Fakultät IV Elektrotechnik und Informatik
Informatik 1 WS 2001/02, Jähnichen / Flik / Lechtchinsky / Neunast / Tutoren

Aufgabenblatt 2

letzte Aktualisierung: 02. November, 15:34

Ausgabe: *2.11.2001*
Abgabe: *12.11. / 13.11.2001* *Prozent:* *100*

Thema: *Rekursion; lokale Bindungen*

1. Aufgabe (40 Prozent): Rekursion verwenden

In dieser Aufgabe soll die vorgegebene Struktur `Recursion` geeignet erweitert werden.

1.1. **(Tut)** Implementiert die Funktion `plus`, die rekursiv die Summe zweier natürlicher Zahlen berechnet.

Hinweis: Verwendet die Funktionen `succ` und `pred` aus der Struktur `Nat`, die den Nachfolger bzw. den Vorgänger einer natürlichen Zahl liefern.

1.2. Implementiert die Funktion `mult`, die rekursiv das Produkt zweier natürlichen Zahlen berechnet.

Hinweis: Verwendet die Funktionen `+` und `-` aus der Struktur `Nat`.

1.3. **(Tut)** Implementiert die Funktion `sum`, die die Summe aller natürlichen Zahlen zwischen den Zahlen `m` und `n` berechnet.

1.4. Implementiert die Funktion `sumEven`, die die Summe aller geraden Zahlen zwischen den Zahlen `m` und `n` berechnet.

1.5. **(Tut)** Implementiert die Funktion `in?`, die überprüft, ob ein bestimmtes Zeichen in einer Zeichenkette vorkommt.

1.6. Implementiert die Funktion `count`, die bestimmt, wie oft ein bestimmtes Zeichen in einer Zeichenkette vorkommt.

1.7. Implementiert die Funktion `countDigits`, die bestimmt, wieviele Ziffern eine Zeichenkette enthält.

2. Aufgabe (30 Prozent): Rekursion verstehen

2.1. **(Tut)** Was berechnen die Funktion `h` und ihre Hilfsfunktionen `h1` und `h2`? Simuliert die Funktionsauswertung für den Aufruf `h(11,3)`.

```
FUN h : nat ** nat -> nat ** nat
DEF h == \\m,n. (h1(m,n), h2(m,n))

FUN h1 : nat ** nat -> nat
DEF h1 == \\m,n. IF m < n THEN 0 ELSE h1(m-n,n) + 1 FI

FUN h2 : nat ** nat -> nat
DEF h2 == \\m,n. IF m < n THEN m ELSE h2(m-n,n) FI
```

2.2. Was berechnet die Funktion `f`?

```
FUN f : nat ** nat -> nat
DEF f == \\a,b. IF a > b THEN f1(a,b,a)
                         ELSE f1(b,a,b)
                FI

FUN f1 : nat ** nat ** nat -> nat
DEF f1 == \\a,b,z. IF mod(a,b) = 0 THEN a
                                   ELSE f2(a+z,b,z)
                   FI
```

2.3. Was berechnet die Funktion g?

```
FUN g : nat ** nat -> nat
DEF g == \\a,b. IF b = 0 THEN a
                         ELSE g(b,mod(a,b))
                FI
```

3. Aufgabe (30 Prozent): Lokale Bindungen mit `LET` und `WHERE`

In dieser Aufgabe soll die vorgegebene Struktur `Solve` geeignet erweitert werden.

3.1. **(Tut)** Implementiert die Funktion `triangleArea`, die die Fläche eines Dreiecks mit den Seitenlängen `a`, `b` und `c` nach der Heron'schen Formel berechnet.

3.2. Implementiert die Funktion `qsolve`, die die Lösungen einer quadratischen Gleichung der Form $a_2x^2 + a_1x + a_0 = 0$ mit $a_2 \neq 0$ aus den Koeffizienten a_0, a_1 und a_2 mit Hilfe der pq–Formel berechnet.

3.3. Implementiert die Funktion `qunique?`, die überprüft, ob eine quadratische Gleichung genau eine Lösung hat.

Hinweis: Verwendet hierfür die Funktion `qsolve` aus der vorherigen Aufgabe.