TÖL304G
Forritunarmál
Verkefnablað 2

Snorri Agnarsson

31. ágúst 2022

Inngangur

Þið eigið að prófa öll verkefnin ykkar í einhverju Scheme kerfi, til dæmis DrRacket.

Hér eru nokkur Scheme föll og lykilorð sem mögulegt er að þið viljið nota til að leysa þessi verkefni: define, lambda, if, and, or, car, cdr, cons, null?, list, =, *, +.

Athugið að Scheme report[1] inniheldur lýsingar á öllum þessum föllum og lykilorðum og einnig má finna nákvæma skjölun í fylgigögnum fyrir MIT-Scheme og DrRacket.

Hópverkefni

1. Skrifið Scheme fall `last` sem tekur lista sem viðfang, sem ekki má vera tómur, og skilar aftasta gildinu í listanum. Til dæmis skulu segðirnar `(last '(1 2 3))` og `(last (list 1 2 3))` skila `3`.

```
;; Notkun: (last x)
;; Fyrir:  x=(x1 x2 ... xN) er listi, ekki tómur.
;; Gildi:  xN, þ.e. aftasta gildi x.
(define (last x)
   ...
}
```

[1] http://www.hi.is/ snorri/downloads/r5rs.pdf

2. Skrifið Scheme fall `remove-last` sem tekur lista sem viðfang, sem ekki má vera tómur, og skilar lista allra gilda nema aftasta í viðfangslistanum. Til dæmis skal segðin `(remove-last '(1 2 3))` skila `(1 2)`.

```
;; Notkun: (remove-last x)
;; Fyrir:  x=(x1 x2 ... xN) er listi, ekki tómur.
;; Gildi:  (x1 x2 ... xN-1), þ.e. listi allra
;;         gilda í x fyrir utan aftasta.
(define (remove-last x)
   ...
)
```

3. Skrifið Scheme fall `product`, sem tekur eitt viðfang x, sem skal vera listi talna $x_1, \ldots, x_n$, og skilar $\Pi_{i=1}^{n} x_i$. Þið skuluð leyfa að listinn sé tómur og skila viðeigandi gildi í því tilviki. Fallið skal vera halaendurkvæmt.[2]

```
;; Notkun: (product x)
;; Fyrir:  x=(x1 x2 ... xN) er listi talna.
;; Gildi:  Talan x1*x2*...*xN.
(define (product x)
   ...
)
```

4. Skrifið Scheme fall `myappend` sem tekur tvo lista, `x` og `y` sem viðföng, og skilar lista sem inniheldur fremst öll gildin úr `x` (í sömu röð og í `x`) og síðan öll gildin úr `y` (í sömu röð og í `y`). Fallið `myappend` skal útfæra með því að nota föllin `last` og `remove-last`, að ofan, og einnig má nota `define`, `if`, `cons`, `null?`, en ekki önnur föll eða lykilorð. Fallið `myappend` verður eðlilega halaendurkvæmt, þannig að oftast þegar kallað er á það mun það enda á að kalla á sjálft sig. Tímaflækja þessa falls er hins vegar ekkert til að hrópa húrra fyrir. Athugið líka að þótt `myappend` sé halaendurkvæmt þá er næstum öruggt að hjálparfallið `remove-last` verður trúlega ekki halaendurkvæmt þannig að heildarlausnin er þá ekki halaendurkvæm.

```
;; Notkun: (myappend x y)
;; Fyrir:  x=(x1 x2 ... xN) er listi,
;;         y=(y1 y2 ... yM) er einnig listi.
;; Gildi:  Listinn (x1 x2 ... xN y1 y2 ... yM).
(define (myappend x)
   ...
)
```

[2] Það dugar að útreikningarnir séu framkvæmdir af halaendurkvæmu hjálparfalli, jafnvel þótt `product` sé ekki beint halaendurkvæmt. Markmiðið er að takmarka dýpt hlaðans sem forritið notar fyrir milliniðurstöður, sem við munum sjá að er hlaði svokallaðra vakningarfærslna (*activation records*).

1 Einstaklingsverkefni

1. Skrifið Scheme fall `sum`, sem tekur eitt viðfang x, sem skal vera listi talna $x_1, \ldots, x_n$, og skilar $\sum_{i=1}^{n} x_i$. Þið skuluð leyfa að listinn sé tómur og skila viðeigandi gildi í því tilviki.

```
;; Notkun: (sum x)
;; Fyrir:  x=(x1 x2 ... xN) er listi talna.
;; Gildi:  Talan x1+x2+...+xN.
(define (sum x)
   ...
}
```

2. Skrifið halaendurkvæmt Scheme fall `squaresum` sem tekur eitt viðfang x þar sem x er listi talna $x_1, \ldots, x_n$, og skilar summu kvaðrata talnanna. Til dæmis skal Scheme segðin `(squaresum (list 1 2 3))` skila gildinu á $1^2+2^2+3^2$. Lausnin verður að vera halaendurkvæm, annars fáið þið einungis hálf stig. Einnig skal lausnin skila vitrænu gildi þegar listinn er tómur. Athugið að til þess að skrifa halaendurkvæma lausn er eðlilegt að nota hjálparfall sem tekur tvö viðföng.

```
;; Notkun: (squaresum x)
;; Fyrir:  x=(x1 x2 ... xN) er listi talna.
;; Gildi:  Talan x1*x1+x2*x2+...+xN*xN.
(define (squaresum x)
   ...
}
```

3. Skrifið Scheme fall `incall`, sem tekur tölu y sem viðfang og skilar falli, sem tekur lista talna, $(x_1 \ldots x_n)$, sem viðfang, og skilar listanum $(y + x_1 \ldots y + x_n)$. Til dæmis skal `((incall 2) '(1 2 3))` skila listanum `(3 4 5)`.

```
;; Notkun: ((incall y) x)
;; Fyrir:  y er tala, x=(x1 x2 ... xN)
;;         er listi talna.
;; Gildi:  Talnalistinn (x1+y x2+y ... xN+y).
(define (incall y)
   ...
}
```