Nimi: 2. detsember 2024. a.
Rühm:

Kontrolltöö 2 (näidis)

Tüübituletuse ülesande lahendus *tuleb esitada paberil.* Ülesanded 2 kuni 5 esitada Moodlesse.

1 Tüübituletus (10/31)

Kasuta tüübituletusalgoritmi, et proovida tuletada järgneva avaldise printsipiaalne tüüp. Selleks, a) annoteeri term, b) ehita tüübituletuse puu, c) kirjuta välja kõik kitsendused ja d) lahenda kitsenduste süsteem.

Lihtsustused:

- ei pea kirjutama ”$x^{\alpha} \in \Gamma$”-sid
- kitsendusi ei pea kirjutama puu sisse (mis oli slaididel roheline)
- kitsenduste lahendamise vahesamme ei pea välja kirjutama.

Term: $(\lambda x.\ (\lambda y.\ (y(\lambda z.\ z))))$ Lahendus:

1. $(\lambda x^a.(\lambda y^b.(y^c(\lambda z^d.z^e)^f)^g)^h)^i$

2. $$\dfrac{\dfrac{\dfrac{\dfrac{}{x^a, y^b \vdash y^c}\,\text{VAR} \qquad \dfrac{\dfrac{}{x^a, y^b, z^d \vdash z^e}\,\text{VAR}}{x^a, y^b \vdash (\lambda z^d.z^e)^f}\,\text{ABS}}{x^a, y^b \vdash (y^c(\lambda z^d.z^e)^f)^g}\,\text{APP}}{x^a \vdash (\lambda y^b.(y^c(\lambda z^d.z^e)^f)^g)^h}\,\text{ABS}}{\vdash (\lambda x^a.(\lambda y^b.(y^c(\lambda z^d.z^e)^f)^g)^h)^i}\,\text{ABS}$$

3. $b = c, d = e, f = d \to e, c = f \to g, h = b \to g, i = a \to h$

4. $$\begin{array}{r} b = c, d = e, f = d \to e, c = f \to g, h = b \to g, i = a \to h \\ d = e, f = d \to e, c = f \to g, h = c \to g, i = a \to h \\ f = e \to e, c = f \to g, h = c \to g, i = a \to h \\ c = (e \to e) \to g, h = c \to g, i = a \to h \\ h = ((e \to e) \to g) \to g, i = a \to h \\ i = a \to ((e \to e) \to g) \to g \end{array}$$

   Lõpplahendus: $a \to ((e \to e) \to g) \to g$

2 Sisend-väljund (6/31)

Kirjuta protseduur yl2 n m, mis genereerib n juhuarvu nullist m-ni ja tagastab genereeritud arvude aritmeetilise keskmise.

```
yl2 : Nat → Int32 → IO Double
yl2 = ?yl2_rhs
```

Näited:

```
Main> :exec yl2 1 30 >>= printLn
12.0
Main> :exec yl2 1 30 >>= printLn
8.0
Main> :exec yl2 10000 30 >>= printLn
14.9808
Main> :exec yl2 10000 30 >>= printLn
15.0275
```

Lahendus:

```
yl2 : Nat → Int32 → IO Double
yl2 n m = do xs ← sequence l
             pure (cast (sum xs) / cast n)

           where l : List (IO Int32)
                 l = replicate n (randomRIO (0,m))
```

Alternatiivne lahendus:

```
yl2' : Nat → Int32 → IO Double
yl2' n m = do xs ← f n
              pure (cast (sum xs) / cast n)
   where  f : Nat → IO (List Int32)
          f 0 = pure []
          f (S k) = do x ← randomRIO (0,m)
                       xs ← f k
                       pure (x :: xs)
```

3 Sõltuvad tüübid (5 / 31)

Etteantud kood:

```
half : Nat → Nat
half 0 = 0
half 1 = 0
half (S (S k)) = 1 + half k
```

Kirjuta täielik funktsioon pooledElem : Vect n a → Vect (half n) a nii et testid (all) töötaksid.

```
total
pooledElem : Vect n a → Vect ( half n) a
pooledElem = ?pooledElem_rhs
```

Testid:

- pooledElem [1,2,3,4,5,6,7] == [1, 3, 5]
- pooledElem [1,2,3,4,5,6,7,8] == [1, 3, 5, 7]
- pooledElem [1,2,3,4,5,6,7,8,9] == [1, 3, 5, 7]

Lahendus:

```
total
pooledElem : Vect n a → Vect ( half n) a
pooledElem [] = []
pooledElem (x :: []) = []
pooledElem (x :: (y :: xs)) = x :: pooledElem xs
```

4 Tõestamine (5 / 31)

Etteantud kood:

```
infixl 11 \/
data (\/) : Type → Type → Type where
    DisjIl :   a
             -----
           → a \/ b

    DisjIr :   b
             -----
           → a \/ b

DisjE : a\/b → (a→ c) → (b→ c)
        ------------------------
    →               c
DisjE ( DisjIl x) q w = q x
DisjE ( DisjIr x) q w = w x
```

```
infixl 10 /\
data (/\) : Type → Type → Type where
    ConI : a → b
           -----
         → a /\ b

VoidE : Void
        ---
    →   b
VoidE q impossible
```

Ülesanne: Tõesta idrises järgnev väide.

```
total
ex4 : (a → Void) /\ (b → a)
    -------------------------
```

```
  →          (b → c)
ex4 = ?ex4_rhs
```

Lahendus:

```
total
ex4 : (a → Void) /\ (b → a)
    -------------------
  →          (b → c)
ex4 (ConI x y) q = VoidE (x (y q))
```

5 Tõestamine rewrite-ga (5 / 31)

Etteantud kood:

```
liida  : Nat → Nat → Nat
liida  0 y = y
liida  (S k) y = S ( liida  k y)

mymap : (a → b) → List  a → List  b
mymap f [] = []
mymap f (x :: xs) = f x  ::  mymap f xs
```

Ülesanne: Tõesta idrises järgnev võrdus.

```
total
mymapLiida0 :      (xs: List  Nat)
              -------------------
             →    mymap (liida 0) xs = xs
mymapLiida0 = ?mymapLiida0_rhs
```

Lahendus:

```
mymapLiida0 :     (xs: List  Nat)
              ----------------
          →   mymap (liida 0) xs = xs
mymapLiida0 [] = Refl
mymapLiida0 (x :: xs) =
    -- x :: mymap (liida 0) xs = x :: xs
    rewrite  mymapLiida0 xs in
    -- x :: xs = x :: xs
    Refl
```