Bevezetés a programozáshoz 2., 2. zárthelyi dolgozat
2006. március 30. (csütörtök) 8 óra

Specifikáld az alábbi feldatokat, majd add meg a megoldó programokat a visszavezetés módszerével!

1. Feladat (10 pont)
Keressük meg az $[a, b]$ intervallumon értelmezett, egész értékű f függvénynek az első n-nél kisebb vagy 0 értékét!

Specifikáció:
$A = \underset{a}{\mathbb{N}} \times \underset{b}{\mathbb{N}} \times \underset{n}{\mathbb{Z}} \times \underset{i}{\mathbb{N}} \times \underset{l}{\mathbb{L}}$
$B = \underset{a'}{\mathbb{N}} \times \underset{b'}{\mathbb{N}} \times \underset{n'}{\mathbb{Z}}$
$Q = (a = a' \wedge b = b' \wedge a \leq b + 1 \wedge n = n')$
$R = (Q \wedge l = (\exists j \in [a..b] : (f(j) < n \vee f(j) = 0)) \wedge l \to (i \in [a..b] \wedge (f(i) < n \vee f(i) = 0) \wedge \forall j \in [a..i-1] : (f(j) \neq n \wedge f(j) \neq 0)))$

A specifikáció nagyon hasonló a lin. ker. 2.8 programozási tételéhez. Az eltéréseket az alábbi táblázattal foglalhatjuk össze:

feladat		lin. ker. 2.8
a	$\leftrightarrow$	m
b	$\leftrightarrow$	n
$f(i) = 0 \vee f(i) < n$	$\leftrightarrow$	$\beta(i)$

Ez a visszavezetés nem természetes, mert az állapottér bővebb és a helyettesítő táblázatban a $\beta(i)$ helyettesítésekor fel is használjuk ezt a plusz komponenst. Ugyanakkor megjegyezhetjük, hogy a bevezetett n az előfeltétel szerint adott értékű és a program során nem változik (az utófeltételben is szerepel rejtve a $n = n'$). Amennyiben ezek a feltételek teljesülnek egy ilyen kiegészítő komponensre, akkor őt a visszavezetés *paraméterének* nevezzük, mivel ilyenkor a visszavezetés *paraméteres visszavezetés*.

$i, l := a - 1, \text{hamis}$
$\neg l \wedge i \neq b$
> $l := (f(i+1) = 0 \vee f(i+1) < n)$
> $i := i + 1$

2. Feladat (10 pont)
Keressük meg az x vektorban a legnagyobb olyan értéket, amely egyenlő a közvetlen szomszédai átlagával!

Specifikáció:
$\mathbb{V} = \text{vect}(\mathbb{Z}, \mathbb{Z})$
$A = \underset{x}{\mathbb{V}} \times \underset{max}{\mathbb{Z}} \times \underset{l}{\mathbb{L}}$
$B = \underset{x'}{\mathbb{V}}$
$Q = (x = x')$
$R = (Q \wedge l = (\exists j \in [x.lob+1..x.hib-1] : (x_j = \frac{x_{j-1}+x_{j+1}}{2})) \wedge l \to \exists i \in [x.lob+1..x.hib-1] : (x_i = \frac{x_{i-1}+x_{i+1}}{2} \wedge \forall j \in [x.lob+1..x.hib-1] : (x_j = \frac{x_{j-1}+x_{j+1}}{2} \to x_j \leq x_i) \wedge max = x_i))$

feladat		felt. max. ker.
$x.lob + 1$	$\leftrightarrow$	m
$x.hib - 1$	$\leftrightarrow$	n
$x_i = \frac{x_{i-1}+x_{i+1}}{2}$	$\leftrightarrow$	$\beta(i)$

A visszavezetés alteres általánosított.

<table>
<tr><td colspan="5">$k,l := x.lob, \text{hamis}$</td></tr>
<tr><td colspan="5">$k \neq x.hib - 1$</td></tr>
<tr><td rowspan="4"></td><td>$x_{k+1} \neq \frac{x_k + x_{k+2}}{2}$</td><td>$x_{k+1} = \frac{x_k + x_{k+2}}{2} \wedge \neg l$</td><td colspan="2">$x_{k+1} = \frac{x_k + x_{k+2}}{2} \wedge l$</td></tr>
<tr><td rowspan="2">SKIP</td><td rowspan="2">$l, i, max := \text{igaz}, k+1, x_{k+1}$</td><td>$x_{k+1} \geq max$</td><td>$x_{k+1} \leq max$</td></tr>
<tr><td>$i, max := k+1, x_{k+1}$</td><td>SKIP</td></tr>
<tr><td colspan="4">$k := k+1$</td></tr>
</table>

3. Feladat (12 pont)
Adott a kezdőpontja szerint növekvő sorrendben a számegyenes N darab intervalluma. Mutassunk egy olyat, amelyikbe a (szinén adott) k beleesik.
Megoldás lineáris keresés 3.0-val: +2 pont.

