Aufgabenblatt zur 11. Übung

Zeitraum: 27.06. bis 01.07.2011

1. Aufgabe: (AGS 15.12)

(a) Transformieren Sie die folgende Funktion `h` eines H_0-Programmes in ein AM_0-Programm mit baumstrukturierten Adressen, berechnen Sie also *funtrans*(`h :: Int -> Int   h x1 = ...`). Sie brauchen dabei keine Zwischenschritte anzugeben.

```
h :: Int -> Int
h x1 = if x1 > 1
       then if x1==2 then x1 else h (x1-1)
       else 0
```

(b) Gegeben sei der folgende Ausschnitt aus einem C_0-Programm:

```
...
x2 = x1 + 2;
while (x1 < 2)
  if (x2 <= x1) x2 = x2+1;
  else x1 = x2;
printf("%d", x1);
...
```

Durch Nutzung der Transformationsfunktion aus der Vorlesung/Übung ensteht daraus das folgende Fragment eines H_0-Programms:

```
f_3     x1 x2 = A
f_4     x1 x2 = if x1 < 2   then B
                            else C
f_4_1   x1 x2 = if x2 <= x1 then D
                            else E
f_4_1_1 x1 x2 = F
f_4_1_2 x1 x2 = G
f_5     x1 x2 = H
```

Vervollständigen Sie diesen Programmrumpf durch die Angabe der Ausdrücke für `A` bis `H`!

2. Aufgabe: (AGS 15.9*)

Gegeben ist das folgende C_0-Programm:

```
#include <stdio.h>
int main() {
  int x1, x2;
  scanf("%i", &x1);
  while (x1 > 0) {
    x2 = x1;
    while (x2 > 0) x2 = x2 - 1;
    x1 = x1 - 1;
  }
  printf("%d", x1);
  return 0;
}
```

Durch Nutzung der Transformationsfunktionen aus der Vorlesung entsteht daraus ein H_0-Programm (ohne Angabe der Funktionstypen), von dem nur das folgende Fragment gegeben ist:

```
module Main where

f_1       x1 x2 = if x1 > 0 then A
                            else B
f_1_1     x1 x2 = C
f_1_1_1   x1 x2 = D
f_1_1_2   x1 x2 = if x2 > 0 then E
                            else F
f_1_1_2_1 x1 x2 = G
f_1_1_3   x1 x2 = H
f_2       x1 x2 = I

main = do x1 <- readLn
          print (J)
```

Ergänzen Sie das Fragment durch die Angabe der Ausdrücke für `A`, `B`, `C`, `D`, `E`, `F`, `G`, `H`, `I` und `J`!

3. Aufgabe: (AGS 15.15)

(a) Schreiben Sie ein C_0-Programm auf, das durch Anwendung der Ihnen bekannten Transformationsfunktion in das folgende H_0-Programm (auf die Typdefinition wurde hier verzichtet) überführt werden kann:

```
module Main where

f_1     x1 x2 = if x1 > 0 then f_1_1 x1 x2
                      else f_2 x1 x2
f_1_1   x1 x2 = f_1_1_1 x1 x2
f_1_1_1 x1 x2 = f_1_1_2 x1-1 x2
f_1_1_2 x1 x2 = f_1 x1 x2*2
f_2     x1 x2 = x2

main = do x1 <- readLn
          x2 <- readLn
          print (f_1 x1 x2)
```

(b) Folgendes H_0-Programm sei gegeben:

```
module Main where

g :: Int -> Int -> Int
g x1 x2 = if x1 > 0 then g (x1-1) (x2*2)
                    else h x2 x2

h :: Int -> Int -> Int
h x1 x2 = x2

main = do x1 <- readLn
          x2 <- readLn
          print (g x1 x2)
```

Vervollständigen Sie die Angaben `<A>` bis `<E>` in der folgenden Übersetzung des H_0-Programmes in ein äquivalentes C_0-Programm:

```
#include <stdio.h>
int main()
{
  int x1, x2, function, flag, result;
  <A>
  while (flag == 1)
  {
    if (function == 1)
      if (<B>) {
        <C>
      } else {
        <D>

        function = 2;
      }
    else if (function == 2) {
      <E>
    }
  }
  printf("%d", result);
  return 0;
}
```

Zusatzaufgabe 1: (AGS 15.5*)

(a) Geben Sie ein H_0–Programm P an, welches die Eingabe einer Zahl $n \in \mathbb{N}$ fordert und dann als Ergebnis $s(n) = \sum_{j=1}^{n} (j+1)^2$ ermittelt und ausgibt. Ermitteln Sie zunächst eine geeignete tail-rekursive Funktion.

(b) Gegeben sei der folgende Ausschnitt aus einem C_0-Programm:

```
...
if (x1 > x2) x1 = x1 + x2;
else while (x1 < x2) x1 = x1 + 1;
printf("%d", x1);
...
```

Durch Nutzung der Transformationsfunktionen aus der Vorlesung/Übung entsteht daraus das folgende Fragment eines H_0-Programms:

```
...
f_3     x1 x2 = if x1 > x2 then A
                           else B
f_3_1   x1 x2 = C
f_3_2   x1 x2 = if x1 < x2 then D
                           else E
f_3_2_1 x1 x2 = F
f_4     x1 x2 = G
...
```

Ergänzen Sie das Fragment durch die Angabe der Ausdrücke `A`, `B`, `C`, `D`, `E`, `F` und `G`.

(c) Übersetzen Sie den C_0-Programmausschnitt aus Teilaufgabe (b) entsprechend der Ihnen bekannten Übersetzungsfunktionen in ein baumstrukturiertes AM_0-Programm. Das `if`-Statement habe die baumstrukturierte Adresse `a`. Zwischenschritte brauchen Sie nicht anzugeben.

Zusatzaufgabe 2: (AGS 15.14*)

(a) Transformieren Sie die folgende Funktion `h` eines H_0-Programmes in ein AM_0-Programm mit baumstrukturierten Adressen, berechnen Sie also $\underline{functrans}$(`h :: Int -> Int   h x1 = ...`). Sie brauchen dabei keine Zwischenschritte anzugeben.

```
h :: Int -> Int
h x1 = if x1 == 0 then h 1 else x1
```

(b) Folgendes H_0-Programm sei gegeben:

```
module Main where

f :: Int -> Int -> Int
f x1 x2 = if x1 < x2 then g (x1 + x1) (x2 * x2 + 3)
                      else f (x1 - x2) 10

g :: Int -> Int -> Int
g x1 x2 = x2

main = do x1 <- readLn
          print (f x1 x1)
```

Vervollständigen Sie die Angaben `<A>` bis `<F>` in der folgenden Übersetzung des H_0-Programmes, so dass ein äquivalentes C_0-Programm entsteht:

```
#include <stdio.h>

int main()
{
  int x1, x2, function, flag, result;
   <A>
  flag = 1;
  function = 1;
  while (flag == 1)
  {
    if ( <B> )
      if (x1 < x2) {
         <C>
        function = 2;
      } else {
         <D>
      }
    else if ( <E> ) {
       <F>
    }
  }
  printf("%d", result);
  return 0;
}
```