CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

Prof. Bernd Finkbeiner, Ph.D.
Jana Hofmann, M.Sc.
Reactive Systems Group

Programmierung 1 (WS 2020/21)
Übungsblatt B

Lesen Sie im Buch Kapitel 1.5 - 2.3.

Hinweis: Über Aufgaben, die mit 🤔 markiert sind, müssen Sie eventuell etwas länger nachdenken. Falls Ihnen keine Lösung einfällt - kein Grund zur Sorge. Kommen Sie in die Office Hour, unsere Tutor:innen helfen gerne.

Schnellkurs

Aufgabe B.1

(a) Geben Sie ein Tupel mit 3 Positionen und nur einer Komponente an.

(b) Geben Sie einen Tupelausdruck an, der den Typ `int * (bool * (int * unit))` hat.

(c) Geben Sie ein Paar an, dessen erste Komponente ein Paar und dessen zweite Komponente ein Tripel ist.

Aufgabe B.2

Schreiben Sie eine Prozedur `min : int * int → int`, die zu zwei Zahlen die kleinere liefert. Deklarieren Sie `min` auf 3 verschiedene Arten: mit einem kartesischen Argumentmuster, mit einer lokalen Deklaration und mit Projektionen.

Aufgabe B.3

Gegeben sei die folgende rekursive Prozedurdeklaration:

```
fun f (n : int, a : int) : int = if n = 0 then a else f (n - 1, a * n)
```

(a) Geben Sie die Rekursionsfolge für den Aufruf `f (3, 1)` an.

(b) Geben Sie ein detailliertes Ausführungsprotokoll für den Aufruf `f (3, 1)` an. Wenn es mehrere direkt ausführbare Teilausdrücke gibt, soll immer der am weitesten links stehende zuerst ausgeführt werden. Sie sollten insgesamt 18 Ausführungsschritte bekommen.

(c) Geben Sie ein verkürztes Ausführungsprotokoll für den Aufruf `f (3, 1)` an.

(d) Welche Funktion berechnet die Prozedur? 🤔

Aufgabe B.4

Implementieren Sie Multiplikation mit Addition, d.h. schreiben Sie eine Prozedur `times : int*int → int`, s.d. für positive Argumente, `times(a,b)` $= a * b$. Verwenden Sie für `times` nicht den Multiplikationsoperator. Schreiben Sie `times` auf zwei Arten, als rekursive Prozedur und mit einer endrekursiven Hilfsprozedur. Geben Sie in beiden Fällen zuerst die Rekursionsgleichungen an.

Aufgabe B.5

(a) Schreiben Sie eine rekursive Prozedur `quer : int → int`, welche die Quersumme einer natürlichen Zahl berechnet. Die Quersumme einer Zahl ist die Summe ihrer Dezimalziffern. Beispielsweise hat die

Zahl 3754 die Quersumme 19. Geben Sie zunächst die Rekursionsgleichungen für `quer` an. Verwenden Sie Restbestimmung modulo 10, um die letzte Ziffer einer Zahl zu bestimmen.

(b) Schreiben Sie `quer` nun mithilfe einer *endrekursiven* Hilsprozedur `quer' : int*int → int`.

Aufgabe B.6

(a) Schreiben Sie eine rekursive Prozedur `sort : int * int * int → int * int * int`, die ganzzahlige Tripel sortiert. Beispielsweise soll `sort (7, 2, 5)` das Tupel `(2, 5, 7)` zurückgeben. Verwenden Sie nur 2 Konditionale.

(b) Schreiben Sie mithilfe von `sort` eine Prozedur `max : int * int * int → int`, die zu drei Zahlen die Größte liefert. Verwenden Sie dabei keine kartesischen Muster.

Aufgabe B.7 *(Caesars Cipher)*
Die Caesar-Verschlüsselung ist ein Verschlüsselungsverfahren, das von Gaius Julius Caesar für die geheime Kommunikation benutzt wurde. Bei der Verschlüsselung wird jeder Buchstabe um drei Buchstaben im Alphabet verschoben. Zum Beispiel wird das Wort "Reaktiv" als "Uhdnwly" verschlüsselt.

Schreiben Sie eine Prozedur `caeser : int → int`, welche die Caesar-Verschlüsselung für ganze Zahlen berechnet. Dabei soll die Prozedur jede Ziffer einer ganzen Zahl um drei verschieben. Die Zahl -987 wird beispielsweise als -210 verschlüsselt.

Aufgabe B.8
Sei die folgende Prozedur gegeben:

```
fun p (n : int) : int = if n = 0 then 0
else if n <= 2 then 1
else p (n - 1) + p (n - 2)
```

Geben Sie die Rekursionsfolge für den Aufruf `p 5` an. Was ist der Unterschied zu den Folgen, die Sie bisher gesehen haben?

Aufgabe B.9 *(Primzahlen I)*
Per Definition sind Primzahlen natürliche Zahlen, die echt größer als 1 sind und nur durch 1 und sich selbst teilbar sind. Schreiben Sie eine Prozedur `nextprim : int → int`, die zu einer gegebenen natürlichen Zahl n die nächstgrößere Primzahl liefert. Beispielsweise wertet `nextprim 7` zu `11` aus. Sie dürfen davon ausgehen, dass eine Hilfsprozedur `istprim : int → bool` gegeben ist, die überprüft, ob eine Zahl Primzahl ist.

