Uvažujte jednoduchý embedded systém, ve kterém je procesor se třemi registry (dva obecné, jeden čítač aktuální adresy) a virtuálně nekonečný počet buněk paměti. V jedné buňce může být zapsána instrukce (komplet, včetně operandů) nebo číslo, v jednom registru může být zapsáno jen číslo (N označuje přirozená čísla):

$$\begin{aligned} \mathit{Cell} \rightarrow & N \mid \\ & \text{HALT} \mid \\ & \text{CONST } \mathit{Reg}, N \mid \\ & \text{LOAD } \mathit{Reg}, N \mid \\ & \text{STORE } \mathit{Reg}, N \mid \\ & \text{ADD} \mid \\ & \text{SUBTR} \mid \\ & \text{COND } N \\ \mathit{Reg} \rightarrow & \text{A} \mid \text{B} \end{aligned}$$

kde N reprezentuje přirozená čísla.

Konfigurace operační sémantiky tohoto stroje je čtveřice $N \times N \times N \times (N \rightarrow \mathit{Cell})$, kde první složka reprezentuje obsah obecného registru A, druhá obsah obecného registru B, třetí obsah čítače aktuální adresy a čtvrtá složka reprezentuje paměť jako funkci z indexu do obsahu buňky na daném indexu. Nad množinou všech konfigurací je nadefinována přepisovací relace $\rightsquigarrow$ ($a, b, i, m, n, n' \in N, \mathit{mem} \in N \rightarrow \mathit{Cell}$):

$$\frac{\mathit{mem}(i) = \text{CONST A}, n}{(a, b, i, \mathit{mem}) \rightsquigarrow (n, b, i+1, \mathit{mem})}$$

$$\frac{\mathit{mem}(i) = \text{CONST B}, n}{(a, b, i, \mathit{mem}) \rightsquigarrow (a, n, i+1, \mathit{mem})}$$

$$\frac{\mathit{mem}(i) = \text{LOAD A}, n \quad \mathit{mem}(n) = n'}{(a, b, i, \mathit{mem}) \rightsquigarrow (n', b, i+1, \mathit{mem})}$$

$$\frac{\mathit{mem}(i) = \text{LOAD B}, n \quad \mathit{mem}(n) = n'}{(a, b, i, \mathit{mem}) \rightsquigarrow (a, n', i+1, \mathit{mem})}$$

$$\frac{\mathit{mem}(i) = \text{STORE A}, n}{(a, b, i, \mathit{mem}) \rightsquigarrow (a, b, i+1, \lambda m.\text{if } m = n \text{ then } a \text{ else } \mathit{mem}(m) \text{ fi})}$$

$$\frac{\mathit{mem}(i) = \text{STORE B}, n}{(a, b, i, \mathit{mem}) \rightsquigarrow (a, b, i+1, \lambda m.\text{if } m = n \text{ then } b \text{ else } \mathit{mem}(m) \text{ fi})}$$

$$\frac{\mathit{mem}(i) = \text{ADD}}{(a, b, i, \mathit{mem}) \rightsquigarrow (a+b, 0, i+1, \mathit{mem})}$$

$$\frac{\mathit{mem}(i) = \text{SUBTR} \quad a > b}{(a, b, i, \mathit{mem}) \rightsquigarrow (a-b, 0, i+1, \mathit{mem})}$$

$$\frac{\mathit{mem}(i) = \text{SUBTR} \quad a \leq b}{(a, b, i, \mathit{mem}) \rightsquigarrow (0, 0, i+1, \mathit{mem})}$$

$$\frac{mem(i) = \text{COND } n \quad a = b}{(a, b, i, mem) \rightsquigarrow (a, b, n, mem)}$$
$$\frac{mem(i) = \text{COND } n \quad a \neq b}{(a, b, i, mem) \rightsquigarrow (a, b, i + 1, mem)}$$

Množina finálních konfigurací je $\{(a, b, i, mem) \in N \times N \times N \times (N \to Cell) | mem(i) = \text{HALT}\}$, vstupní funkce *input* je definována jako $input(a, b, mem) = (a, b, 0, mem)$, výstupní funkce *output* je definována jako $output((a, b, i, mem)) = (a, b)$.

1. Rozepište všechny možnosti (tranzitivního) přepsání konfigurace $(0, 0, 0, \lambda n.\text{if } n < 3 \text{ then CONST A, 3 else 3 fi})$.

$$(0, 0, 0, \lambda n.\text{if } n < 3 \text{ then CONST A, 3 else 3 fi}) \rightsquigarrow$$
$$(3, 0, 1, \lambda n.\text{if } n < 3 \text{ then CONST A, 3 else 3 fi}) \rightsquigarrow$$
$$(3, 0, 2, \lambda n.\text{if } n < 3 \text{ then CONST A, 3 else 3 fi}) \rightsquigarrow$$
$$(3, 0, 3, \lambda n.\text{if } n < 3 \text{ then CONST A, 3 else 3 fi})$$

2. Nadefinujte instrukce CALL n a RETURN, které provedou volání a návrat z podprogramu a použijí k tomu registr B jako ukazatel na vrchol zásobníku. Instrukce CALL n by měla i) uložit aktuální adresu na adresu uloženou v registru B, ii) zvýšit hodnotu registru B o jedna a iii) skočit na adresu n, zatímco instrukce RETURN by měla i) snížit hodnotu registru B o jedna a ii) skočit na adresu, jejíž adresa je uložená v buňce odkazované registrem B.

$$\frac{mem(i) = \text{CALL } n}{(a, b, i, mem) \rightsquigarrow (a, b + 1, n, \lambda m.\text{if } m = b \text{ then } i \text{ else } mem(m))}$$
$$\frac{mem(i) = \text{RETURN}}{(a, b, i, mem) \rightsquigarrow (a, b - 1, mem(b - 1), mem)}$$

3. Dá se o každém programu v tomto jazyce říct, že je terminující? Svoji odpověď zdůvodněte.

   Ne. Je možné v něm napsat nekonečný cyklus.