OEP 5. gyakorlat

Visszavezetés szekvenciális inputfájl felsorolásával <u>Témakör</u>: szekvenciális inputfájl feldolgozása programozási tételek összetételével

1. Keressük meg egy pozitív egész számokat tartalmazó nem üres szekvenciális inputfájlban a legnagyobb számot, és döntsük el azt is, hogy van-e páros szám. (Három megoldást is mutatunk.)

Első változat:

```
Specifikáció:
```

```
A = (x:infile(\mathbb{N}^+), I:\mathbb{L}, nagy:\mathbb{N})
Ef = (x=x' \land |x| \ge 1)
Uf = (nagy = MAX_{e \in x'} e \land I = SEARCH_{e \in x'} (e páros))
```

Maxkiv és linker összevonva

```
t:enor(E) \sim x:infile(\mathbb{N}^+) (st,e,x:read)
f(e) \sim e, e páros
s \sim nagy, l
```

Algoritmus:

st,e,x : read
nagy, I := e, e páros
st,e,x : read
st=norm
nagy < e
nagy := e —
e páros
I := igaz —
st,e,x : read

e:N st:Status

ÉS táblázata						
IGAZ	és	IGAZ	\Rightarrow	IGAZ		
IGAZ	és	HAMIS	\Rightarrow	HAMIS		
HAMIS	és	IGAZ	\Rightarrow	HAMIS		
HAMIS	és	HAMIS	\Rightarrow	HAMIS		

VAGY táblázata						
IGAZ	vagy	IGAZ	\Rightarrow	IGAZ		
IGAZ	vagy	HAMIS	\Rightarrow	IGAZ		
HAMIS	vagy	IGAZ	\Rightarrow	IGAZ		
HAMIS	vagy	HAMIS	\Rightarrow	HAMIS		

Második változat:

$$Uf = (\text{nagy} = \triangle_{e \in x'} e \land I = V_{e \in x'} (e \text{ páros}))$$

ahol:

$$\triangle$$
: $\mathbb{N} \times \mathbb{N} \to \mathbb{N}$, \triangle (a,b):=max(a,b), 0 V: $\mathbb{L} \times \mathbb{L} \to \mathbb{L}$, $V(a,b)$:= a \vee b, hamis

Két összegzés összevonva

t:enor(E)
$$\sim$$
 x:infile(\mathbb{N}^+) (st,e,x:read)

$$H, +, 0 \sim (\mathbb{N}, \underline{\mathbb{A}}, 0), (\mathbb{L}, V, hamis)$$

Algoritmus:

nagy, I := 0, hamis st,e,x : read

st=norm

nagy := max(nagy, e)

 $I := I \lor e páros$

st,e,x : read

e:ℕ st:Status

Harmadik változat:

$$Uf = ((nagy, I) = (AV)_{e \in x'} (e, e páros))$$
 ahol

$$(\underline{\mathbb{A}},V): (\mathbb{N},\mathbb{L})\times (\mathbb{N},\mathbb{L})\to (\mathbb{N},\mathbb{L}), (0, \text{ hamis})$$

 $(\underline{\mathbb{A}},V)((a,p),(b,q)):=(\max(a,b), (a\lor b))$

```
Egy (dupla) összegzés

t:enor(E) \sim x:infile(\mathbb{N}^+) (st,e,x:read)

f(e) \sim e, e páros

s \sim nagy, l

H, +, 0 \sim ((\mathbb{N},\mathbb{L}), (\triangle,\mathbb{V}), (0,hamis))
```

Algoritmus: e:N st:Status

nagy, I := 0, hamis

st,e,x : read

st=norm

nagy, I := max(nagy, e), $I \lor e páros$

st,e,x:read

2. 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 súlyából (kg) és hosszából (cm) áll. Keressünk olyan horgászt, aki az 50 cm-esnél hosszabb pontyokból legalább 10 kilogramnyit fogott.

```
Specifikáció:
```

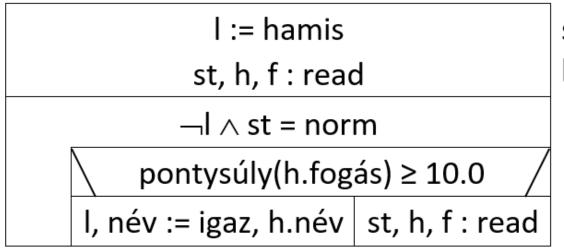
```
A = ( \text{ f:infile}(\mathsf{Horg\acute{a}sz}), \ \mathsf{I:L}, \ \mathsf{n\acute{e}v:S} ) \qquad \qquad \mathsf{Horg\acute{a}sz} = \mathsf{rec}( \ \mathsf{n\acute{e}v:S}, \ \mathsf{fog\acute{a}s:Fog\acute{a}s*} ) \\ \mathsf{Fog\acute{a}s} = \mathsf{rec}( \ \mathsf{id\acute{o}:S}, \ \mathsf{hal:S}, \ \mathsf{s\acute{u}ly:R}, \ \mathsf{hossz:R} ) \\ \mathit{Ef} = ( \ \mathsf{f}=\mathsf{f}_0 ) \\ \mathit{Uf} = ( \ (\mathsf{I}, \ \mathsf{elem}) = \mathbf{SEARCH}_{\mathsf{h}\in\mathsf{f}_0} \ \mathsf{pontys\acute{u}ly}(\mathsf{h.fog\acute{a}s}) \geq 10.0 \ \land \ \mathsf{I} \rightarrow (\mathsf{n\acute{e}v} = \mathsf{elem.n\acute{e}v}) \ ) \\ \mathsf{ahol} \ \mathsf{pontys\acute{u}ly}(\mathsf{h.fog\acute{a}s}) = \sum_{e \in \mathsf{h.fog\acute{a}s}} \mathsf{e.} \ \mathsf{s\acute{u}ly} \\ \mathsf{e.hal="ponty"} \land \mathsf{e.hossz} \geq 50.0 \\ \mathsf{hossz} \geq \mathsf{so.0}
```

