

5. Algoritmus minták felsorolóra

- Válogassuk ki kaktuszok sorozatából egyrészt a piros virágú kaktuszoknak, másrészt a mexikói őshazájú kaktuszoknak neveit!

Specifikáció:

$A = (x : \text{enor}(\text{Kaktusz}), y, z : \mathbb{S}^*)$
 $\text{Kaktusz} = \text{rec}(\text{név}:\mathbb{S}, \text{szín}:\mathbb{S}, \text{ős}:\mathbb{S}, \text{méret}:\mathbb{N})$
 $Ef = (x = x_0)$
 $Uf = (y = \bigoplus_{e \in x_0} \langle e.\text{név} \rangle \wedge z = \bigoplus_{e \in x_0} \langle e.\text{név} \rangle)$
 $\quad \quad \quad e.\text{szín} = \text{"piros"} \quad \quad \quad e.\text{ős} = \text{"Mexikó"}$

2 összegzés (kiválogatás) közös ciklusba vonva

$H, +, 0 \sim (\mathbb{S}^*, \bigoplus, <>), (\mathbb{S}^*, \bigoplus, <>)$
 $f_1(e) \sim \langle e.\text{név} \rangle \text{ ha } e.\text{szín} = \text{"piros"}$
 $f_2(e) \sim \langle e.\text{név} \rangle \text{ ha } e.\text{ős} = \text{"Mexikó"}$

Szekvenciális inputfájltra:

$y, z := <>, <>$ $st, e, x : \text{read}$	$e : \text{Kaktusz}$
$st = \text{norm}$	$st : \text{Status}$
$e.\text{szín} = \text{"piros"}$	
$y := y \bigoplus \langle e.\text{név} \rangle$	—
$e.\text{ős} = \text{"Mexikó"}$	
$z := z \bigoplus \langle e.\text{név} \rangle$	—
$st, e, x : \text{read}$	

Algoritmus:

$y, z := <>, <>$	
$e \text{ in } x$	$e : \text{Kaktusz}$
$e.\text{szín} = \text{"piros"}$	
$y := y \bigoplus \langle e.\text{név} \rangle$	—
$e.\text{ős} = \text{"Mexikó"}$	
$z := z \bigoplus \langle e.\text{név} \rangle$	—

Lehetne indexeléssel is implementálni, és ehhez akár számlálós ciklust ($i=1 \dots |x|$) is használhatunk.

Tömbre:

$y, z := <>, <>$	$i : \mathbb{N}$
$i := 1$	
$i \leq x $	
$x[i].\text{szín} = \text{"piros"}$	
$y := y \bigoplus \langle e.\text{név} \rangle$	—
$x[i].\text{ős} = \text{"Mexikó"}$	
$z := z \bigoplus \langle e.\text{név} \rangle$	—
$i := i + 1$	

- Keressük meg egy pozitív egész számokat tartalmazó nem üres sorozatban a legnagyobb számot, és közben döntsük el azt is, hogy vajon minden szám páros-e.

Specifikáció:

$A = (x : \text{enor}(\mathbb{N}^+), l : \mathbb{L}, m : \mathbb{N})$
 $Ef = (x = x_0 \wedge |x| \geq 1)$
 $Uf = ((m, _) = \text{MAX}_{e \in x_0} e \wedge (l, _) = \text{VSEARCH}_{e \in x_0} (e \text{ páros}))$

Az utófeltétel másképpen: két összegzéssel

$Uf = (m = \text{MAX}_{e \in x_0} e \wedge l = \text{AND}_{e \in x_0} (e \text{ páros}))$
 ahol $\text{max} : \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$, $\text{max}(a, b) ::= \max(a, b)$ neutrális elem: 0
 $\text{and} : \mathbb{L} \times \mathbb{L} \rightarrow \mathbb{L}$, $\text{and}(a, b) ::= a \wedge b$ neutrális elem: igaz

Két összegzés összevonva

$H, +, 0 \sim (\mathbb{N}, \text{max}, 0), (\mathbb{L}, \text{and}, \text{igaz})$
 $f(e) \sim e, e \text{ páros}$
 $s \sim m, l$

Algoritmus:

$m, l := 0, \text{igaz}$	
$e \text{ in } x$	$e : \mathbb{N}$
$e > m$	
$m := e$	—
$l := l \wedge e \text{ páros}$	$m := \text{max}(m, e)$

3. Adott egész számoknak egy felsorolása.

a) Hány páros szám van az első negatív szám előtt?

