rengések időpontját. Mondjuk meg, hogy hány másodpercenként volt földrengés!

Specifikáció:

- Bemenet: $N \in \mathbb{N}$, $R_{1..N} \in Idő^{N}$
  $Idő = Ó \times P \times Mp$, $Ó, P, Mp = \mathbb{N}$
- Kimenet: $T_{1..N-1} \in \mathbb{N}^{N-1}$
- Előfeltétel: $\forall i(1 \leq i \leq N)$: $0 \leq R_i.ó \leq 23$ és
  $0 \leq R_i.p \leq 59$ és $0 \leq R_i.mp \leq 59$ és
  $\forall i(1 \leq i < N)$: $R_i < R_{i+1}$
- Utófeltétel: $\forall i(1 \leq i \leq N-1)$: $T_i = R_{i+1} - R_i$
- Definíció: $-$ $: Idő \times Idő \rightarrow \mathbb{N}$
  $i1 - i2 :=$ … ??? …
  $<$ $: Idő \times Idő \rightarrow \mathbb{L}$
  $i1 < i2 :=$ … ??? …

Rekordok vektora

Idők különbsége

1. megoldási ötlet:
   Felfoghatjuk úgy, mint két háromjegyű szám különbsége, ahol a három jegy nem azonos alapú. (Vegyes alapú számrendszer.) Majd másodpercekké konvertáljuk.
2. megoldási ötlet:
   Kifejezzük az időket másodpercben, így már két egész szám különbségét kell kiszámolni.

   másodpercben(idő):=idő.ó*3600+idő.p*60+idő.mp

   Meggondolandó, hogy mekkora egész szám kell hozzá? (24*3600=86 400) Milyen típusú lehet? (>2 byte)

Rekordok vektora

Specifikáció:

- Bemenet: $N \in \mathbb{N}$, $R_{1..N} \in Idő^N$
  $Idő = Ó \times P \times Mp,\ \ Ó,P,Mp = \mathbb{N}$
- Kimenet: $T_{1..N-1} \in \mathbb{N}^{N-1}$
- Előfeltétel: $\forall i(1 \leq i \leq N)$: $0 \leq R_i.ó \leq 23$ és
  $0 \leq R_i.p \leq 59$ és $0 \leq R_i.mp \leq 59$ és
  $\forall i(1 \leq i < N)$: $R_i < R_{i+1}$
- Utófeltétel: $\forall i(1 \leq i \leq N-1)$: $T_i = R_{i+1} - R_i$
- Definíció:
  $- : Idő \times Idő \rightarrow \mathbb{N}$
  $i1 - i2 := i1.ó*3600+i1.p*60+i1.mp - (i2.ó*3600+i2.p*60+i2.mp)$

Rekordok vektora

Algoritmus$_1$:

- Utófeltétel: $\forall i$ $(1 \le i \le N-1)$: $T_i = R_{i+1} - R_i$
- Definíció:

  – :Idő×Idő→N

  i1 – i2 := i1.o*3600+i1.p*60+i1.mp – (i2.o*3600+i2.p*60+i2.mp)

i=1..N	
	S[i]:=R[i].ó*3600+ R[i].p*60+R[i].mp
i=1..N–1	
	T[i]:=S[i+1]–S[i]

Változó
i:**Egész**
S:Tömb[…]

Megjegyzések:

1. Egy S segédtömböt használunk.
2. A TIdők közötti „–” operátor az S-en keresztül, közvetve kerül az algoritmusba.

Rekordok vektora

Algoritmus$_2$:

- Utófeltétel: $\forall i\ (1 \le i \le N-1)$: $T_i = R_{i+1} - R_i$
- Definíció:

  másodpercben(i):=i.o*3600+i.p*60+i.mp

i=1..N
S[i]:=másodpercben(R[i])
i=1..N–1
T[i]:=S[i+1]–S[i]

Változó
i:**Egész**
S:Tömb[…]

Megjegyzések:

1. A másodpercben függvény implementálandó!
2. Ha a különbség (óra, perc, másodperc)-ben kell, akkor T[i]-ből vissza kell alakítani! **Újabb művelet.**