Specifikáció:
$Iv = (ah : \mathbb{Z}, fh : \mathbb{Z})$
$I_{Iv}(i) = (i.ah \leq i.fh + 1)$
$\mathbb{V} = \text{vect}(\mathbb{Z}, Iv)$
$A = \underset{v}{\mathbb{V}} \times \underset{k}{\mathbb{Z}} \times \underset{i}{\mathbb{Z}} \times \underset{l}{\mathbb{L}}$
$B = \underset{v'}{\mathbb{V}} \times \underset{k'}{\mathbb{Z}}$
$Q = (v = v' \wedge k = k' \wedge v.hib - v.lob + 1 = N \wedge \forall i \in [v.lob..v.hib - 1] : v_i.ah \leq v_{i+1}.ah)$
$R = (Q \wedge l = (\exists j \in [v.lob, v.hib] : v_j.ah \leq k \leq v_j.fh) \wedge l \leftrightarrow (i \in [v.lob, v.hib] \wedge v_i.ah \leq k \leq v_j.fh))$

feladat		lin. ker. 3.0
l	$\leftrightarrow$	u
g	$\leftrightarrow$	v
$v.lob$	$\leftrightarrow$	m
$v_i.ah \leq k \leq v_i.fh$	$\leftrightarrow$	$\gamma(i)$
$i > v.hib \vee v_i.ah > k$	$\leftrightarrow$	$\delta(i)$

A visszavezetés általánosított (a lin. ker. 3.0 nem csak N hosszú intervallumokra működik), paraméteres (k szerint). Vegyük észre, hogy a Q-ban lévő rendezettségi kikötést is figyelve az R feltétel gyengébb, mint a lin. ker. 3.0-ból helyettesítéssel kapott feltétel, emiatt is általánosított ez a visszavezetés.

<table>
<tr><td colspan="2">$i, l, g := v.lob - 1, \text{hamis}, \text{hamis}$</td></tr>
<tr><td colspan="2">$\neg l \wedge \neg g$</td></tr>
<tr><td rowspan="2"></td><td>$l, g := v_{i+1}.ah \leq i \leq v_{i+1}.fh, i+1 > v.hib \vee v_{i+1}.ah > k$</td></tr>
<tr><td>$i := i+1$</td></tr>
</table>

4. Feladat (14 pont)
Adjuk meg az f függvény értelmezési tartományának azt a leghosszabb szakaszát, amelyen belül az értékek növekevőek! Megoldás rekurzív függvénnyel: +5 pont.

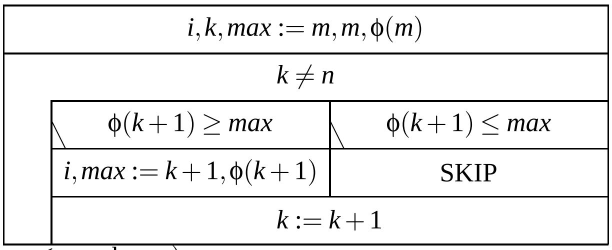
Átfogalmazás:
$R = (\text{kezdoindex} : \mathbb{Z}, \text{hossz} : \mathbb{N}_0)$
$\phi : [m, n] \rightarrow R$
$\phi(m) = (m, 1)$, $\forall i \in [m+1, n] : \phi(i) = F(i, \phi(i-1))$, ahol $F : [m+1, n] \times R \rightarrow R$ definíciója:

$$F(i,x) = \begin{cases} (x.\text{kezdoindex}, x.\text{hossz} + 1) & \text{, ha } f_{i-1} \leq f_i \\ (i, 1) & \text{, egyébként} \end{cases}$$

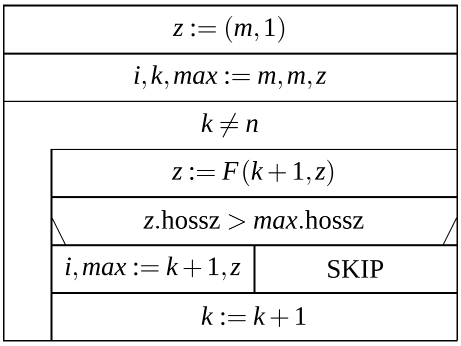
Ezzel a ϕ függvénnyel a feladat a maximum keresésre vezethető vissza. A ϕ fv. maximumát keressük, az R típus rendezettségét annak hossz komponense adja. Az eredményszakasz kezdőindexét a megtalált *max* érték kezdoindex nevű komponensében szolgáltatja a program, míg a növekvő rész utolsó értéke a kezdoindex+hossz-1 képlettel számolható.

