13. Übung (12.–16. Juli 2021)

Programmierung

Aufgabe 1 (AGS 17.24 a, b)

(a) Geben Sie ein H_0-Programm an, welches die Eingabe einer Zahl $n \in \mathbb{N}$ fordert und dann als Ergebnis $\sum_{i=1}^{n}(-1)^i \cdot i$ ermittelt und ausgibt.

(b) Transformieren Sie die folgende Funktion `f` eines H_0-Programmes in ein AM_0-Programm mit baumstrukturierten Adressen, berechnen Sie also *funtrans*(`f :: Int -> ...`). Geben Sie dabei keine Zwischenschritte an.

```
f :: Int -> Int -> Int
f x1 x2 = if x2 == 0
          then x1
          else f x2 (x1 `mod` x2)
```

Zusatzaufgabe 1 (AGS 17.13 a ⋆)

Eine Folge e_i $(i \geq 1)$ von ganzen Zahlen soll wie folgt konstruiert werden:

- Das erste Glied der Folge sei 1.
- Das zweite Glied der Folge sei 2.
- Das dritte Glied soll von der Eingabe gelesen werden.
- Ab dem vierten Glied der Folge soll gelten: Jedes Folgeglied ist gleich der Summe der drei Vorgängerglieder.

Geben Sie ein H_0–Programm `P` an, welches das n-te Folgenelement, also e_n, dieser Folge berechnet und ausgibt.

Zusatzaufgabe 2 (AGS 17.14 a ⋆)

Transformieren Sie das folgende H_0-Programm mittels der Funktion *trans* in ein AM_0-Programm mit linearen Adressen. Sie brauchen dabei keine Zwischenschritte anzugeben.

```
1 module Main where
2
3 fac :: Int -> Int -> Int
4 fac x1 x2 = if x1 > 0 then fac (x1 - 1) (x1 * x2) else x2
5
6 main = do x1 <- readLn
7           print (fac x1 1)
```