Rekordok vektora

Algoritmus$_3$:

- Utófeltétel: $\forall i$ $(1 \le i \le N-1)$: $T_i = R_{i+1} - R_i$
- Definíció:

  másodpercben(i):=i.o*3600+i.p*60+i.mp

Változó
i:**Egész**

i=1..N–1

T[i]:=másodpercben(R[i+1])–másodpercben(R[i])

Megjegyzés:

A másodpercben függvény segítségével (sőt anélkül is) megspórolható az S segédtömb; és így az előkészítő ciklus… de cserében majdnem minden R[i]-t kétszer számítunk át másodpercekre.

Rekordok vektora

Algoritmus$_4$:

- Utófeltétel: $\forall i\ (1 \le i \le N-1)$: $T_i = R_{i+1} - R_i$
- Definíció:

  $-$:Idő×Idő→N

  i1 $-$ i2 := i1.o*3600+i1.p*60+i1.mp $-$ (i2.o*3600+i2.p*60+i2.mp)

i=1..N–1
T[i]:=R[i+1]–R[i]

Változó
  i:**Egész**

Megjegyzés:

A $-$ operátort definiálni kell, amelyben a másodpercekben függvény (vagy annak törzse) felhasználható! Az időigényt a művelet és a függvény paraméterátadása növeli.

Sorozat → halmaz transzformáció

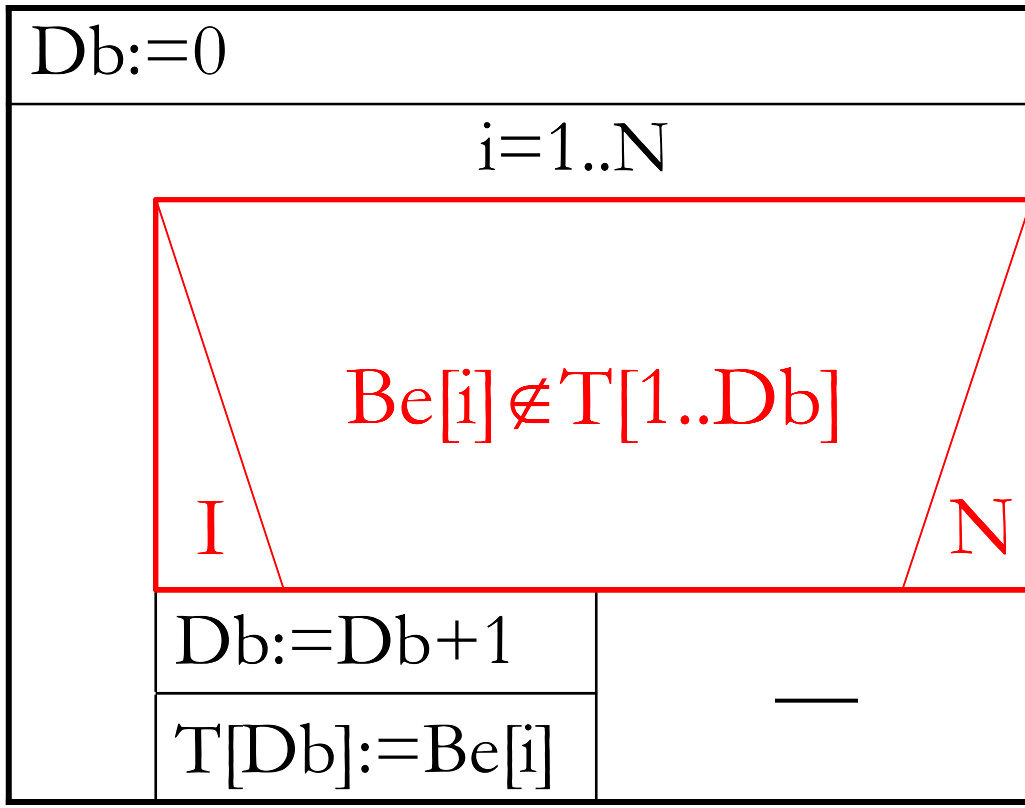
Egyes feladatoknál, mint pl. a metszet és unió tételnél a kiinduló adatok halmazban vannak. Ha a bemeneten tetszőleges sorozatot kapunk, akkor szükség lehet rá, hogy abból halmazt készítsünk.

