- 5. Algoritmus minták felsorolóra
- 1. Válogassuk ki kaktuszok sorozatából egyrészt a piros virágú kaktuszoknak, másrészt a mexikói őshazájú kaktuszoknak neveit!

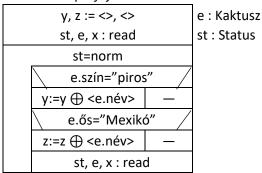
# Specifikáció:

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

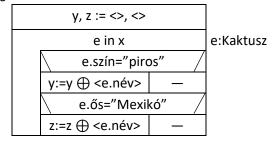
2 összegzés (kiválogatás) közös ciklusba vonva

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

### Szekvenciális inputfájlra:

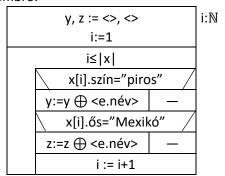


## Algoritmus:



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

### Tömbre:



2. 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 : enor(\mathbb{N}^+), I : \mathbb{L}, m : \mathbb{N})

Ef = (x = x_0 \land |x| \ge 1)

Uf = ((m, _) = MAX_{e \text{ in } x_0} e \land (I, _) = \forall SEARCH_{e \text{ in } x_0} (e \text{ páros}))
```

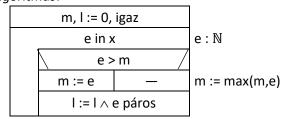
Az utófeltétel másképpen: két összegzéssel

$$Uf = (m = \underline{MAX}_{e \text{ in } x_0} e \land I = \bigwedge_{e \text{ in } x_0} (e \text{ páros}))$$
  
ahol  $\underline{max}: \mathbb{N} \times \mathbb{N} \to \mathbb{N}, \ \underline{max}(a,b) ::= max(a,b)$  neutrális elem: 0  
 $\wedge: \mathbb{L} \times \mathbb{L} \to \mathbb{L}, \quad \wedge (a,b) ::= a \wedge b$  neutrális elem: igaz

## Két összegzés összevonva

H, +, 0 
$$\sim$$
 (N,  $\underline{max}$ , 0), (L,  $\wedge$ , igaz)  
f(e)  $\sim$  e, e páros  
s  $\sim$  m, I

## Algoritmus:



- 3. Adott egész számoknak egy felsorolása.
  - a) Hány páros szám van az első negatív szám előtt?

Specifikáció:

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

$$Ef = (x=x_0)$$

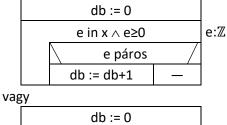
$$Uf = (db = \sum_{e \text{ in } x_0}^{e \ge 0} 1)$$

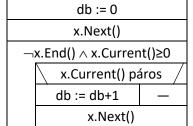
$$e \text{ páros}$$

A∑jobbfelső sarkába írt feltétel mutatja, hogy meddig kell a felsorolást folytatni.

Számlálás, feltétel fennállásáig

Algoritmus:





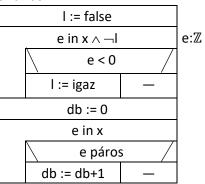
b) Hány páros szám van az első negatív szám után?

Specifikáció:

$$\begin{split} A &= ( \ x:enor(\mathbb{Z}), \ db: \mathbb{N} \ ) \\ Ef &= ( \ x=x_0 ) \\ Uf &= ( \ ( \ \_ \ , \ \_ \ , \ x') = \textbf{SEARCH}_{e \ in \ x_0} \ (e<0) \land \\ & \land \quad db = \sum_{e \ in \ x'} 1) \\ & \quad e \ p\'{a}ros \end{split}$$

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.

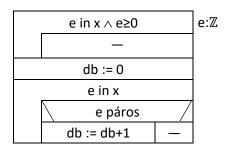
Algoritmus:



Másik megoldás:

$$Uf$$
 = ( ( \_ , x') = SELECT<sub>e in x<sub>0</sub></sub> (e<0  $\lor$  |x|=0)  $\land$  db =  $\sum_{\substack{e \text{ in } x' \\ e \text{ páros}}}$  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').



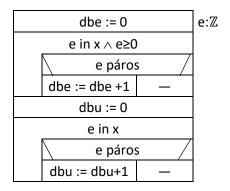
c) Hány páros szám van az első negatív szám előtt, és hány azután?

Specifikáció:

$$A = (x: enor(\mathbb{Z}), dbe, dbu:\mathbb{N})$$
  
 $Ef = (x=x_0)$   
 $Uf = ((dbe, __, x') = \sum_{e \text{ in } x_0}^{e \ge 0} 1 \land dbu = \sum_{e \text{ in } x'} 1)$   
 $e \text{ páros}$   
 $e \text{ páros}$ 

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:



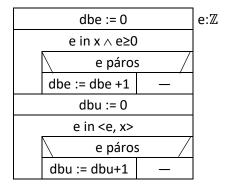
d) Hány páros szám van az első negatív szám előtt, és vele kezdődően hány utána?

Specifikáció:

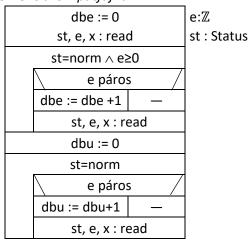
$$\begin{split} A &= (\text{ x:enor}(\mathbb{Z}), \text{ dbe, dbu:} \mathbb{N} \text{ )} \\ Ef &= (\text{ x=x}_0) \\ Uf &= (\text{ (dbe, e', x')} = \sum_{\substack{e \geq 0 \\ e \text{ páros}}} 1 \\ &\quad \text{ } \land \text{ dbu} = \sum_{\substack{e \text{ in } < e' > \oplus x'}} 1) \end{split}$$

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:



Szekvenciális inputfájlra:



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. Egymás 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 vajon minden nap fagypont alatt maradt-e a hőmérséklet, és ezek között (az első fagypont alatti napot is beleértve) mi volt a legalacsonyabb hőmérséklet?

Specifikáció:

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

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

$$Uf = ( (s, e', x') = \sum_{e \text{ in } x_0}^{e \ge 0} (e) \land (db, e', x') = \sum_{e \text{ in } x_0}^{e \ge 0} 1 \land a = s/db \land \land l = \forall \textbf{SEARCH}_{e \text{ in } < e' > \bigoplus x'} (e < 0) \land kicsi = \textbf{MIN}_{e \text{ in } < e' > \bigoplus x'} e )$$

$$\land l = \bigwedge_{e \text{ in } < e' > \bigoplus x'} (e < 0) \land kicsi = \textbf{MIN}_{e \text{ in } < e' > \bigoplus x'} e )$$

Algoritmus:

e, s:ℝ, db:ℕ 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)

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

	-,,		
	s, db := 0.0, 0		
	st, e, x : read		
	st=norm ∧ e≥0		
	s, db := s+e, db +1		
	st, e, x : read		
a := s / db			
	I, kicsi := igaz, e		
	st, e, x : read		
st=norm			
	l := l ∧ e<0		
	e < kicsi		
	kicsi := e —		
	st, e, x : read		

5. 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.

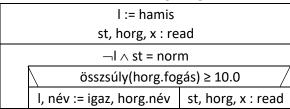
# Specifikáció:

