PROGRAMOZÁS

Programozási tételek felsorolóra

Gregorics Tibor

http://people.inf.elte.hu/gt/prog

Típus

Típus-specifikáció

típus-értékek	T	$F_1 \quad \dots \quad F_n$ *olyan feladatok, amelyek állapotterében szerepel a T*	műveletek
reprezentáció	R $repr: R \to T$ $inv : R \to \mathbb{L}$	$S_1 \quad \dots \quad S_m$ *olyan programok, amelyek állapotterében szerepel az R*	implementáció

Típus-implementáció

Típus szerkezet

- elemi típusok
- összetett típusok
  - rekord
  - alternatív
  - iterált

a típusértéket reprezentáló elemek egymáshoz való viszonya

a típusértékét egy másik típus véges sok értékeinek gyűjteménye reprezentálja

T = it(E)

a típusértékét több másik típus egy-egy értékéből álló érték-együttese reprezentálja

$T = rec(s_1:T_1, \dots ,s_n:T_n)$

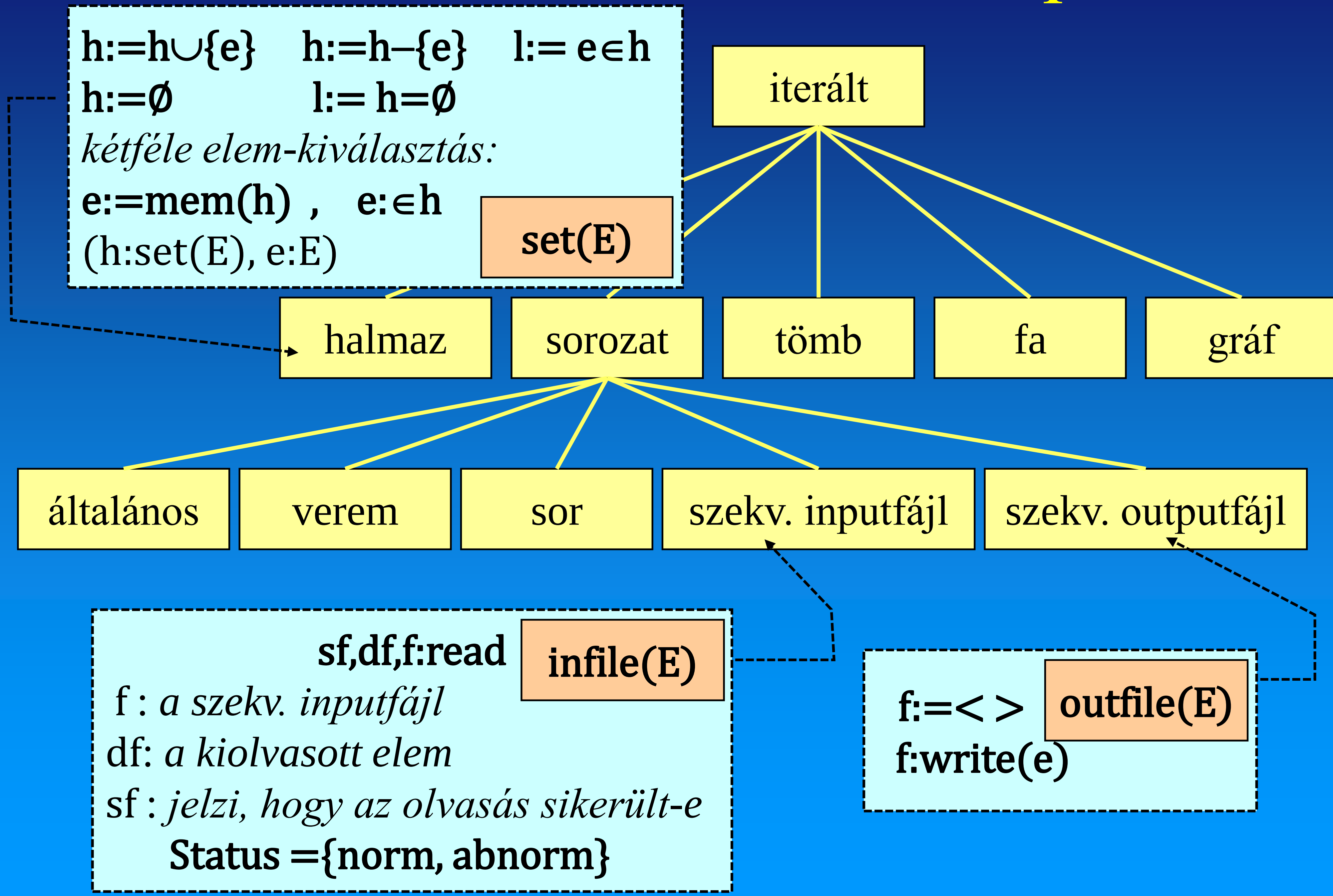
t:T-*nek* i-*edik komponense:* $t.s_i$

a típusértékét több másik típus valamelyikének értéke reprezentálja

$T = alt(s_1:T_1, \dots ,s_n:T_n)$

ha t:T *típusa* T_i, *akkor* $t.s_i$ *igaz*

Nevezetes iterált szerkezetű típusok



Gyűjtemény feldolgozása

- A gyűjtemény (tároló, kollekció, iterált) egy olyan adat (objektum), amely valamilyen elemek tárolására alkalmas.
  - Ilyenek az összetett szerkezetű, de különösen az iterált szerkezetű típusok értékei: halmaz, sorozat (verem, sor, fájl), fa, gráf .
  - Vannak úgynevezett virtuális gyűjtemények is, amely elemeit nem kell explicit módon tárolni: pl. egész számok egy intervallumának elemei, vagy egy természetes szám prím-osztói.
- Egy gyűjtemény feldolgozásán a benne levő elemek feldolgozását értjük.
  - Keressük a halmaz legnagyobb elemét!
  - Hány negatív szám van egy számsorozatban?
  - Válogassuk ki egy fa leveleiben elhelyezett értékeket!
  - Járjuk be az [*m* .. *n*] intervallum minden második elemét visszafelé!
  - Adjuk össze az *n* természetes szám prím-osztóit!

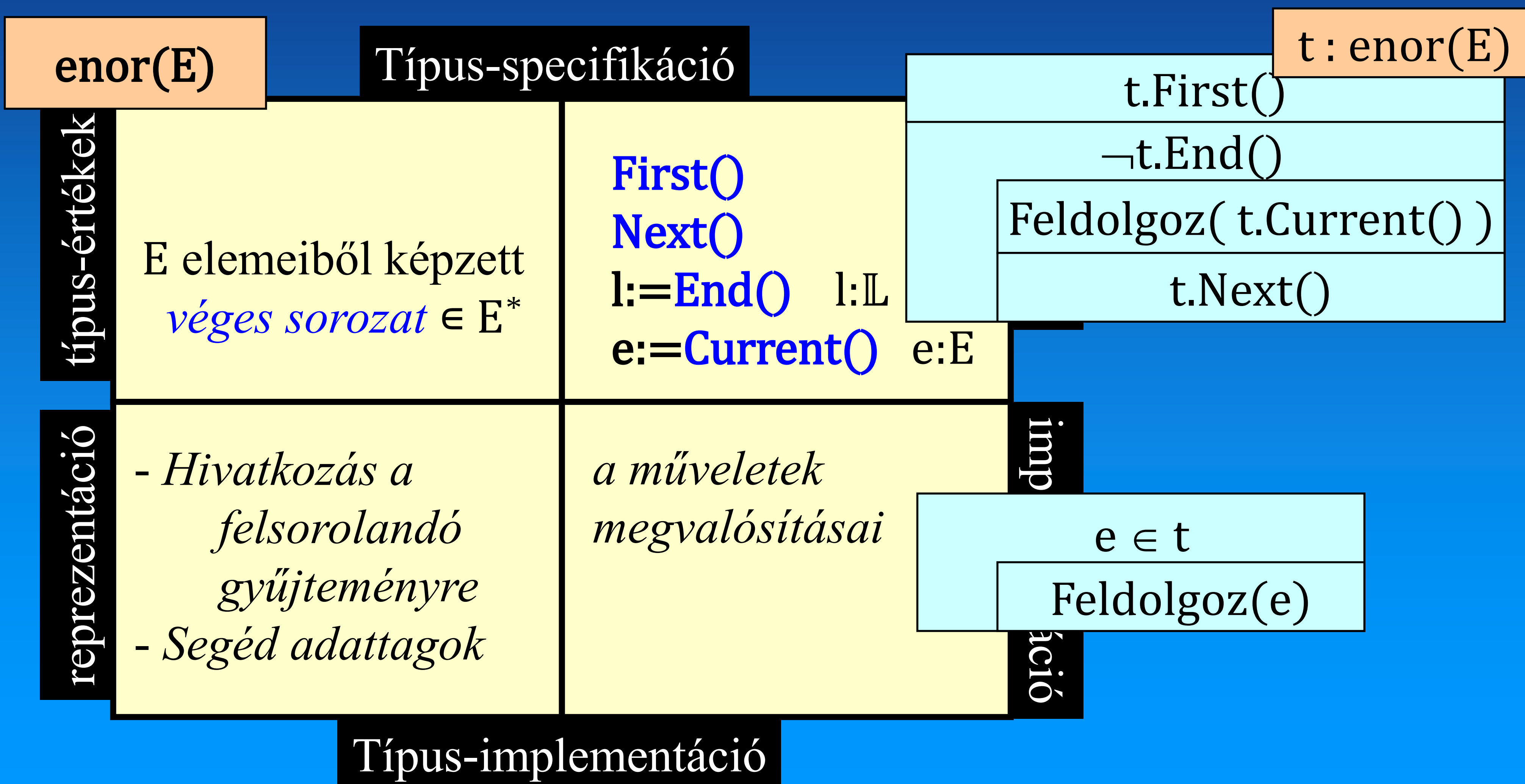
Feldolgozás felsorolással

- A feldolgozni kívánt elemek felsorolását (bejárását) az alábbi műveletekkel szabványosítjuk:
  - ***First()*** : Rááll a felsorolás első elemére, azaz elkezdi a felsorolást
  - ***Next()*** : Rááll az elkezdett felsorolás soron következő elemére
  - ***End()*** : Mutatja, ha a felsorolás végére értünk
  - ***Current()*** : Visszaadja a felsorolás aktuális elemét
- Egy felsorolásnak különböző állapotai vannak (***indulásra kész***, ***folyamatban van***, ***befejeződött***), és a műveletek csak bizonyos állapotokban értelmezhetők (máshol a hatásuk nem definiált).
- A feldolgozó algoritmus garantálja, hogy a felsoroló műveletek mindig megfelelő állapotban kerüljenek végrehajtásra.

```
First()
¬End()
    Feldolgoz(Current())
    Next()
```

Felsoroló objektum és típusa

- A felsorolást sohasem a felsorolni kívánt gyűjtemény, hanem egy külön felsoroló objektum végzi, aminek típusa jellegzetes.