Példa: N vásárlásról ismerjük, hogy egy vásárló milyen terméket vásárolt (Be[1..N]). Adjuk meg a vásárlásokban szereplő termékeket (T[1..Db])!

A megoldás egy **kiválogatás tétel**: válogassuk ki a bemenet azon elemeit, amelyek a kiválogatás eredményében még nem szerepeltek (**eldöntés**)!

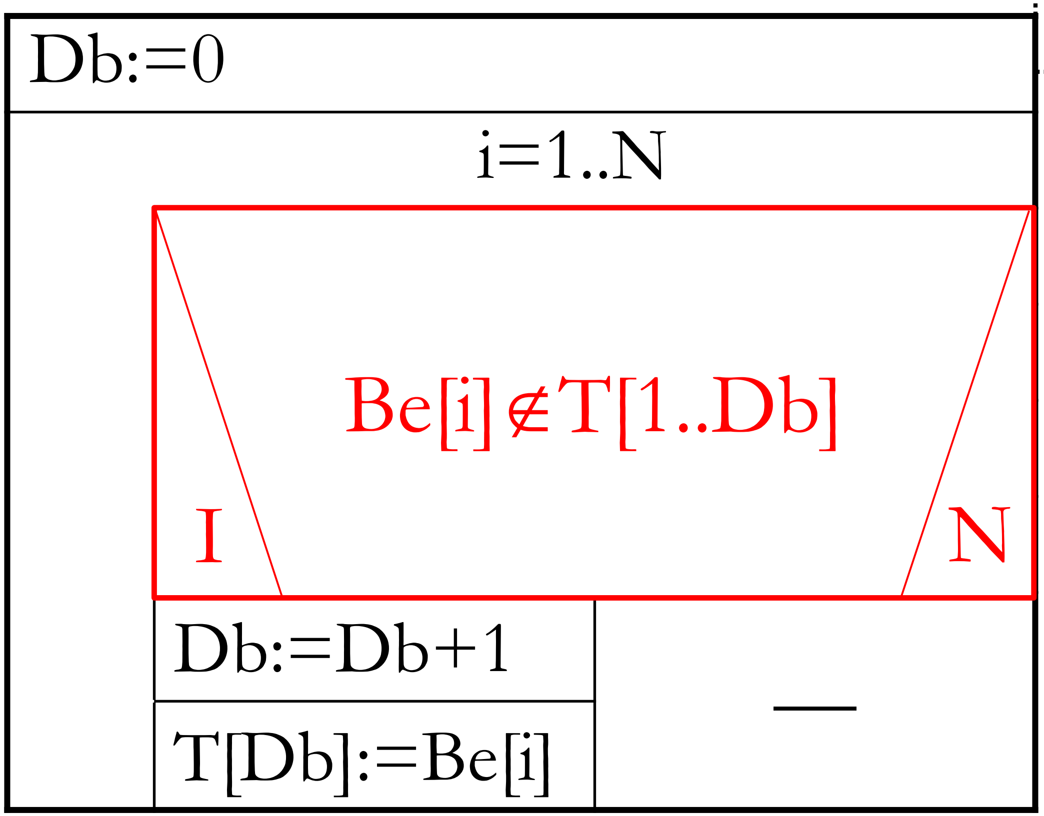
Sorozat → halmaz transzformáció

Változó
i:Egész



Sorozat → halmaz transzformáció

Változó
i,j:**Egész**



Halmaz típus

Értékhalmaz:

Az alaphalmaz (amely az Elemtípus által van meghatározva) iteráltja („mely elemek lehetnek benne a halmazban”).

Az Elemtípus általában valamely véges diszkrét típus lehet, legtöbbször még az elemszámát is korlátozzák (<256).

Ha nyelvi elemként nem létezik, akkor a megvalósításunkban lehet nagyobb elemszámú is.