Lineáris keresés

t:enor(E) ~ f:infile(Horgász) (st,h,f:read) felt(e) ~ pontysúly(h.fogás)≥10.0

ahol pontysúly : Fogás* $\rightarrow \mathbb{R}$

Algoritmus:



st:Status h:Horgász

Részfeladat: s := pontysúly(x)

$$A = (x:Fogás^*, s:\mathbb{R})$$
 Fogás = rec(idő: \mathbb{S} , hal: \mathbb{S} , súly: \mathbb{R} , hossz: \mathbb{R})

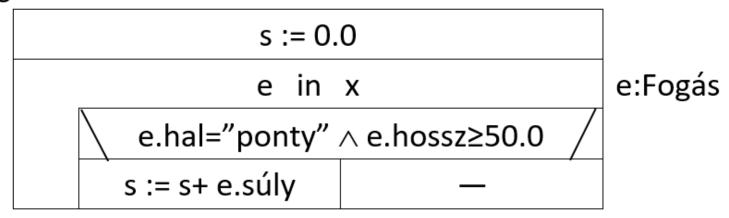
 $Ef = (x=x_0)$
 $Uf = (x=x_0 \land s = \sum_{e \in x_0} e. súly)$
 $e.hal="ponty" \land e.hossz \ge 50.0$

Összegzés (feltételes összegzés)

t:enor(E) ~ e in x
f(e) ~ e.súly
ha e.hal="ponty"

$$\wedge$$
 e.hossz \geq 50.0
H,+,0 ~ \mathbb{R} ,+,0

Algoritmus:



Megjegyzés:

A feltételes összegzés itt is lehetne a fájlból történő olvasás része, azaz az olvasás a horgász nevén kívül csak ezt az összeget adná meg, a teljes fogás listát nem. 3. 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.

1. megoldás

Specifikáció:

```
A = (f:infile(Számla), bevét:\mathbb{N})
Számla=rec(név:\mathbb{S}, lista:Áru*) \qquad \acute{A}ru = rec(cikkszám:\mathbb{S}, \acute{a}r:\mathbb{N})
Ef = (f=f_0)
Uf = (bevét = \sum_{Sz \in f_0} \ddot{o}ssz(sz. lista)) \quad ahol \ddot{o}ssz(sz. lista) = \sum_{e \in sz. lista} e. \acute{a}r
```

```
Összegzés
```

```
t:enor(E) \sim f:infile(Számla) (st,sz,f:read) f(e) \sim össz(sz.lista) ahol össz : Áru*\rightarrow N s \sim bevét H, +, 0 \sim N, +, 0
```

Algoritmus:

bevét := 0

st, sz, f: read

st = norm

bevét := bevét + össz(sz.lista)

st, sz, f: read

st:Status sz:Számla

Részfeladat:

$sum := \ddot{o}ssz(x)$

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

$$Ef = (x=x_0)$$

$$Uf = (sum = \sum_{p \in x_0} p. \acute{a}r)$$

Összegzés

t:enor(E)
$$\sim$$
 p in x

Algoritmus:

$$sum := 0$$

p:Áru

2. megoldás (kihagyható)

```
Specifikáció:

A = (f:infile(\mathbb{N}), bevét:\mathbb{N})

Ef = (f=f_0)
```

$$Uf = (bevét = \sum_{sum \in f_0} sum)$$

Mintha az eredeti szekvenciális inputfájl helyett egy olyan fájlunk lenne, amely csak a számlák végösszegeit tartalmazná: úgy kell olvasni az eredeti állományt, hogy egy olvasás a soron következő sor számlájában szereplő árucikkei árának összegét adja meg.

Összegzés

```
t:enor(E) \sim f:infile(N) (st,sum,f:read)
```

f(e) ~ sum

s ~ bevét

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

Algoritmus:

bevét := 0

st, sum, f: read

st = norm

bevét := bevét + sum

st, sum, f: read

st : Status

 $sum: \mathbb{N}$