Intervallum klasszikus felsorolója

Típus-specifikáció

típus-értékek / **műveletek**			$l:\mathbb{L}$	$e:\mathbb{Z}$
egész számok egy intervallumába eső egész számok növekvő sorozata	**First()**	**Next()**	**l:= End()**	**e:= Current()**
reprezentáció / **implementáció**: **m , n : $\mathbb{Z}$** **i : $\mathbb{Z}$**	**i:=m**	**i:=i+1**	**l:= i>n**	**e:=i**

Típus-implementáció

i:=m
$i \le n$
Feldolgoz(i)
i:=i+1

Vektor klasszikus felsorolója

Típus-specifikáció

típus-értékek					műveletek
	adott indexelésű **E**-beli értékekből álló vektor elemeinek sorozata elejétől a végéig	**First()**	**Next()**	l:= **End()** l:𝕃 l:𝕃	e:= **Current()** e:E e:E	
reprezentáció	$v : E^{m..n}$ $i : \mathbb{Z}$	i:=m	i:=i+1	l:= i>n	e:=v[i]	implementáció

Típus-implementáció

i:=m
i≤n
Feldolgoz(v[i])
i:=i+1

Sorozat klasszikus felsorolója

Típus-specifikáció

típus-értékek	**E**-beli értékek véges sorozata	**First()**	**Next()**	$l:\mathbb{L}$ **l:=** **End()**	e:E **e:=** **Current()**
reprezentáció	**s : E*** **i : N**	**i:=1**	**i:=i+1**	**l:= i>|s|**	**e:=s_i**

műveletek

implementáció

Típus-implementáció

$i:=1$
$i \le \lvert s \rvert$
Feldolgoz(s_i)
$i:=i+1$

Mátrix sorfolytonos felsorolója

Típus-specifikáció

típus-értékek	**E**-beli értékekből álló mátrix sorfolytonos sorrendben vett elemeinek sorozata	**First()**	**Next()**	l:$\mathbb{L}$ **l:=** **End()**	e:E **e:=** **Current()**	műveletek
prezentáció	**a : E$^{n\times m}$** **i, j : $\mathbb{Z}$**	**i,j:=1,1**	**if j<m then j:=j+1 else i,j:=i+1,1**	**l:= i>n**	**e:=a[i,j]**	implementáció

nentáció

i,j:=1,1
i≤n
Feldolgoz(a[i,j])
j<m
j:=j+1 | i,j:=i+1,1

helyett:

i = 1 .. n
j = 1 .. m
Feldolgoz(a[i,j])

Halmaz felsorolója

Típus-specifikáció

típus-értékek	véges halmaz **E**-beli elemeinek sorozata	**First()**	**Next()**	l:𝕃 **l:=** **End()**	e:E **e:=** **Current()**	műveletek
reprezentáció	**h : set(E)**	—	h:=h-{mem(h)}	l:= h=∅	e:= mem(h)	implementáció

Típus-implementáció

—
$h \neq \emptyset$
Feldolgoz(mem(h))
h:=h-{mem(h)}

Szekvenciális inputfájl felsorolója

Típus-specifikáció

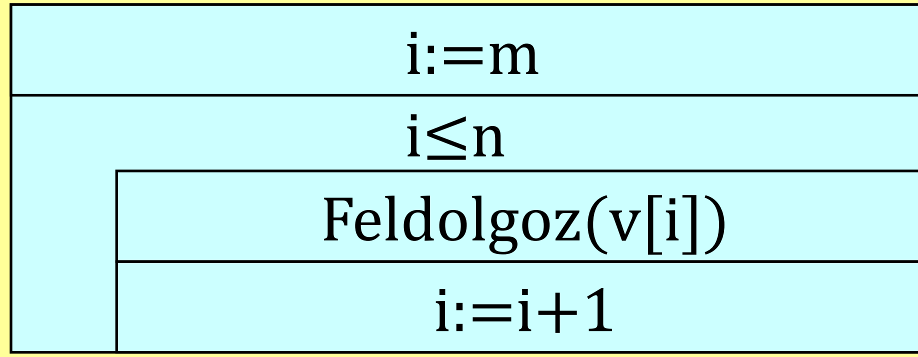
típus-értékek	szekvenciális inputfájl **E**-beli elemeinek sorozata	**First()**	**Next()**	l:𝕃 **l:=** **End()**	e:E **e:=** **Current()**	műveletek
reprezentáció	**f : infile(E)** **df : E** **sf : Status**	sf,df,f:read	sf,df,f:read	l:= sf=abnorm	e:= df	implementáció

