3. táblás gyakorlat – visszavezetés

6/b. Határozzuk meg két pozitív egész szám legnagyobb közös osztóját. Vezessük vissza feltételes maximumkeresésre.

$$A = (a : \mathbb{N}^+, b : \mathbb{N}^+, o : \mathbb{N}^+, l : \mathbb{L})$$

$$ef = (a = a' \wedge b = b')$$

$$uf = \left(ef \wedge (l, o) = \begin{matrix} \min(a,b) \\ \mathrm{MAX}_{1,2} \quad i \\ i = 1 \\ i|a \wedge i|b \end{matrix} \right)$$

Visszavezetés (feltételes maximumkeresés):

H	~	$\mathbb{N}^+$
[m..n]	~	[1..min(a,b)]
max	~	o
f(i)	~	i
β(i)	~	i|a és i|b
ind		elhagytuk

Struktogram:

<table>
<tr><td colspan="4">$l :=\downarrow$</td></tr>
<tr><td colspan="4">$i := 1..\min(a,b)$</td></tr>
<tr><td>$\neg(\mathrm{i|a} \wedge \mathrm{i|b})$</td><td colspan="2">$\mathrm{i|a} \wedge \mathrm{i|b} \wedge l$</td><td>$\mathrm{i|a} \wedge \mathrm{i|b} \wedge \neg l$</td></tr>
<tr><td rowspan="2">$SKIP$</td><td colspan="2">$i > o$</td><td rowspan="2">$l, o, :=\uparrow, i$</td></tr>
<tr><td>$o := i$</td><td>$SKIP$</td></tr>
</table>

$i : \mathbb{N}$

A pirossal megjelölt kifejezés nem megengedett, se a specifikációban se a stukiban, ezért definiáljuk:

$$min : \mathbb{N}^+ \times \mathbb{N}^+ \to \mathbb{N}^+, \text{ úgy hogy:}$$

$$\min(a,b) = \begin{cases} a, ha\ a \leqq b \\ b, különben \end{cases}$$

Innentől a specifikáció már teljesen korrekt, de a struktogram még mindig nem. A gond az, hogy én tételekkel és transzformációkkal mindig nem megengedett utasításokat tudok helyettesíteni, min(a,b) pedig nem egy *utasítás*, hanem egy *kifejezés*.

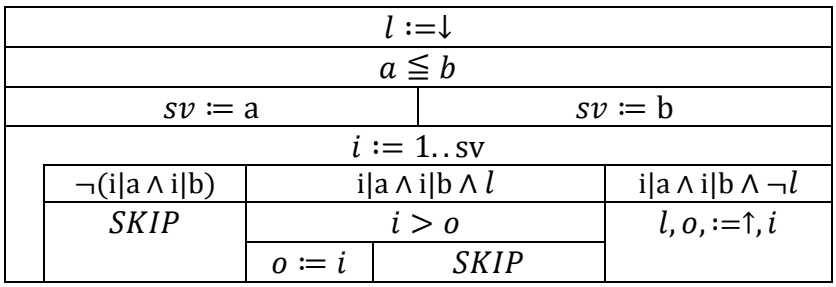
Alakítsuk hát át a programot, a *függvény helyettesítése változóval* transzformáció segítségével:

<table>
<tr><td colspan="4">$l :=\downarrow$</td></tr>
<tr><td colspan="4">$sv := \min(a,b)$</td></tr>
<tr><td colspan="4">$i := 1..\mathrm{sv}$</td></tr>
<tr><td>$\neg(\mathrm{i|a} \wedge \mathrm{i|b})$</td><td colspan="2">$\mathrm{i|a} \wedge \mathrm{i|b} \wedge l$</td><td>$\mathrm{i|a} \wedge \mathrm{i|b} \wedge \neg l$</td></tr>
<tr><td rowspan="2">$SKIP$</td><td colspan="2">$i > o$</td><td rowspan="2">$l, o, :=\uparrow, i$</td></tr>
<tr><td>$o := i$</td><td>$SKIP$</td></tr>
</table>

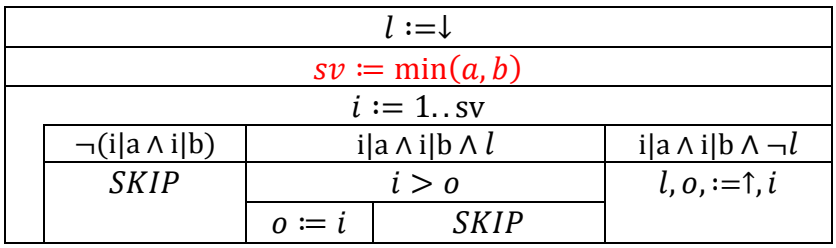
$sv : \mathbb{N}^+$

$i : \mathbb{N}$

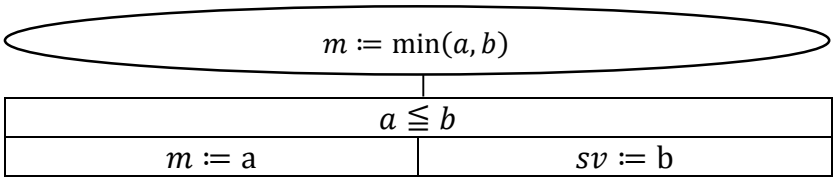
Ezzel a nem megengedett kifejezést elcseréltük egy nem megengedett értékadásra, amit már az *esetszétválasztással definiált függvény kibontása* tétellel könnyedén el tudunk tüntetni:



Bár erről csak egy későbbi órán lesz részletesebben szó, de ezt a problémát kissé eltérő módon is lehet kezelni. Képzeljük el azt, hogy a transzformációkkal létrehozott új kódrészlet nagyon bonyolult, esetleg a struktogramban több helyen is megjelenik ugyanaz. Ekkor célszerűbb inkább egy függvényt[1] írni rá, és ezt a függvényt meghívni, hiszen úgy sokkal átláthatóbb, és bár több egység lesz, de egy egység rövidebb, érthetőbb lesz. Valamint a kódismétlést és az abból fakadó hibalehetőségeket is elkerültük így. A kódolás során már eddig is sokszor csináltunk ilyesmit, de a struktogram szintjén is szabad!



És definiálunk a nem megengedett értékadáshoz is egy azt megoldó alprogramot:



Tehát ez a kis alprogram annyit tesz, hogy a paraméterül kapott a és b (*formális paraméterek*) közül visszaadja a return értékként kezelt m nevű változóba a kisebbet.

Ezt a főprogramban meghívtuk a és b *aktuális paraméterek*kel, amelyek nevei véletlen egybeesnek a formálisakéval, de ez nem kötelező, és mivel az sv-nek adjuk értékül, ezért az alprogramban m-mel jelölt formális kimenő változó, a hívás helyén az sv változó lesz. Azaz sv-be bekerül a kisebb szám értéke, épp úgy, mintha ezt az elágazást „beégettem” volna az eredeti stuktogramba.

[1] most nem specifikációs értelemben, mint a fenti esetszétválasztásos függvény, hanem „alprogram” értelemben, mint egy C++-os függvény… ami persze nem lesz más, mint a specifikációban megadott függvény mint feladat megoldása.