Liidesed

Idrisis tuleb tihti kirjutada funktsioone, mis erinevad ainult natuke tüübi poolest:

```
equalChar   :   Char →   Char → Bool
equalInt    :    Int →    Int → Bool
equalString : String → String → Bool
```

Sellist koodi *ei saa* kirjutada ühe puhta parameetrilise polümorfse funktsiooniga, kuna funktsioonide implementatsioon on erinev:

```
equal : a → a → Bool
equal = ?eq_v  -- pole def. mis töötaks iga tüübi korral
```

Selleks saame teha liidese (Haskellis tüübiklassi)

```
interface Equal a where
    (==) : a → a → Bool

Equal Int where
    a == b = ?eq_Int    -- Int'ide võrdsus
Equal Bool where
    a == b = ?eq_Bool   -- Bool'ide võrdsus
```

Tüübiklassid standardteegis

Idrise standardteegis on defineeritud tüübiklass `Eq`:

```
interface Eq a where
   (==), (≠) : a → a → Bool

    -- Minimaalne definitioon peab sisaldama:
    -- (==) või (≠)
   x ≠ y = not (x == y) -- vaikedefinitsioon
   x == y = not (x ≠ y) -- vaikedefinitsioon
```

Võrldusoperaatori tüüp on:

```
(==) : Eq a ⇒ a → a → Bool
```

s.t me saame operaatorit kasutada, kui tema argumentitüübil on defineeritud `Eq` instants!

Kõikidel polümorfsetel funktsioonidel, mis kasutavad võrdust peab tüübi *kontekstis* olema `Eq`:

```
lookup : Eq a ⇒ a → List (a, b) → Maybe b
```

Näide

```
data ValgusFoor = Punane | Kollane | Roheline

Eq ValgusFoor where
   Punane   == Punane   = True
   Kollane  == Kollane  = True
   Roheline == Roheline = True
   _        == _        = False

test : Bool
test = Kollane == Roheline   -- False
```

Range tüübiklass

Enumeratsioone kirjeldab järgnev tüübiklass:

```
interface Range a where
   rangeFromTo : a → a → List a            -- [x..y]
   rangeFromThenTo : a → a → a → List a    -- [x,y..z]

   rangeFrom : a → Stream a                -- [x..]
   rangeFromThen : a → a → Stream a        -- [x,y..]
```

Sisend-väljund

+ `IO` on hea näide, kuidas mittepuhtaid arvutusi saab modelleerida puhaste funktsioonide abil.
+ Selline modelleerimine on teoreetiliselt huvitav, kuna võimaldab katsetada erinevaid võimalusi ja kitsendusi. Näiteks erindid ja jätkud.

- Tekitab palju segadust, kui mõisted pole (veel) selged.
- Praktiline kasu pole ilmne: lihtsam kasutada mittepuhast programmeerimiskeelt.

Seetõttu: Selles kursuses harjutame `IO`-d aga ei lähe seda rada kaugemale.

Konstantidega lambda-arvutus

- Konstantidega λ-termide süntaks:

$$
\begin{array}{lcll}
e & ::= & x & \text{muutuja} \\
 & | & c & \text{konstant} \\
 & | & (e_1\ e_2) & \text{aplikatsioon} \\
 & | & (\lambda x.\ e) & \text{abstraktsioon}
\end{array}
$$

- Iga konstandiga seotakse mingi arv δ-reduktsiooni reegleid.
- Näide: Naturaalarvud $(0, 1, 2, \ldots)$ ja liitmine $(+)$.

$$
\begin{array}{llll}
+\ 0\ 0 & \twoheadrightarrow_\delta & 0 & \qquad +\ 1\ 0 \twoheadrightarrow_\delta 1 \\
+\ 0\ 1 & \twoheadrightarrow_\delta & 1 & \qquad +\ 1\ 1 \twoheadrightarrow_\delta 2 \\
 & \ldots & & \qquad \ldots
\end{array}
$$

- δ-reeglite lisamine võib kokkuvoolavuse ära rikkuda!

Andmete esitamine λ-arvutuses

- Baaskombinaatorid

$$\begin{array}{lcl} \mathrm{I} & \equiv & \lambda x.\ x \\ \mathrm{K} & \equiv & \lambda x\ y.\ x \\ \mathrm{S} & \equiv & \lambda f\ g\ x.\ f\ x\ (g\ x) \end{array}$$

- "Astendamine"

$$\begin{array}{lcl} E^0 E' & \equiv & E' \\ E^n E' & \equiv & \underbrace{E(E(\ldots(E}_{n \text{ tükki}}\ E')\ldots)) \end{array}$$

- NB! $E^n(E\ E') \equiv E^{n+1}E' \equiv E\ (E^n E')$

Tõeväärtused

- Spetsifikatsioon

$$\begin{array}{lcl} \text{not true} & = & \text{false} \\ \text{not false} & = & \text{true} \end{array}$$

- Definitsioon

$$\begin{array}{lcll} \text{true} & \equiv & \lambda xy.\ x & (\equiv \quad \text{K}) \\ \text{false} & \equiv & \lambda xy.\ y & \\ \text{not} & \equiv & \lambda t.\ t\ \text{false}\ \text{true} & \end{array}$$

- Näide:

$$\begin{array}{lcl} \text{not true} & \equiv & (\lambda t.\ t\ \text{false}\ \text{true})\ \text{true} \\ & \to & \text{true false true} \\ & \equiv & (\lambda x.\ \lambda y.\ x)\ \text{false true} \\ & \to & (\lambda y.\ \text{false})\ \text{true} \\ & \to & \text{false} \end{array}$$

Tingimuslause

- Spetsifikatsioon

$$\begin{aligned} \text{cond true } E_1\ E_2 &= E_1 \\ \text{cond false } E_1\ E_2 &= E_2 \end{aligned}$$

- Definitsioon

$$\text{cond} \equiv \lambda t\ x\ y.\ t\ x\ y$$

- Näide:

$$\begin{aligned} \text{cond false } E_1\ E_2 &\equiv (\lambda t\ x\ y.\ t\ x\ y)\ \text{false}\ E_1\ E_2 \\ &\twoheadrightarrow \text{false}\ E_1\ E_2 \\ &\equiv (\lambda x\ y.\ y)\ E_1\ E_2 \\ &\twoheadrightarrow E_2 \end{aligned}$$