Specifikáció:

$$A = (x: \text{enor}(\mathbb{Z}), db: \mathbb{N})$$

$$Ef = (x = x_0)$$

$$Uf = (db = \sum_{\substack{e \in x_0 \\ e \text{ páros}}}^{e \geq 0} 1)$$

A Σ jobb felső sarkába írt feltétel mutatja, hogy meddig kell a felsorolást folytatni.

Számlálás, feltétel fennállásáig

$$\text{felt}(e) \sim e \text{ páros}$$

$$c \sim db$$

$$t: \text{enor}(E) \sim x: \text{enor}(\mathbb{Z}) \text{ amíg: } e \geq 0$$

Algoritmus:

db := 0	
e in x ∧ e ≥ 0	
e páros	
db := db+1	—

vagy

db := 0	
x.Next()	
¬x.End() ∧ x.Current() ≥ 0	
x.Current() páros	
db := db+1	—
x.Next()	

b) Hány páros szám van az első negatív szám után?

Specifikáció:

$$A = (x: \text{enor}(\mathbb{Z}), db: \mathbb{N})$$

$$Ef = (x = x_0)$$

$$Uf = ((_, _, x') = \text{SEARCH}_{e \in x_0} (e < 0) \wedge db = \sum_{\substack{e \in x' \\ e \text{ páros}}} 1)$$

A linker elsődleges outputjaira (a logikai értékre és a keresett elemre) most nincs szükség, csak a helyüket kell jelölni azért, hogy nyilvánvaló legyen, hogy az x' a másodlagos outputot (a még fel nem sorolt elemek felsorolását) jelöli. Ezt folytatja a számlálás.

Algoritmus:

l := false	
e in x ∧ ¬l	
e < 0	
l := igaz	—
db := 0	
e in x	
e páros	
db := db+1	—

Másik megoldás:

$$Uf = ((_, x') = \text{SELECT}_{e \in x_0} (e < 0 \vee |x| = 0) \wedge db = \sum_{\substack{e \in x' \\ e \text{ páros}}} 1)$$

A keresés feltétele biztosan teljesül, ha vagylagosan tartalmazza az $|x|=0$ -t is: vagy találunk negatív elemet a felsorolásban vagy a felsorolás végére érünk). Ezért a lineáris keresés helyett a kiválasztás mintát is alkalmazhatjuk, aminek itt is a másodlagos outputja kell (x').

e in x ∧ e ≥ 0	
—	
db := 0	
e in x	
e páros	
db := db+1	—

c) Hány páros szám van az első negatív szám előtt, és hány azután?

Specifikáció:

$$A = (x: \text{enor}(\mathbb{Z}), \text{dbe}, \text{dbu}: \mathbb{N})$$

$$Ef = (x = x_0)$$

$$Uf = ((\text{dbe}, _, x') = \sum_{\substack{e \in x_0 \\ e \text{ páros}}}^{e \geq 0} 1 \wedge \text{dbu} = \sum_{\substack{e \in x' \\ e \text{ páros}}} 1)$$

A feltételig tartó összegzésnek másodlagos outputjai a feltételt ki nem elégítő első elem (de erre nincs szükségünk, csak a helyét jelöljük), és az ezt követő x' felsorolás.

Algoritmus:

dbe := 0		e:ℤ
e in x ∧ e≥0		
e páros		
dbe := dbe + 1	—	
dbu := 0		
e in x		
e páros		
dbu := dbu+1	—	

d) Hány páros szám van az első negatív szám előtt, és vele kezdődően hány utána?

Specifikáció:

$$A = (x: \text{enor}(\mathbb{Z}), \text{dbe}, \text{dbu}: \mathbb{N})$$

$$Ef = (x = x_0)$$

$$Uf = ((\text{dbe}, e', x') = \sum_{\substack{e \in x_0 \\ e \text{ páros}}}^{e \geq 0} 1$$

$$\wedge \text{dbu} = \sum_{\substack{e \text{ páros}}} e \text{ in } \langle e' \rangle \oplus_{x'} 1)$$

Az e' az e változó értéke, x' az x felsoroló állapota az első számlálás leállásakor. A második számlálás úgy folytatja x felsorolását, hogy ehhez figyelembe veszi az e' elemet is.

