1. **RAM** Definice RAMu + rozmyšlení problémů s různými definicemi velikosti buňky.

2. **programy v RAMu** Co dělají následující funkce?

```
f(x,y):
        if x==0 => return y
        else => return f((x&y) << 1, x^y)

g(x,y):
        if y==0 => return 0
        else if even(y) => return 2*g(x, y/2)
        else => return 2*g(x, y/2) + x

h(x,y):
        if x<y => return (0,x)
        else:
                (a,b) <- 2*h(x/2, y)
                if odd(x) => b <- b+1
                if b>=y => a <- a+1, b <- b-y
                return (a,b)
```

3. **Velká čísla** Mějme RAM s neomezenou velikosti čísel. Vymyslete, jak zakódovat libovolné množství celých čísel $c_1, \ldots, c_n$ do jednoho celého čísla C tak, aby se jednotlivá čísla c_i dala jednoznačně dekódovat.

4. **Co dělá?** Vstupem následujícího alogritmu je přirozené číslo N zapsané v buňce `[0]`. Určete, co je jeho výstupem a spočtěte přesně počet provedených instrukcí vůči N.

```
I := 1
Z := 1
VNEJSI: if I > [0] then halt
J := 1
I := I + 1
VNITRNI: if J > I then goto VNEJSI
[Z] := 1
Z := Z + 1
J := J + 1
goto VNITRNI
```

5. **Složitost** Jaká bude přesná doba běhu programu výše, pokud zavedeme logaritmickou, případně poměrnou logaritmickou cenu instrukce?

6. **Prohození obsahu** Vymyslete, jak na RAMu prohodit obsah dvou paměťových buněk, aniž byste použili jakoukoliv jinou buňku.

7. **Neomezená kapacita II** Navrhněte postup, jak v případě neomezené kapacity paměťové buňky pozměnit libovolný program na RAMu tak, aby používal vždy jen konstantně mnoho paměťových buněk. Program můžete libovolně zpomalit. Kolik nejméně buněk je potřeba?

1. **Jak na RAMu? (8 bodů)** Rozmyslete si, jak do instrukcí RAMu překládat konstrukce známé z vyšších programovacích jazyků: podmínky, cykly, volání podprogramů s lokálními proměnnými a rekurzi.