Halmaz típus

Műveletek (matematika)

- metszet ($\cap$)
- unió ($\cup$)
- különbség ($\setminus$)
- komplemens – nem mindig valósítható meg
- eleme (elem benne van-e a halmazban) ($\in$)
- része (egyik halmaz részhalmaza-e a másiknak) ($\subset, \subseteq$)

Halmaz típus

Műveletek (megvalósítás)

- Halmazba (elem hozzá vétele egy halmazhoz): H:=H ∪ {e}
- Halmazból (elem elhagyása egy halmazból): H:=H \ {e}
- Beolvasás (halmaz beolvasása)
- Kiírás (halmaz kiírása),
- Üres (üres halmaz létrehozás eljárás), vagy Üres'Halmaztípus előre definiált konstans
- Üres? (logikai értékű függvény).

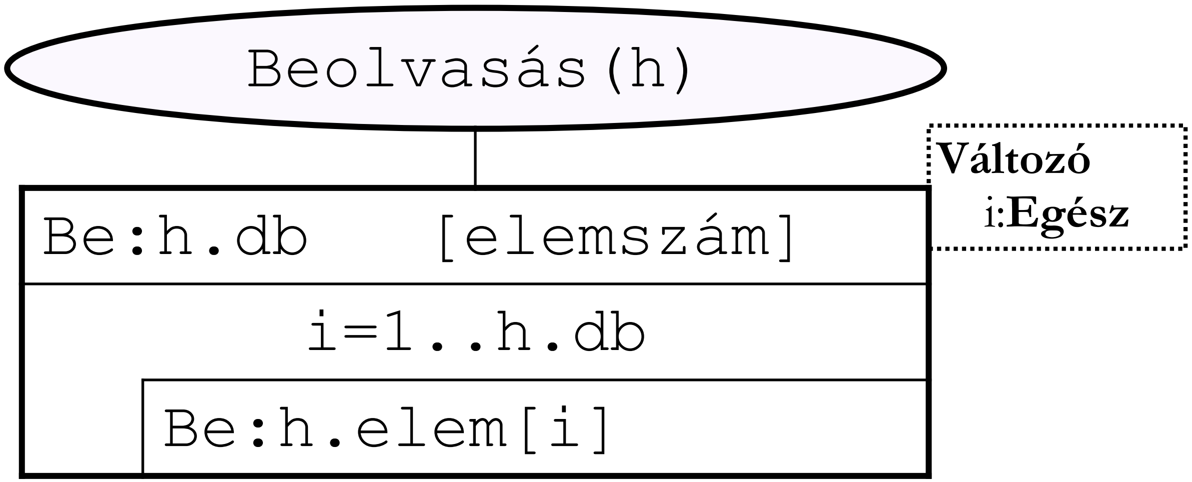
Halmaz típus ábrázolása$_1$

Elemek felsorolása

Halmaz(Elemtípus)=
Rekord(db: **Egész**,
elem: **Tömb**[1..MaxDb:Elemtípus])

A halmaz elemeinek felsorolásával adjuk meg a halmazt, annyi elemű tömbben, ahány elemű éppen a halmaz (pontosabban az első db darab elemében).

Halmaz típus



Feltesszük, hogy „halmazság” és a méretkorlát teljesül.

Műveletigény számítása:

A ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a halmaz elemszámával arányos.

Halmaz típus

```
Kiírás(h)
Ki:h.db     [elemszám]
i=1..h.db
    Ki:h.elem[i]
```

Változó
i:**Egész**

Műveletigény számítása:

A ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a halmaz elemszámával arányos.

Halmaz típus

Üres(h)

h.db:=0

Műveletigény számítása:

Nem függ a halmaz elemszámától.

Üres?(h):Logikai

Üres?:=h.db=0

Műveletigény számítása:

Nem függ a halmaz elemszámától.

Halmaz típus

```
Halmazba(h,e)
i:=1
i≤h.db és h.elem[i]≠e
    i:=i+1
i>h.db
I                           N
h.db:=h.db+1           |  —
h.elem[h.db]:=e        |
```

Változó
i:**Egész**

Az Eldöntés programozási tétel alkalmazása

Műveletigény számítása:

A ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a halmaz elemszámával arányos.