Algoritmus:

dbe := 0		e:ℤ
e in x ∧ e≥0		
e páros		
dbe := dbe + 1	—	
dbu := 0		
e in <e, x>		
e páros		
dbu := dbu+1	—	

Szekvenciális inputfájlra:

<div>dbe := 0</div> <div>st, e, x : read</div>	<div>e:ℤ</div> <div>st : Status</div>
<div>st=norm ∧ e≥0</div>	
<div>e páros</div>	
<div>dbe := dbe + 1</div> <div>—</div>	
<div>st, e, x : read</div>	
<div>dbu := 0</div>	
<div>st=norm</div>	
<div>e páros</div>	
<div>dbu := dbu+1</div> <div>—</div>	
<div>st, e, x : read</div>	

Tömbre:

dbe := 0		i:ℕ
i:=1		
i≤n ∧ x[i]≥0		
x[i] páros		
dbe := dbe +1	—	
i := i+1		
dbu := 0		
i≤n		
x[i] páros		
dbu := dbu+1	—	
i := i+1		

4. Egy más utáni napok átlaghőmérsékleteit egy szekvenciális inputfájl tartalmazza. Mennyi az első fagypont alatti értéket megelőző napok (ilyenek biztosan vannak) hőmérsékleteinek átlaga, továbbá az első fagypont alatti értéktől kezdődően (az első fagypont alatti napot is beleértve) vajon minden nap fagypont alatt maradt-e a hőmérséklet, és mi volt a legalacsonyabb hőmérséklet?

Specifikáció:

$A = (x:infile(\mathbb{R}), a:\mathbb{R}, l:\mathbb{L}, kicsi:\mathbb{R})$

$Ef = (x=x_0 \wedge \exists i \in [2..|x|]: x[i] < 0 \wedge x[1] \geq 0)$

$Uf = ((s, e', x') = \sum_{e \in x_0}^{e \geq 0} (e) \wedge (db, e', x') = \sum_{e \in x_0}^{e \geq 0} 1 \wedge a = s/db \wedge$
 $\wedge l = \bigwedge_{e \in \langle e' \rangle \oplus x'} (e < 0) \wedge kicsi = \min_{e \in \langle e' \rangle \oplus x'} e)$

$\wedge l = \bigwedge_{e \in \langle e' \rangle \oplus x'} (e < 0) \wedge kicsi = \min_{e \in \langle e' \rangle \oplus x'} e)$

Algoritmus:

$e, s:\mathbb{R}, db:\mathbb{N} \text{ st:Status}$

két összegzés közös ciklusban
szekvenciális inputfájl feltételig tartó
felsorolásával

átlagszámítás

opt. lin. ker. és min. kiv. közös ciklusban
inputfájl felsorolásának folytatásával
(az első elemet már korábban beolvastuk)

(l, kicsi := e < 0, e) inicializálás helyett,
mivel e biztosan negatív szám,
írhatjuk, hogy l, kicsi := igaz, e

s, db := 0.0, 0	
st, e, x : read	
st=norm \wedge e \geq 0	
s, db := s+e, db +1	
st, e, x : read	
a := s / db	
l, kicsi := igaz, e	
st, e, x : read	
st=norm	
l := l \wedge e < 0	
e < kicsi	
kicsi := e	—
st, e, x : read	

5. Számoljuk ki egy számítástechnikai szaküzlet napi bevételét az aznapi forgalom alapján. A forgalmat a kiadott számlák mutatják, amelyeket egy szöveges állományban (szekvenciális inputfájl) rögzítettek. Az állomány minden sora egy-egy számla adatait tartalmazza: a vásárló nevét és az általa vásárolt termékek (cikkszám és ár párok) sorozatát.

Specifikáció:

$A = (f:\text{infile}(\text{Szám}la), \text{bevét}:\mathbb{N})$ $\text{Szám}la = \text{rec}(\text{név}:\mathbb{S}, \text{lista}:\text{Áru}^*)$ $\text{Áru} = \text{rec}(\text{cikkszám}:\mathbb{S}, \text{ár}:\mathbb{N})$

$Ef = (f=f_0)$

$Uf = (\text{bevét} = \sum_{sz \text{ in } f_0} \text{össz}(sz.\text{lista}))$ ahol $\text{össz}(sz.\text{lista}) = \sum_{e \text{ in } sz.\text{lista}} e.\text{ár}$

Összegzés

$H, +, 0 \sim \mathbb{N}, +, 0$

$f(e) \sim \text{össz}(sz.\text{lista})$

ahol $\text{össz} : \text{Áru}^* \rightarrow \mathbb{N}$

$s \sim \text{bevét}$

$t:\text{enor}(E) \sim f:\text{infile}(\text{Szám}la) \ (st, sz, f:\text{read})$

Algoritmus:

bevét := 0	st:Status sz:Szám}la
st, sz, f : read	
st = norm	
bevét := bevét + össz(sz.lista)	
st, sz, f : read	