Aufgabe B.10 *(Primzahlen II)*
Wir werden nun eine Prozedur schreiben, die entscheidet, ob eine Zahl eine Primzahl ist.

(a) Schreiben Sie eine Prozedur `teilbar : int * int → bool`, die `true` zurückgibt, wenn die erste Zahl durch die zweite Zahl teilbar ist.

(b) Schreiben Sie eine rekursive Prozedur `teiler : int * int → int`, die den kleinsten Teiler der ersten Zahl zurückgibt, der größer oder gleich der zweiten Zahl ist. Gehen sie davon aus, dass die erste Zahl größer gleich der zweiten Zahl ist.

(c) Schreiben Sie eine Prozedur `istprim : int → bool`, die entscheidet, ob die übergebene Zahl eine Primzahl ist.

Hinweis: Versuchen Sie, geschickt die Prozedur aus Aufgabenteil (b) zu verwenden.

Aufgabe B.11
Unter der Reversion *rev* n einer natürlichen Zahl n wollen wir die natürliche Zahl verstehen, die man durch Spiegeln der Dezimaldarstellung von n erhält. Beispielsweise soll *rev* $1234 = 4321$, *rev* $76 = 67$ und *rev* $1200 = 21$ gelten.

(a) Schreiben Sie zunächst eine endrekursive Prozedur `rev': int * int → int`, die zu zwei natürlichen Zahlen m und n die Zahl liefert, die sich ergibt, wenn man die reversierte Dezimaldarstellung von n rechts an die Dezimaldarstellung von m anfügt. So soll `rev'(65,73) = 6537` und `rev'(0,12300) = 321` gelten. Die Arbeitsweise von `rev'` ergibt sich aus diesem verkürzten Ausführungsprotokoll:

`rev'(65,73) = rev'(653,7) = rev'(6537,0) = 6537`

(b) Schreiben Sie mithilfe der Prozedur `rev'` eine Prozedur `rev`, die natürliche Zahlen reversiert.

Aufgabe B.12
Die Fibonacci-Folge ist definiert als:

$$
\begin{aligned}
&\mathit{fib}\ 0 = 1 \\
&\mathit{fib}\ 1 = 1 \\
&\mathit{fib}\ n = \mathit{fib}\ (n-1) + \mathit{fib}\ (n-2) \qquad \text{für} \quad n > 1
\end{aligned}
$$

Im Folgenden sollen Sie drei Prozeduren implementieren, die mit der Fibonacci Folge arbeiten.

(a) n heißt Fibonacci-Teiler zu k, wenn $1 < n < \mathit{fib}(k)$ und $n \mid \mathit{fib}(k)$ gilt (n teilt $\mathit{fib}(k)$). Man kann zeigen, dass es für jedes n ein k gibt, sodass n ein Fibonacci-Teiler zu k ist. Implementieren Sie eine Prozedur `teilerFib : int → int`, sodass sie für jedes n das kleinste k findet, sodass n Fibonacci-Teiler zu k ist.

(b) n heißt primitiver Fibonacci-Teiler zu k, wenn n Fibonacci-Teiler zu k ist und für alle $k' < k$ gilt: n ist kein Fibonacci-Teiler zu k'. Implementieren Sie eine Prozedur `primFib : int → int`, die gegeben k, einen primitiven Fibonacci-Teiler n zu k berechnet, wenn dieser existiert.

(c) n heißt primitiver Fibonacci-Primteiler zu k wenn n primitiver Fibonacci-Teiler zu k ist und gleichzeitig eine Primzahl. Ist e die höchste Zahl, sodass n^e immernoch Fibonacci-Teiler zu k ist, so heißt n^e primitive Fibonacci-Primteilerpotenz zu k. Implementieren Sie eine Prozedur `PriFibPriTei : int → int`, die für ein k das Produkt aller primitiven Primteilerpotenzen zu k berechnet.

Programmiersprachliches

Aufgabe B.13
Geben Sie die Baumdarstellungen der folgenden durch Zeichendarstellungen beschriebenen Phrasen an.

(a) `x + 3 * f x - 4`

(b) `int * int → bool`

(c) `int * (int * int) * int → real`

(d) `let val x = 2 + y in x - y end`

(e) `fun p (x : int, n : int) : int = if n > 0 then x * p (x, n - 1) else 1`

Aufgabe B.14
Betrachten Sie erneut die Syntaxübersicht von SML auf Seiten 33 und 34. Überlegen Sie sich nun eine Reihe von syntaktisch nicht zulässigen Ausdrücken und geben Sie die Gleichung an, gegen die Ihre Ausdrücke verstoßen.

kNobelpreis

Nächste Woche startet der kNobelpreis. Alle wichtigen Informationen und Regularien finden Sie hier. Wenn Sie am kNobelpreis teilnehmen möchten, beachten Sie bitte die folgenden Punkte:

(a) Melden Sie Ihr Team mit Teamname, Teambadge (ihr Logo, das im Forum sichtbar sein wird) und Namen der Teammitglieder bei Jana Hofmann (jana.hofmann@cispa.saarland) an. Auch wenn Sie alleine abgeben möchten, müssen Sie ein Team gründen.

(b) Wenn Sie noch auf Teamsuche sind, hilft Ihnen unsere Partner-Börse im Forum weiter.

(c) Bis zum 31.12.2020 können Sie neue Teams bilden, vorhandene zusammenführen oder neue Mitglieder aufnehmen.