Halmaz típus

A Keresés programozási tétel alkalmazása

```
Halmazból(h,e)

i:=1
i≤h.db és h.elem[i]≠e
    i:=i+1
i≤h.db
I                                   N
h.elem[i]:=h.elem[h.db]     |   –
h.db:=h.db-1                |
```

Vált
i:E

Műveletigény számítása:

A ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a halmaz elemszámával arányos.

Halmaz típus

```
Eleme(e,h):Logikai
  i:=1
  i≤h.db és h.elem[i]≠e
    i:=i+1
  Eleme:=i≤h.db
```

Változó
i:**Egés**

Az Eldöntés programozási tétel alkalmazása

Műveletigény számítása:

A ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a halmaz elemszámával arányos.

Halmaz típus

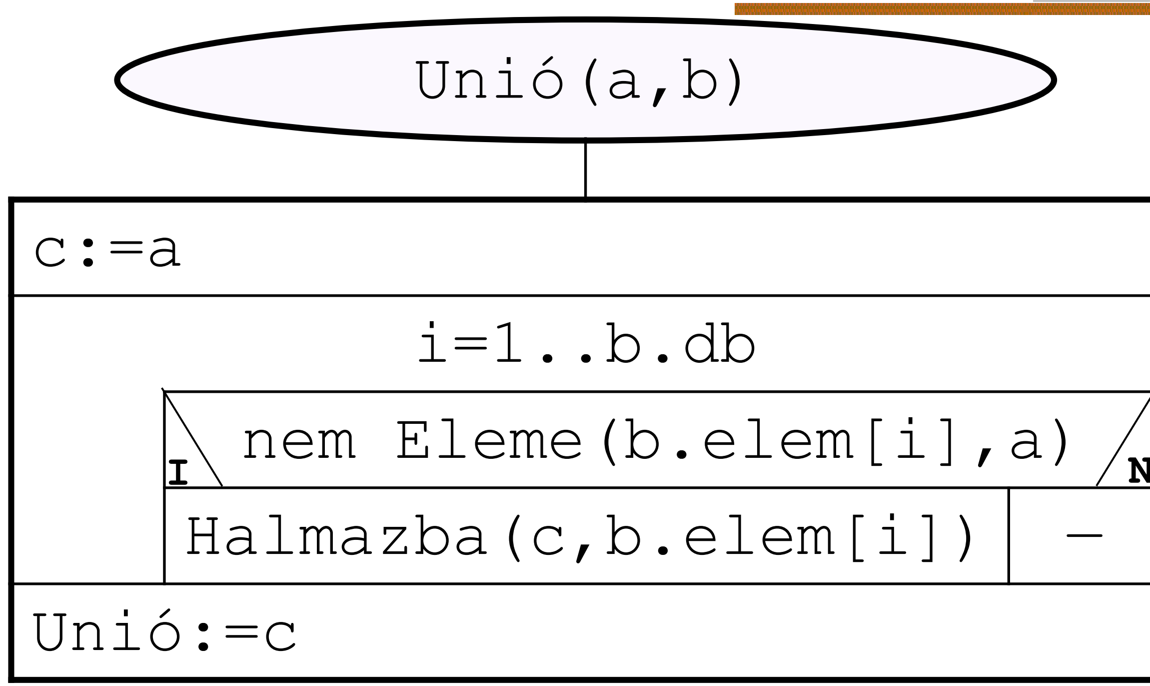
```
Része(a,b):Logikai
i:=1
i≤a.db és Eleme(a.elem[i],b)
    i:=i+1
Része:=i>a.db
```

Az Eldöntés programozási tétel alkalmazása, eldöntés tulajdonsággal

Műveletigény számítása:

A ciklus az A halmaz elemszámaszor fut le, az Eleme függvény pedig a B halmaz elemszámaszor, azaz a futási idő a két halmaz elemszámának szorzatával arányos.