Típus-implementáció

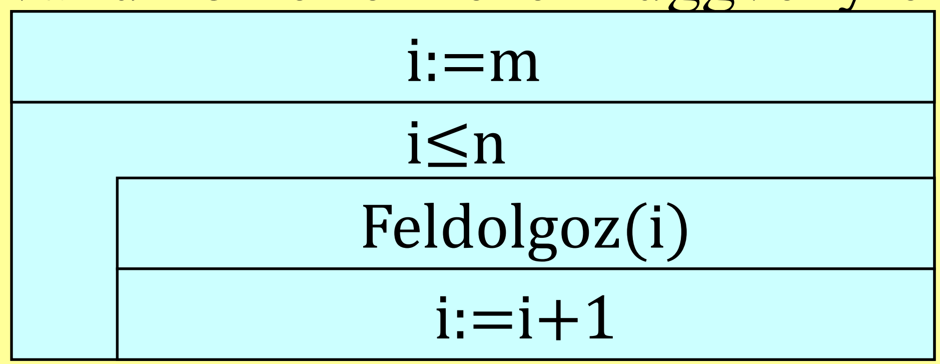
sf,df,f:read	
sf=norm	
	Feldolgoz(df)
	sf,df,f:read

Programozási tételek általánosítása

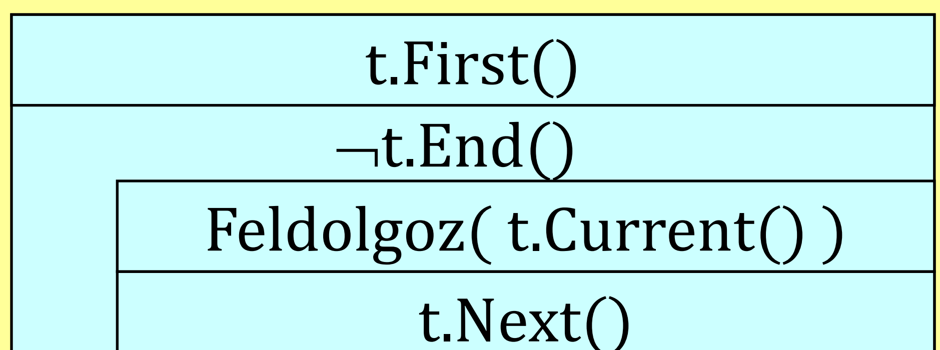
- Programozási tételek tömbre
  - v : tömb([m .. n]: E)
  - f : E→H, β:E → 𝕃



- Programozási tételek intervallumon értelmezett függvényre
  - f : [m .. n] → H
  - β: [m .. n] → 𝕃



- Programozási tételek felsorolóra
  - t : enor(E)
  - f : E → H, β:E → 𝕃



Összegzés

$f : E \to H$

$+ : H \times H \to H$ *asszociatív, baloldali neutrális elemmel*

A : (t:enor(E), s:H)

Ef : (t = t')

$$Uf: \left(s = \sum_{i=1}^{|t'|} f(t'_i) \right) = \left(s = \sum_{e \in t'} f(e) \right)$$

s := 0

t.First()

¬t.End()

    s := s+f(t.Current())

    t.Next()

