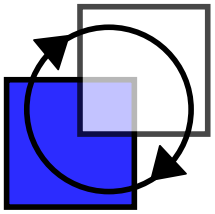
CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

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



Programmierung 1 (WS 2020/21)
Übungsblatt C

Lesen Sie im Buch Kapitel 2.4 - 3.7, sowie 3.13.

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.

Programmiersprachliches

Aufgabe C.1

Prüfen Sie, ob die folgenden Phrasen wohlgetypt sind, indem Sie einen Ableitungsbaum erstellen. Nehmen Sie an, dass `x` und `y` jeweils vom Typ `int` sind, `b` vom typ `bool` ist, `f` vom Typ `int → bool`, `g` vom Typ `int * int → bool` und `q` vom Typ `bool → int` sind.

Falls die Phrase *nicht* semantisch zulässig ist, markieren Sie die Stelle im Baum, bei der die Typprüfung fehlschlägt. Markieren Sie, welche Typen (unterhalb des Fehlschlages) damit nicht inferiert werden können.

(a)

```
1 if x > 3 then f(8) else if x * 2 = y then true else g(x,y)
```

(b)

```
1 if if b = b then q(b) else 42 then b mod 2 = 0 else true
```

Aufgabe C.2

Welche Bindungen berechnet das folgende Programm?

```
1 fun f (x : bool) = if x then 1 else 0
2 val x = 5 * 7
3 fun g (z : int) = f (z < x) < x
4 val x = g 5
```

Aufgabe C.3

Entscheiden Sie für die folgenden Programme, ob sie semantisch zulässig sind. Orientieren Sie sich an den vier Ausführungsphasen eines Interpreters und geben Sie an, in welcher Phase die Ausführung gegebenfalls scheitert.

(a)

```
1 val a = 5
2 fun f (x : int) = x + a
```

(b)

```
1 val a = true
2 fun f (x : int) = x + a
```

(c)

```
1 val a = 5
2 fum f (x : int) = x + a
```

(d)

```
1 val a = 5
2 fun f (x : int) = a +
```

Aufgabe C.4

Geben Sie die Tripeldarstellung der folgenden Prozedur an.

```
1 fun fak (n : int) : int = if n < 1 then 1 else n * fak (n - 1)
```

Aufgabe C.5
Betrachten Sie das folgende Programm:

```
val x = 3 + 2
fun f (y : int) = x + y
fun g (y : int) : int = if y < x then 0 else y + g (y - 1)
```

(a) Welche Umgebung liefert die Ausführung des Programms in der leeren Umgebung?

(b) Geben Sie die Umgebung an, in der der Rumpf der Prozedur f bei der Ausführung des Aufrufs f 7 ausgeführt wird.

(c) Geben Sie die Umgebung an, in der der Rumpf der Prozedur g bei der Ausführung des Aufrufs g 13 ausgeführt wird.

Aufgabe C.6
Seien U und V zwei Umgebungen. Unter welchen Bedingungen über V und U gilt für die Adjunktion $+$:

$$U + V = V + U$$

Aufgabe C.7
Entscheiden Sie, ob folgende Programme jeweils semantisch äquivalent sind:

(a)

```
fun fac (x : int) = if x = 0 then 1
  else fac (x - 1)
fun b (p : int * int) = (#2 p, #1 p)
val it = b (fac (~1), fac 1)
```

```
fun fac (x : int) = if x = 0 then 1
  else x * fac (x - 1)
fun b (p : int * int) = p
val it = b (fac 1, fac (~1))
```

(b)

```
val it = 111111111111 mod 1
```

```
val it = 111111 mod 1
```

Höherstufige Prozeduren

Aufgabe C.8
Schreiben Sie zwei Prozeduren

```
cas : (int * int → int) → int → int → int
car : (int → int → int) → int * int → int
```

so, dass *cas* zur kartesischen Darstellung einer zweistelligen Operation die kaskadierte Darstellung und *car* zur kaskadierten Darstellung die kartesische Darstellung liefert. Erproben Sie *cas* und *car* mit einem Interpreter!

Aufgabe C.9
Geben Sie *geschlossene Abstraktionen* an, welche die folgenden Typen haben. Die Abstraktionen sollen nur mit Prozeduranwendungen, Tupeln und Bezeichnern gebildet werden. Konstanten und Operatoren sollen nicht verwendet werden.