Halmaz típus



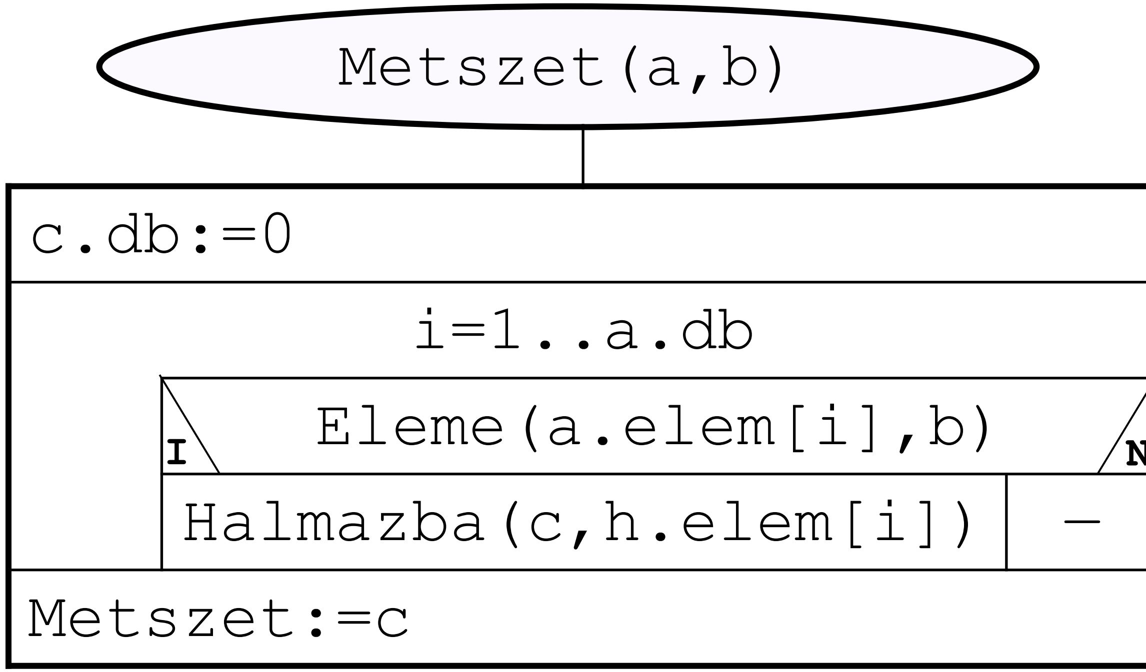
Változó
i:**Egés**
c:Haln

Másolás
+Kiválogatás
+Eldöntés

Műveletigény számítása:

A külső ciklus a B halmaz elemszámaszor fut le, az Eleme függvény pedig az A halmaz elemszámaszor, azaz a futási idő a két halmaz elemszámának szorzatával arányos.

Halmaz típus



Változó
i:**Egész**
c:Halma

Kiválogatás
+Eldöntés

Műveletigény számítása:

A ciklus az A halmaz elemszámaszor fut le, az Eleme pedig legrosszabb esetben a B halmaz elemszámaszor, azaz a futási idő a két halmaz elemszámának szorzatával arányos.

Halmaz típus

Megjegyzések:

Az így ábrázolt halmazok elemtípusára semmilyen megkötést nem kell tennünk, hiszen egy tömbben bármilyen elem elhelyezhető.

Arra sincs korlátozás, hogy mekkora lehet az alaphalmaz számossága, amiből a halmaz elemei származnak. Csak a konkrét halmazok elemszámát korlátozzuk.

Halmaz típus ábrázolása$_2$

Bittérkép – logikai vektor

Halmaz(Elemtípus)=
Tömb[Min'Elemtípus..Max'Elemtípus:Logikai]

