

Cvičení 10

Příklad 1: Zjistěte, co dělá následující stroj RAM:

```

LOAD    =1
STORE   3
READ
loop:   JZERO  output
        STORE  2
        LOAD   3
        ADD    0
        STORE  3
        LOAD   2
        SUB    =1
        JUMP   loop
output: LOAD   3
        WRITE
        HALT

```

Příklad 2: Pro každý z následujících problémů navrhnete stroj RAM, který ho řeší.

Poznámka: Při konstrukci stroje nemusíte řešit chybná data na vstupu, která neodpovídají zadání.

a) VSTUP: celá čísla x, y (tj. $x, y \in \mathbb{Z}$)

VÝSTUP: hodnota $x + y$

b) VSTUP: celá čísla x, y (tj. $x, y \in \mathbb{Z}$)

VÝSTUP: $\max\{x, y\}$

c) VSTUP: přirozené číslo n (tj. $n \in \mathbb{N}$)

VÝSTUP: sekvence čísel $1, 2, \dots, n$

Poznámka: Pro $n = 0$ bude sekvence na výstupu prázdná.

d) VSTUP: sekvence čísel $a_1, a_2, \dots, a_n, 0$, kde $n \geq 0$ a $a_i \in \mathbb{Z} - \{0\}$ pro $1 \leq i \leq n$

VÝSTUP: $\prod_{i=1}^n a_i$

Poznámka: Pro $n = 0$ bude výstupem hodnota 1.

e) VSTUP: sekvence čísel $a_1, a_2, \dots, a_n, 0$, kde $n \geq 0$ a $a_i \in \mathbb{Z} - \{0\}$ pro $1 \leq i \leq n$

VÝSTUP: sekvence čísel a_n, a_{n-1}, \dots, a_1

Příklad 3: Sestavte program pro stroj RAM, který přečte ze vstupu číslo n a vypíše na výstup n -té Fibonacciho číslo F_n . Můžete předpokládat, že číslo n na vstupu je nezáporné (tj. nemusíte řešit situaci, kdy $n < 0$). Připomeňme, že Fibonacciho čísla F_0, F_1, F_2, \dots jsou

definována následujícím rekurentním vztahem:

$$F_n = \begin{cases} 0 & \text{pro } n = 0 \\ 1 & \text{pro } n = 1 \\ F_{n-1} + F_{n-2} & \text{pro } n > 1 \end{cases}$$

Příklad 4: Sestavte program pro stroj RAM, který přečte ze vstupu dvě čísla x a k a na výstup vypíše hodnotu k -tého bitu čísla x (tj. 0 nebo 1), přičemž bity jsou číslovány od 0 a 0-tý bit je nejméně významný bit. Můžete předpokládat, že $x \geq 0$ a $k \geq 0$ (tj. nemusíte řešit situace, kdy $x < 0$ nebo $k < 0$).

Příklad 5: Navrhněte program pro stroj RAM, který načte ze vstupu dvě čísla x a y (můžete předpokládat, že $x \geq 0$ a $y \geq 0$) a na výstup vypíše jejich součin $x \cdot y$. Aby to nebylo tak jednoduché, musíte navíc dodržet následující omezení:

- Ve vašem programu *nesmíte* použít instrukce MUL a DIV. Můžete ovšem použít novou speciální instrukci RSHIFT, která má stejný význam jako DIV =2.
- Celkový počet instrukcí, který váš program provede, musí být polynomiální vzhledem k počtům bitů nutných pro zápis čísel x a y .