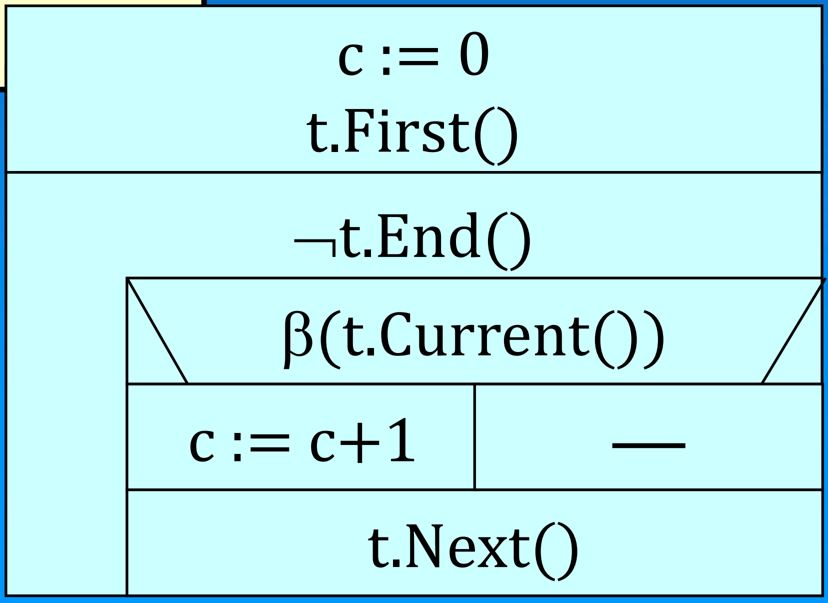
Számlálás

$\beta : E \to \mathbb{L}$

$A\ :\ (t{:}enor(E),\ c{:}\mathbb{N})$

$Ef : (\ t = t'\ )$

$$Uf: \left(c = \sum_{\substack{i=1 \\ \beta(t'_i)}}^{|t'|} \mathbf{1} \right) = \left(c = \sum_{\substack{e \in t' \\ \beta(e)}} \mathbf{1} \right)$$