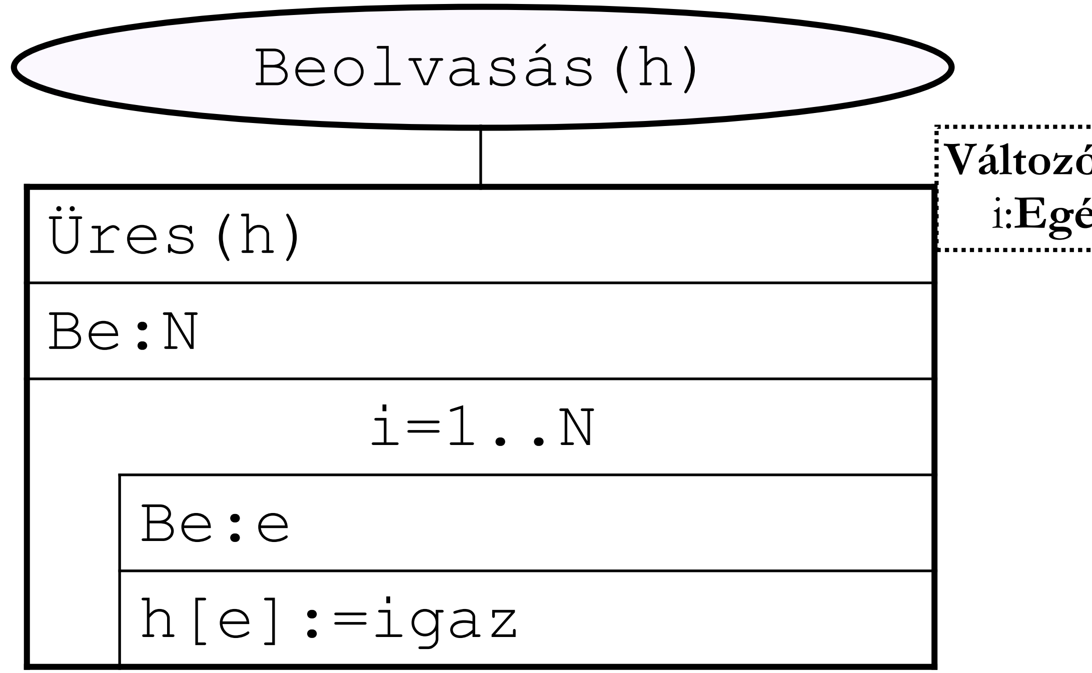
A halmazt {igaz,hamis} (azaz benne van-e) elemekből álló vektorként értelmezzük, ahol **index**ként használjuk az **elem típus**ú értéket vagy indexet számolunk belőle.

Elemtípus például lehet:
- egész számok intervalluma (-9..9)
- karakter-intervallum ("A".."Z")

Az ilyen halmaz mindig rendezett halmaz, definiálható rajta a távolság fogalom (→implementálható a tömb címkiszámító függvénye).

Kérdés: tároljuk-e a halmaz elemszámát is?

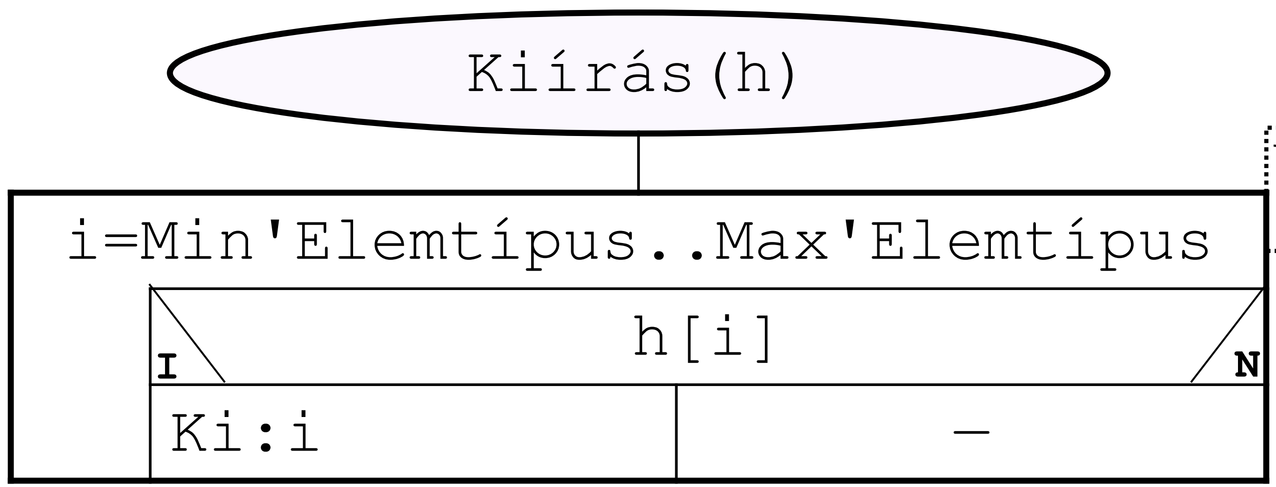
Halmaz típus



Műveletigény számítása:

