Universität Siegen
Lehrstuhl Theoretische Informatik
Markus Lohrey

Übungsblatt 13

Aufgabe 1.

(a) Schreiben Sie ein LOOP-Programm, das für eine Zahl n die n-te *Fibonacci-Zahl* berechnet.

(b) Schreiben Sie ein LOOP-Programm, das die Funktion $f(x, y) = x^y$ für $x \neq 0$ berechnet.

(c) Schreiben Sie ein LOOP-Programm, das die Funktion $f(x, y) = \max(x, y)$ berechnet.

(d) Schreiben Sie ein LOOP-Programm, das die Funktion $f(x, y, z) = \min(x, y, z)$ berechnet.

(e) Schreiben Sie ein WHILE-Programm, das für eine gegebene Zahl $n \geq 2$ den kleinsten Teiler p von n mit $p \geq 2$ ausgibt.

(f) Schreiben Sie ein WHILE-Programm, das die Funktion $f(n) = \lceil \sqrt{n} \rceil$ berechnet.

(g) Schreiben Sie ein GOTO-Programm für Aufgabenteil (c).

Lösung zu Aufgabe 1.

(a) Wir verwenden die folgende Definition der Fibonacci-Zahlen:

$$F(n) = \begin{cases} 0 & \text{falls } n = 0 \\ 1 & \text{falls } n = 1 \\ F(n-2) + F(n-1) & \text{falls } n \geq 2. \end{cases} \tag{1}$$

Zu Beginn ist der Eingabewert n in der Variablen x_1 gespeichert.

In x_1 und x_2 speichern wir $F(i)$ und $F(i+1)$, beginnend mit $i = 0$. Mit jeder Iteration setzen wir $x_1 := x_2$ und $x_2 := x_1 + x_2$.

$x_4 := x_1;$
$x_1 := 0;$
$x_2 := 1;$
LOOP x_4 DO
  $x_3 := x_1 + x_2;$
  $x_1 := x_2;$
  $x_2 := x_3$
END

Beachte: Das Ergebnis steht bei Termination des Programms in der Variablen x_1 (siehe Folie 388).

(b) $f(x, y) = x^y$. Zu Beginn steht x in x_1 und y in x_2.

$x_3 := x_1;$
$x_1 := 1;$
LOOP x_2 DO
  $x_1 := x_1 \cdot x_3$
END

(c) Wir simulieren zunächst die (abgeschnittene) Subtraktion von Variablen ($x_i := x_j - x_k$ für $i \neq k$) als LOOP-Programm:

$x_i := x_j;$
LOOP x_k DO $x_i := x_i - 1$ END

$f(x, y) = \max(x, y)$. Zu Beginn steht x in x_1 und y in x_2.

$x_3 := x_2 - x_1;$
LOOP x_3 DO $x_1 := x_2$ END

Falls $y > x$ wird die LOOP-Schleife (mindestens einmal) ausgeführt und der „Rückgabewert“ mit y (x_2) überschrieben.

(d) $f(x, y, z) = \min(x, y, z)$. Zu Beginn steht x in x_1, y in x_2 und z in x_3.

$x_4 := x_2 - x_3$;
LOOP x_4 DO $x_2 := x_3$ END;
$x_4 := x_1 - x_2$;
LOOP x_4 DO $x_1 := x_2$ END;

Wir verwenden eine ähnliche Technik wie bei Aufgabenteil (c).

Mit den ersten zwei Zeilen prüfen wir, ob $x_2 > x_3$. Falls ja, ist $x_4 > 0$ und wir überschreiben x_2 mit x_3 ($x_2 := \min(y, z)$).

Dann wiederholen wir den Vorgang mit x_1 und x_2. Am Ende steht in x_1 der Wert $\min(x, \min(y, z)) = \min(x, y, z)$.

(e) Definiere zunächst $x_i := x_j \bmod x_k$ (für $i \neq j$ und $i \neq k$) durch

$x_i := 0$;
LOOP x_j DO
  $x_i := x_i + 1$;
  $x_h := x_k - x_i$; // h verschieden von i, j, k
  IF $x_h = 0$ THEN $x_i := 0$ END;
END

Idee: Wir inkrementieren x_i x_j-mal um 1. Nach jedem Durchlauf der Schleife prüfen wir mit Hilfe der Hilfsvariable x_h, ob $x_i = x_k$ gilt (also $x_k - x_i = 0$). Falls ja, setzen wir x_i auf 0 zurück.

Zu Beginn steht n in x_1. Falls p ein Teiler von n ist, gilt $n \bmod p = 0$.

$x_2 := 1$; // Speicher für $x_1 \bmod x_3$
$x_3 := 1$; // Zähler für den Teiler p
WHILE $x_2 \neq 0$ DO
  $x_3 := x_3 + 1$;
  $x_2 := x_1 \bmod x_3$
END;
$x_1 := x_3$ // p in x_1 kopieren

(f) $f(n) = \lceil \sqrt{n} \rceil$, n steht in x_1.

$x_2 := 0;$ // Counter c
$x_3 := 0;$ // Speicher für c^2
$x_4 := x_1;$ // Speicher für $n - c^2$
WHILE $x_4 \neq 0$ DO
  $x_2 := x_2 + 1;$
  $x_3 := x_2 \cdot x_2;$
  $x_4 := x_1;$
  $x_4 := x_4 - x_3;$
END;
$x_1 = x_2$

Idee: Wir zählen c hoch und berechnen nach jedem Schritt $n - c^2$. Falls c die kleinste Zahl ist, sodass $c^2 \geq n$ (also $c = \lceil \sqrt{n} \rceil$) ist x_4 dann 0, wir beenden die WHILE-Schleife und speichern c in x_1 (dem Rückgabewert).