```
A = ( \text{ x:infile(Horgász), l:L, név:S} ) \\ \text{Fogás} = \text{rec}( \text{ név:S, zsákmány:Fogás*}) \\ \text{Fogás} = \text{rec}( \text{ idő:S, fajta:S, hossz:}\mathbb{R}, \text{súly:}\mathbb{R}) \\ Ef = ( \text{ x=x}_0 ) \\ Uf = ( \text{ (I, elem)} = \textbf{SEARCH}_{\text{horg in } \text{x}_0} \text{ összsúly(horg.zsákmány)} \geq 10.0 \\ \wedge \text{ I} \rightarrow (\text{név} = \text{elem.név}) ) \\ \text{ahol összsúly(horg.zsákmány)} = \sum_{\text{hal in horg.zsákmány}} \text{hal. súly ( összsúly : Fogás*} \rightarrow \mathbb{R} ) \\ \text{hal.fajta="ponty"} \\ \wedge \text{ hal.hossz>0.5} \\ \end{cases}
```

#### Lineáris keresés

```
felt(e) ~ összsúly(horg.zsákmány)≥10.0
t:enor(E) ~ x:infile(Horgász) (st,horg,x:read)
```





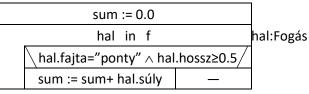
## Részfeladat: sum := pontysúly(x)

```
A = (f:Fogás*, sum:\mathbb{R}) Fogás = rec(idő:\mathbb{S}, fajta:\mathbb{S}, hossz:\mathbb{R}, súly:\mathbb{R}) 
 Ef = (f=f_0) 
 Uf = (f=f_0 \land sum = \sum_{hal \ in \ f_0} hal. \ súly) 
 hal.fajta="ponty" \land hal.hossz \ge 0.5
```

### Összegzés (feltételes összegzés)

```
H,+,0 \sim \mathbb{R},+,0 f(e) \sim hal.súly ha hal.fajta="ponty" \land hal.hossz\geq0.5 t:enor(E) \sim hal in f
```

#### Algoritmus:



# Megjegyzés:

- 1. A fogások fenti összegzése lehetne a horgász (Horgász típusú objektum) egy metódusa is. Ekkor a horgász fogásainak (kezdetben üres) listájához egy másik metódussal 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.

6. 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.

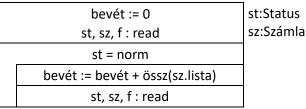
# Specifikáció:

$$A = (f:infile(Számla), bevét:\mathbb{N})$$
 Számla=rec(név: $\mathbb{S}$ , lista:Áru\*) Áru = rec(cikkszám: $\mathbb{S}$ , ár: $\mathbb{N}$ )   
  $Ef = (f=f_0)$   $Uf = (bevét = \sum_{sz in f_0} \ddot{o}ssz(sz. lista))$  ahol  $\ddot{o}ssz(sz. lista) = \sum_{e in sz. lista} e. ár$ 

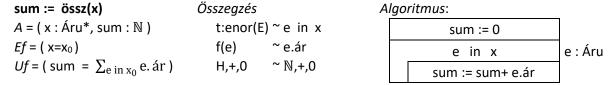
### Összegzés

$$\begin{array}{ccc} \text{H, +, 0} & \sim & \mathbb{N}, +, 0 \\ \text{f(e)} & \sim & \text{\"ossz(sz.lista)} \\ & & \text{ahol \"ossz} : \text{\'Aru*} \rightarrow \mathbb{N} \\ \text{s} & \sim & \text{bev\'et} \\ \text{t:enor(E)} & \sim & \text{f:infile(Sz\'amla)} \text{ (st,sz,f:read)} \end{array}$$

### Algoritmus:



### Részfeladat:



## Megjegyzés:

- 1. A tételek fenti összegzése lehet a számla (Számla 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 (Számla 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.