Részfeladat:

sum := össz(x)

$A = (x : \text{Áru}^*, \text{sum} : \mathbb{N})$

$Ef = (x=x_0)$

$Uf = (\text{sum} = \sum_{e \text{ in } x_0} e.\text{ár})$

Összegzés

$t:\text{enor}(E) \sim e \text{ in } x$

$f(e) \sim e.\text{ár}$

$H, +, 0 \sim \mathbb{N}, +, 0$

Algoritmus:

sum := 0	e : Áru
e in x	
sum := sum + e.ár	

Megjegyzés:

1. A tételek fenti összegzése lehet a számla (Szám}la típusú objektum) egy metódusa. A számla tételeinek (kezdetben üres) listájához egy másik metódussal lehetne felvenni egy új tételt a fájlból történő olvasás során.
2. A tételek összegzésének eredménye lehet a számla (Szám}la típusú objektum) része (adattagja), amelyet akkor módosítunk, amikor a fájlból történő olvasás során a számla tételeinek (kezdetben üres) listájához egy újabb tételt adunk hozzá. Sőt a tételek listája sem kell: elég az összegzés eredményét adattagként felvenni.

6. Egy horgászversenyen a horgászok eredményét egy szekvenciális inputfájlban rögzítették. A fájl egy eleme egy horgász nevét és a halfogásainak sorozatát tartalmazza. Egy fogás egy időpontból, a kifogott hal fajtájának nevéből, a hal hosszából (m) és súlyából (kg) áll. Keressünk olyan horgászt, aki az 50 cm-esnél hosszabb pontyokból legalább 10 kilogramnyit fogott.

Specifikáció:

$A = (x:infile(Horgász), l:\mathbb{L}, név:\mathbb{S})$

$Horgász = rec(név:\mathbb{S}, zsákmány:Fogás^*)$

$Fogás = rec(idő:\mathbb{S}, fajta:\mathbb{S}, hossz:\mathbb{R}, súly:\mathbb{R})$

$Ef = (x=x_0)$

$Uf = (l, elem) = \text{SEARCH}_{horg \text{ in } x_0} \text{összsúly}(horg.zsákmány) \geq 10.0 \wedge l \rightarrow (név = elem.név)$

ahol $\text{összsúly}(horg.zsákmány) = \sum_{hal \in horg.zsákmány} hal.súly \text{ (összsúly : } Fogás^* \rightarrow \mathbb{R} \text{)}$
 $hal.fajta = \text{"ponty"} \wedge hal.hossz \geq 0.5$

Lineáris keresés

$felt(e) \sim \text{összsúly}(horg.zsákmány) \geq 10.0$

$t:enor(E) \sim x:infile(Horgász) \text{ (st,horg,x:read)}$

Algoritmus: st:Status, horg:Horgász

l := hamis	
st, horg, x : read	
¬l ∧ st = norm	
összsúly(horg.fogás) ≥ 10.0	
l, név := igaz, horg.név	st, horg, x : read

Részfeladat: sum := pontysúly(x)

$A = (f:Fogás^*, sum:\mathbb{R}) \quad Fogás = rec(idő:\mathbb{S}, fajta:\mathbb{S}, hossz:\mathbb{R}, súly:\mathbb{R})$

$Ef = (f=f_0)$

$Uf = (f=f_0 \wedge sum = \sum_{hal \text{ in } f_0} hal.súly)$
 $hal.fajta = \text{"ponty"} \wedge hal.hossz \geq 0.5$

Összegzés (feltételes összegzés)

$H, +, 0 \sim \mathbb{R}, +, 0$

$f(e) \sim hal.súly \text{ ha } hal.fajta = \text{"ponty"} \wedge hal.hossz \geq 0.5$

$t:enor(E) \sim hal \text{ in } f$

Algoritmus:

sum := 0.0	
hal in f	
hal.fajta = "ponty" ∧ hal.hossz ≥ 0.5	
sum := sum + hal.súly	—

Megjegyzés:

1. A fogások fenti összegzése lehetne a horgász (Horgász típusú objektum) egy módszere is. Ekkor a horgász fogásainak (kezdetben üres) listájához egy másik módszerrel lehetne hozzáadni új fogást a fájlból történő olvasás során.
2. A fogások összegzésének eredménye lehetne egy adattagja a horgász objektumnak, amelyet akkor módosítunk, valahányszor egy újabb fogást teszünk hozzá horgász fogásaihoz. Ekkor a fogások listájára sincs szükség.