Maximum kiválasztás

$f : E \to H$

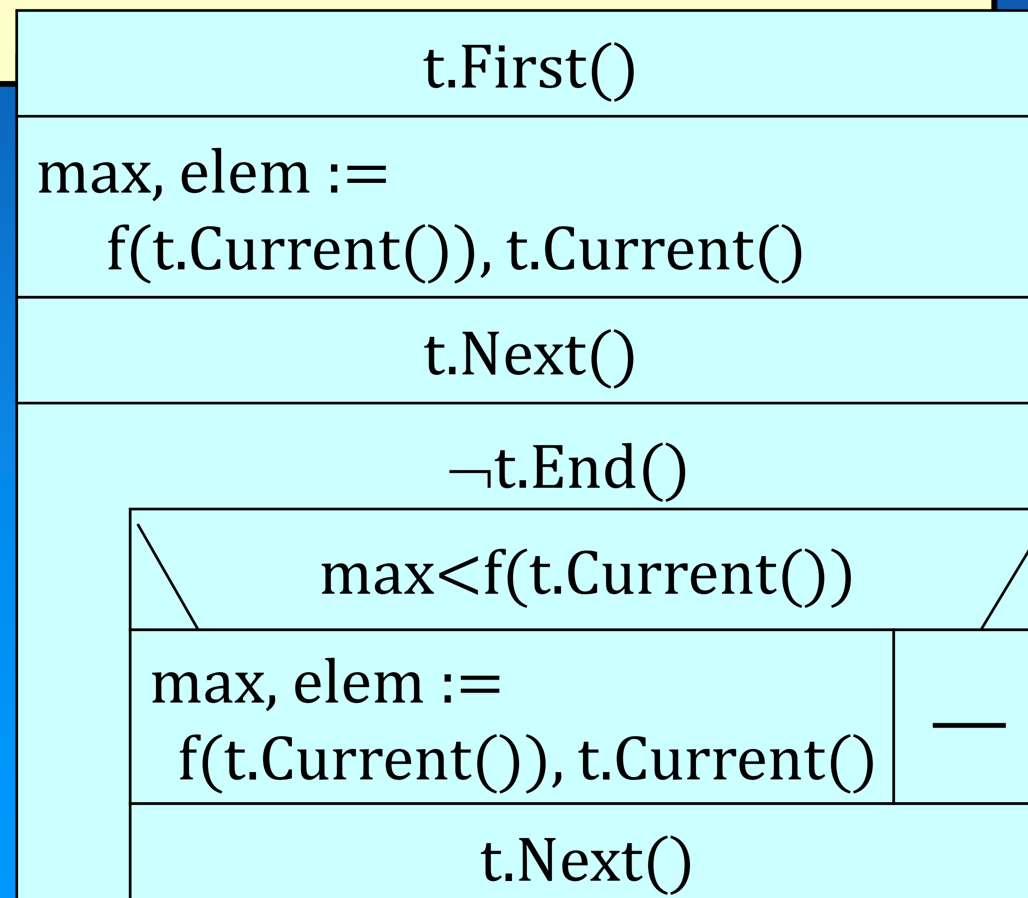
H *halmaz elemei rendezhetőek*

$A\;:\; (t{:}enor(E),\ elem{:}E,\ max{:}H$

$Ef : (t = t' \wedge |t|>0\ )$

$Uf : (\ (max\ ,\ ind) = \mathbf{MAX}_{i=1}^{|t'|}\ f(t'_i) \wedge elem = t'_{ind}\ )$

$\quad = (\ (max\ ,\ elem) = \mathbf{MAX}_{e \in t'}\ f(e)\ )$



Kiválasztás

$\beta : E \to \mathbb{L}$

$A\ : (t{:}enor(E),\ elem{:}E\ )$

$Ef : (\ t = t' \wedge\ \exists i \in [1\ ..\ |t|] : \beta(t_i)\ )$

$\exists e \in t : \beta(e)$

$$Uf : (\ (ind, t) = \mathbf{SELECT}_{ind \geq 1}\ \beta(t'_{ind}) \wedge elem = t'_{ind}\ )$$

$$= (\ (elem, t) = \mathbf{SELECT}_{e \in t'}\ \beta(e)\ )$$

A t felsorolása a kiválasztás végén még „*folyamatban van*” és $t = t'[ind\ ..\ |t'|]$

t.First()

$\neg\beta$(t.Current())

t.Next()

elem := t.Current()

Lineáris keresés

$\beta : E \to \mathbb{L}$

$A \;:\; (t{:}enor(E),\ l{:}\mathbb{L},\ elem{:}E)$

$Ef : (t = t')$

$$Uf : \left((l, ind, t) = \mathbf{SEARCH}_{i=1}^{|t'|}\ \beta(t'_i) \wedge elem = t'_{ind} \right)$$