(g)

$x_3 := x_1;$
$x_4 := x_2;$

M_1 : IF $x_3 = 0$ THEN GOTO M_2 END;
IF $x_4 = 0$ THEN HALT END;
$x_3 := x_3 - 1;$
$x_4 := x_4 - 1;$
GOTO M_1;

M_2 : $x_1 := x_2;$
HALT;

Idee: Zu Beginn steht x in x_1 und y in x_2. Wir zählen parallel beide Zahlen (x in x_3 und y in x_4) runter. Falls x_3 zuerst 0 wird, gilt $\max(x, y) = y$ und wir springen zu M_2, um x_1 (den Rückgabewert) mit x_2 zu überschreiben, und beenden das Programm. Wird x_4 zuerst 0, ist $\max(x, y) = x$. Da x schon in x_1 steht, können wir das Programm direkt beenden.

Aufgabe 2. Zeigen Sie, dass die folgenden Funktionen primitiv rekursiv sind. Es dürfen primitiv rekursive Funktionen verwendet werden, die in der Vorlesung bereits besprochen wurden.

(a) $f(n) = n!$

(b) $g(n) = \frac{n \cdot (n+1)}{2}$

(c) $k(n, m) = m^n$

(d) $h(x_1, x_2, x_3) = \begin{cases} x_2 & \text{für } x_1 = 0 \\ x_3 & \text{sonst} \end{cases}$

Lösung zu Aufgabe 2. Wir bezeichnen mit add die Addition, mit mult die Multiplikation und mit $\text{comp}(g, f_1, \ldots, f_k)$ die Komposition der Funktion g mit den Funktionen $f_1, \ldots, f_k$ (siehe Folie 420 im Skript).

(a) Wir definieren die Funktion $\varphi\colon \mathbb{N}^2 \to \mathbb{N}$ mit $\varphi(x, y) = x \cdot (y + 1)$. Dann können wir $f(n) = n!$ mittels primitiver Rekursion definieren:

$$\begin{aligned} f(0) &= 1 \\ f(n+1) &= \varphi(f(n), n). \end{aligned}$$

Wir erhalten $f(n + 1) = f(n) \cdot (n + 1) = n! \cdot (n + 1) = (n + 1)!$ und für den Basisfall gilt $f(0) = 1 = 0!$. Die Konstante 1 aus dem Basisfall wird formal durch die konstante Funktion $k_1\colon \mathbb{N}^0 \to \mathbb{N}$ mit $k_1() = 1$ repräsentiert. Die Funktion φ ist primitiv rekursiv, da die Nachfolgerfunktion, Kompositionen und Projektionen primitiv rekursiv sind (siehe Folie 420 im Skript). Genauer:

$$\begin{aligned} \varphi(x, y) &= \text{comp}(\text{mult}, \pi_1^2, \text{comp}(s, \pi_2^2))(x, y) \\ &= \text{mult}(\pi_1^2(x, y), \text{comp}(s, \pi_2^2)(x, y)) \\ &= \text{mult}(x, s(\pi_2^2(x, y))) \\ &= \text{mult}(x, s(y)) \\ &= x \cdot (y + 1). \end{aligned}$$

(b) Es gilt $g(n) = \sum_{i=1}^{n} i$ (Gauß'sche Summenformel). Damit lässt sich g analog zur Fakultät aus (a) definieren, wobei der Wert $g(0)$ im Basisfall diesmal 0 statt 1 ist, und add statt mult in der Rekursionsdefinition verwendet wird: Sei $\psi\colon \mathbb{N}^2 \to \mathbb{N}$ definiert als $\psi(x, y) = x + (y + 1)$, dann kann man g mittels primitiver Rekursion wie folgt definieren:

$$\begin{aligned} g(0) &= 0 \\ g(n+1) &= \psi(g(n), n). \end{aligned}$$

Wir erhalten $g(n+1) = g(n) + (n+1) = \sum_{i=1}^{n} i + (n+1) = \sum_{i=1}^{n+1} i$ und für den Basisfall gilt $g(0) = 0 = \sum_{i=1}^{0} i$.

(c) Definiere $\tau\colon \mathbb{N}^3 \to \mathbb{N}$ mit $\tau(x, y, z) = x \cdot z = \text{mult}(x, z)$. Es gilt

$$\begin{aligned} k(0, m) &= 1 \\ k(n+1, m) &= \tau(k(n, m), n, m). \end{aligned}$$

Wir erhalten $k(n+1, m) = k(n, m) \cdot m = m^n \cdot m = m^{n+1}$.

(d) Wir definieren Funktionen $h_1\colon \mathbb{N}^2 \to \mathbb{N}$ mit $h_1(x, y) = \pi_1^2(x, y) = x$ und $h_2\colon \mathbb{N}^4 \to \mathbb{N}$ mit $h_2(a, b, c, d) = \pi_4^4(a, b, c, d) = d$. Es gilt

$$\begin{aligned} h(0, x_2, x_3) &= h_1(x_2, x_3) = x_2 \\ h(n+1, x_2, x_3) &= h_2(h(n, x_2, x_3), n, x_2, x_3) = x_3. \end{aligned}$$