feladat		max. ker
$\phi(i)$	$\leftrightarrow$	$f(i)$

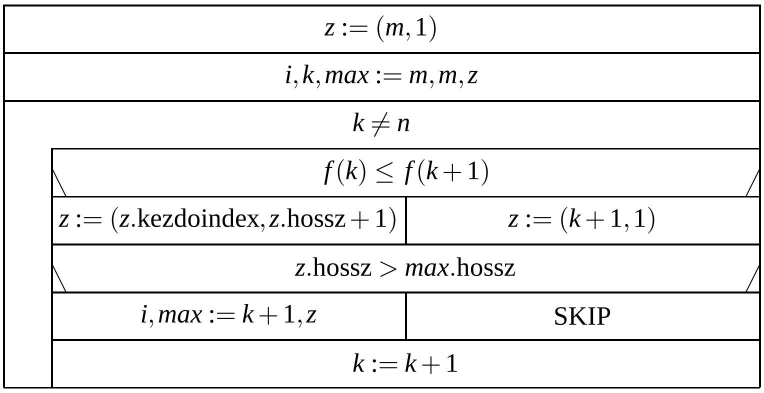
Specifikáció:
$A = \underset{m}{\mathbb{Z}} \times \underset{n}{\mathbb{Z}} \times \underset{max}{R}$
$B = \underset{m'}{\mathbb{Z}} \times \underset{n'}{\mathbb{Z}}$
$Q = (m = m' \land n = n' \land m \leq n)$
$R = (Q \land \exists i \in [m,n] : \phi(i) = max \land \forall j \in [m,n] : \phi(j).\text{hossz} \leq max.\text{hossz})$



Felhasználva a rekurzív függvény változóval való helyettesítésének programtranszformációs módszerét ϕ-re és azt a tényt, hogy két R-beli elemet a hossz komponens alapján hasonlítunk össze:



Ha ismerjük az esetszétválasztással definiált függvény kiszámításának programozási tételét, akkor ebből megkaphatjuk a megoldó programot, ha azt F-re alkalmazzuk:

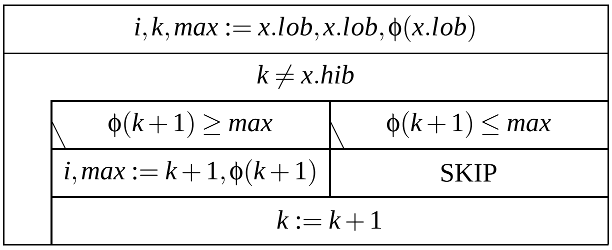


5. *Feladat* (14 pont)
Adott az x vektor, amelynek elemei karakterek. A vektor szavakat tartalmaz, amiket egy-egy vessző választ el egymástól. Adjuk meg a leghosszabb szónak a kezdőindexét!
Megoldás rekurzív függvénnyel: +5 pont.

Átfogalmazás:
$R = (\text{kezdoindex} : \mathbb{Z}, \text{hossz} : \mathbb{N}_0)$
$\phi : [x.lob-1, x.hib] \to R$
$\phi(x.lob-1) = (x.lob, 0)$, $\forall i \in [x.lob, x.hib] : \phi(i) = F(i, \phi(i-1))$, ahol $F : [x.lob, x.hib] \times R \to R$ definíciója:

$$F(i,y) = \begin{cases} (y.\text{kezdoindex}, y.\text{hossz}+1) & \text{, ha } x_i \neq ',' \\ (i+1, 0) & \text{, ha } x_i = ',' \end{cases}$$

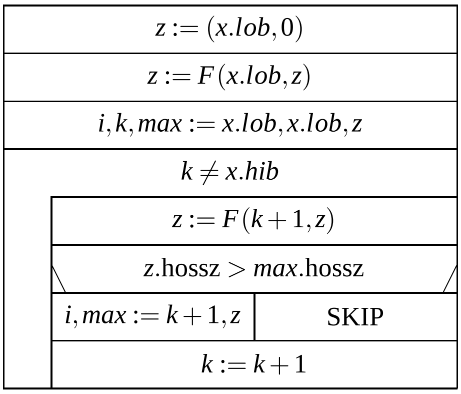
Ezzel a ϕ függvénnyel a feladat a maximum keresésre vezethető vissza. A ϕ fv. maximumát keressük, az R típus rendezettségét annak hossz komponense adja. Az eredményt a megtalált *max* érték kezdoindex nevű komponensében szolgáltatja a program. (Megjegyzés: ha a feltételes maximumkeresésre vezetnénk vissza a feladatot, akkor egy bonyolultabb stuktogram árán egy sokkal kevesebb összehasonlítást végző programot kapnánk.)



Specifikáció:
$\mathbb{V} = \text{vect}(\mathbb{Z}, \text{Ch})$
$A = \underset{x}{\mathbb{V}} \times \underset{max}{R}$
$B = \underset{x'}{\mathbb{V}}$
$Q = (x = x' \land x.dom > 0)$
$R = (Q \land \exists i \in [x.lob, x.hib] : \phi(i) = max \land \forall j \in [x.lob, x.hib] : \phi(j).\text{hossz} \leq max.\text{hossz})$

feladat		max. ker
$x.lob$	$\leftrightarrow$	m
$x.hib$	$\leftrightarrow$	n
$\phi(i)$	$\leftrightarrow$	$f(i)$

Felhasználva a rekurzív függvény változóval való helyettesítésének programtranszformációs módszerét ϕ-re és azt a tényt, hogy két R-beli elemet a hossz komponens alapján hasonlítunk össze:



Ha ismerjük az esetszétválasztással definiált függvény kiszámításának programozási tételét, akkor ebből megkaphatjuk a megoldó programot, ha azt F-re alkalmazzuk: