KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI
www.inf.upol.cz

Michal Krupka
krupka.inf.upol.cz
michal.krupka@upol.cz
17. listopadu 12, 771 46 Olomouc

Paradigmata programování 2 ◇ poznámky k přednášce

8. Přísliby a proudy

verze z 19. května 2019

Text k této přednášce není k dispozici. Podívejte se na zdrojové soubory `08_streams.lisp` a `08_generators.lisp`.

Na přednášce jsem říkal, že pomocí proudů se vždy ušetří paměť oproti obyčejným seznamům. To ale obecně není pravda, jen v některých situacích (viz zdrojový kód `08_streams.lisp`). Proto jsem přidal také ukázku jiného přístupu k řešení téhož problému, a to pomocí **generátorů**. Stručné vysvětlení najdete v souboru `08_generators.lisp`.

Otázky a úkoly na cvičení

1. Napište funkci `stream-ref`, která k zadanému proudu a indexu vrátí příslušný prvek proudu:

```
(stream-ref (stream 'a 'b 'c 'd) 2) => c
```

2. Napište funkci `stream-each-nth`, která k zadanému proudu vrátí proud obsahující každý `n`-tý prvek původního proudu:

```
CL-USER 1 > (stream-to-list
              (stream-each-nth
                3
                (stream 'a 'b 'c 'd 'e 'f 'g 'h 'i 'j 'k 'l)))
(a d g j)
```

3. Napište funkci `stream-append`, která spojí dva zadané proudy do jednoho:

```
CL-USER 1 > (stream-to-list
              (stream-append (stream 1 2 3)
                             (stream 4 5 6)))
(1 2 3 4 5 6)
```

4. Vylepšete předchozí příklad tak, aby se druhý argument nevyhodnocoval, dokud to nebude opravdu potřeba. Operátor `stream-append` budete muset napsat jako makro.

5. Napište funkci `primes`, která vrátí proud všech prvočísel vytvářených pomocí Eratosthenova síta.

6. Napište funkci `primes` tak, aby místo proudu vytvořila generátor.

7. Zobecněte funkci `stream-map` na víc argumentů a vyzkoušejte příklad na funkci `fib2` ze zdrojového kódu k přednášce.

8. Přepište funkci `stream` na makro, které nebude vyhodnocovat své argumenty, dokud to nebude potřeba.