Az Üres műveletigénye + a ciklus. A ciklus a halmaz elemeinek számaszor fut le, azaz a futási idő a halmaz elemszámával arányos.

Halmaz típus



Változó
i:Elemtípus

Műveletigény számítása:

A ciklus a halmaz lehetséges elemeinek számaszor fut le, azaz a futási idő a halmaz elemtípusának számosságával arányos.

Mi lenne, ha tárolnánk a halmaz legkisebb és legnagyobb elemét is?

Halmaz típus

```
Üres(h)
i=Min'Elemtípus..Max'Elemtípus
  h[i]:=hamis
```

A Másolás programozási tétel alkalmazása

Műveletigény számítása:

A ciklus a halmaz lehetséges elemeinek számaszor fut le, azaz a futási idő a halmaz elemtípusának számosságával arányos.

Halmaz típus

Az Eldöntés programozási tétel alkalmazása

```
Üres?(h):Logikai
i:=Min'Elemtípus
  i≤Max'Elemtípus és nem h[i]
    i:=i+1  [=következő Elemtípusú érték]
Üres?:=i>Max'Elemtípus
```

Műveletigény számítása:

A ciklus a halmaz lehetséges elemeinek számaszor fut le, azaz a futási idő a halmaz elemtípusának számosságával arányos.

Ha elemszámot tárolnánk, gyors lehetne (Db=0?).

Halmaz típus

```
Halmazba(h,e)
h[e]:=igaz
```

Műveletigény számítása:

Nem függ a halmaz elemszámától.

```
Halmazból(h,e)
h[e]:=hamis
```

Műveletigény számítása:

Nem függ a halmaz elemszámától

Halmaz típus

```
Eleme(e,h):Logikai
Eleme:=h[e]
```

Műveletigény számítása:

Nem függ a halmaz elemszámától.

Halmaz típus

Az Eldöntés programozási tétel alkalmazása

```
Része(a,b):Logikai
i:=Min'Elemtípus
i≤Max'Elemtípus és nem (a[i] és nem b[i])
    i:=i+1
Része:=i>Max'Elemtípus
```

Műveletigény számítása:

A ciklus a halmaz lehetséges elemeinek számaszor fut le, azaz a futási idő a halmaz elemtípusának számosságával arányos. Gyorsabb az előző ábrázolásnál, ha ez kisebb a két elemszám szorzatánál.

Halmaz típus

A Másolás programozási tétel alkalmazása:

```
Unió(a,b)
i=Min'Elemtípus..Max'Elemtípus
    c[i]:=a[i] vagy b[i]
Unió:=c
```

Műveletigény számítása:

A ciklus a halmaz lehetséges elemeinek számaszor fut le, azaz a futási idő a halmaz elemtípusának számosságával arányos. Gyorsabb az előző ábrázolásnál, ha ez kisebb a két elemszám szorzatánál.

Halmaz típus

A Másolás programozási tétel alkalmazása

```
Metszet(a,b)
i=Min'Elemtípus..Max'Elemtípus
    c[i]:=a[i] és b[i]
Metszet:=c
```

Műveletigény számítása:

A ciklus a halmaz lehetséges elemeinek számaszor fut le, azaz a futási idő a halmaz elemtípusának számosságával arányos. Gyorsabb az előző ábrázolásnál, ha ez kisebb a két elemszám szorzatánál.

Áttekintés

- Összetett adatszerkezetek halmozása
  - Mátrixok – vektorok vektora
  - Rekordok vektora
- Halmazzá alakítás
- Halmaz típus elemek felsorolásával
- Halmaz típus logikai vektorral