$$= \left((l, elem, t) = \mathbf{SEARCH}_{e \in t'}\ \beta(e) \right)$$

A t felsorolása sikeres keresés végén még „*folyamatban van*” és ilyenkor $t = t'[ind\ ..\ |t'|]$, sikertelen keresés esetén „*befejeződött*”

eldöntésre is használható: $\exists$

itt is van optimista eldöntés: $\forall$

$l := hamis;\ t.First()$

$\neg l \wedge \neg t.End()$

$elem := t.Current()$

$l := \beta(elem)$

$t.Next()$

Feltételes maximum keresés

$f : E \to H$
$\beta : E \to \mathbb{L}$
H *halmaz elemei rendezhetőek*

$$A : (t{:}enor(E),\ l{:}\mathbb{L},\ elem{:}E,\ max{:}H\ )$$
$$Ef : (\ t = t'\ )$$
$$Uf : (\ (l, max\ , ind) = \mathbf{MAX}_{\substack{i=1 \\ \beta(t'_i)}}^{|t'|} f(t'_i) \wedge elem = t'_{ind}\ )$$
$$= (\ (l, max, elem) = \mathbf{MAX}_{\substack{e \in t' \\ \beta(e)}} f(e)\ )$$

Feltételes maximum keresés

l := hamis
t.First()

¬t.End()

¬β(t.Current())	l ∧ β(t.Current())		¬l ∧ β(t.Current())
	max<f(t.Current())		
—	max, elem := f(t.Current()), t.Current()	—	l, max, elem := igaz, f(t.Current()), t.Current()

t.Next()