582305 Symbolinen ohjelmointi
4. harjoitus, 17.10.2002

Tehtävä 4.1: Ohjelmoi Scheme-konversiofunktio `luku->numerot` joka muuntaa syötteenä saamansa luonnollisen luvun listaksi sen (kymmenkantaisista) numeroista. Silloin esimerkiksi kutsu `(luku->numerot 1024)` = `(1 0 2 4)`.

Ohjelmoi myös käänteinen Scheme-konversiofunktio `numerot->luku` joka muuntaa tällaisen numerolistan takaisin luonnolliseksi luvuksi. Esimerkiksi `(numerot->luku '(1 0 2 4))` = 1024.

Käsittele kummassakin funktiossa kaikki virheelliset syötteet siten, että tuloksena on `#f`.

Tehtävä 4.2: Luonnollinen luku on palindromi, jos se on sama luettuna etu- ja takaperin. (Etunollat unohdetaan luettaessa, kuten tavallista.) Esimerkiksi luku 121 on palindromi. Yksi tapa tuottaa palindromiluku on lähteä liikkeelle annetusta siemenluvusta n, ja toistaa toimitusta "laske yhteen n ja se luku jonka saat lukemalla n takaperin". Esimerkiksi luvusta 19 saadaan palindromi toimituksilla $19 + 91 = 110$ ja $110 + 11 = 121$.

Ohjelmoi tätä algoritmia käyttävä Scheme-kielinen palindromigeneraattori, joka ottaa syötteenään siemenluvun n. Voit käyttää apuna tehtävän 4.1 konversiofunktioita. Käsittele myös virheelliset syötteet sen tehtävän tavoin.

Löydätkö sellaisia siemenlukuja n joilla näin tuotettu palindromi on paljon siemenlukuaan suurempi?

Tehtävä 4.3: Kirjoita Scheme-funktio `sano`, joka "puhuu" syötteenään saamansa positiivisen kokonaisluvun "tavuttain", eli listana tavuja joista jokainen esitetään symbolina.

Esimerkiksi `(sano 1812)` = `(tu hat kah dek san sa taa kak si tois ta)`.

Yritä ohjelmoida funktiosi siten, että yhä suurempien lukujen kuten "miljardi" lisäämiseksi sen sanavarastoon riittää muokata sen käyttämiä vakioita eikä suoritettavaan ohjelmakoodiin tarvitse koskea.

Tehtävä 4.4: Toteuta tehtävän 3.2 lukulistalomittaja laiskoille listoille.

Tehtävä 4.5: Toteuta laiskoille listoille `lazy-map`, eli se sama funktio `map`, joka löytyy tavallisille listoille Scheme-kielen standardikirjastosta. Voit rajoittua tapaukseen jossa argumentteja on vain kaksi: yksi proseduuri ja yksi laiska lista.

Selitä mitä seuraava Scheme-määritelmä esittää ja miksi.

```
(define myst (cons 1 (delay (lazy-map (lambda (x) (+ x 1)) myst)
```

Tehtävä 4.6: *Hammingin* lukujono koostuu luvuista $2^i \cdot 3^j \cdot 5^k$, missä $i, j, k \in \mathbb{N}$, aidosti kasvavassa järjestyksessä. Määrittele tämä jono laiskana Scheme-listana.

Vihje: Tarvitset vain tehtäviä 4.4 ja 4.5.

Miten ratkaisisit tämän tehtävän ilman laiskuutta? (Hahmotelma riittää, ei tarvitse toteuttaa.) Kumpaa ratkaisutapaa pitäisit selvempänä? Kumpaa tehokkaampana?

(Tehtäviä yhteensä 6 kpl.)