(a) `(unit * int) → unit → int`

(b) `(int * real * unit) → real → int * real`

(c) `int → unit → real → int * int`

Aufgabe C.10
Deklarieren Sie mit Hilfe der Prozedur `first` eine Prozedur `aufrundenBitte : int → int`, die eine gegebene Zahl auf die nächstgrößere durch 10 teilbare Zahl aufrundet. Die Prozedur selbst darf nicht rekursiv sein.

Aufgabe C.11
Geben Sie zu jeder der folgenden Abstraktionen einen semantisch äquivalenten Ausdruck an, der ohne die Verwendung einer Abstraktion gebildet ist.

Hinweis: Verwenden Sie Let-Ausdrücke und Prozedurdeklarationen.

(a) `fn (x : int) ⇒ x * x`

(b) `fn (x : int) ⇒ (fn (y : int) ⇒ x + y)`

Aufgabe C.12
Schreiben Sie eine Prozedur, die n-mal abwechselnd 3 und 7 zu einem anfänglich gegebenen Startwert s, beginnend mit 3, addiert. Verwenden Sie dazu die Hilfsprozedur `iter`.

Aufgabe C.13
Schreiben Sie mit Hilfe von `iter` eine Prozedur `zins : real → real → int → real`, die anhand eines gegebenen Kapitals, eines Zinssatzes und einer Anzahl von Jahren das angesparte Vermögen berechnet. Die Prozedur selbst darf nicht rekursiv sein.

Aufgabe C.14
Schreiben Sie eine Prozedur `fib : int → int`, welche die n-te Fibonacci-Zahl mit Hilfe von `iter` berechnet. Benutzen Sie bei dieser Aufgabe Tupel und Projektionen. 🤔

Aufgabe C.15
Deklarieren Sie eine Prozedur `iterup' : int → int → α → (int * α → α) → α`, sodass sich diese Prozedur für alle Argumente wie das Ihnen bekannte `iterup` verhält. Benutzen Sie hierfür nur `iter` und keine zusätzliche Rekursion. 🤔

Aufgabe C.16
Schreiben Sie eine Prozedur `polyFirst : α → (α → α) → (α → bool) → α`, welche die Prozedur `first` so verallgemeinert, dass der Startwert nicht um eins erhöht wird, sondern durch eine zusätzlich übergebene Schrittprozedur `α → α` berechnet wird.

Aufgabe C.17
Bestimmen Sie die Typschemata folgender Prozeduren:

(a) `fun f a b c = c = (a = b)`

(b) `fun f a b (c, d) = a(b, c (d a))`

Aufgabe C.18
Deklarieren Sie polymorphe Prozeduren, die die folgenden Typschemata besitzen, ohne explizite Typangaben wie `(x:int)` zu benutzen.

(a) $\forall\alpha,\beta.(\alpha \to \alpha \to \beta) \to \alpha \to \beta$

(b) $(\text{int} \to \alpha) \to {'\beta} \to \gamma \to ({'\beta} \to \text{bool})$

(c) $\forall\alpha,\beta,\gamma.\ \alpha * \beta * \gamma \to \beta$

(d) $(\alpha \to \beta)^*(\beta \to \gamma) \to \alpha \to \text{bool}$

Aufgabe C.19
Deklarieren Sie eine Identitätsprozedur `ideq : 'α → 'α`, deren Typschema auf Typen mit Gleichheit eingeschränkt ist. Verzichten Sie dabei auf explizite Typangaben. 🤔

Aufgabe C.20
Dieter Schlau ist begeistert. Scheinbar kann man `let`-Ausdrücke mit `val`-Deklarationen auf Abstraktion und Applikation zurückführen:

$$\text{let val } x = e \text{ in } e' \text{ end} \quad \leftrightsquigarrow \quad (\text{fn } (x : t) \Rightarrow e')\ e$$

Auf der Heimfahrt von der Uni kommen ihm jedoch Zweifel an seiner Entdeckung, da ihm plötzlich einfällt, dass Argumentvariablen nicht polymorph getypt werden können. Können Sie ein Beispiel angeben, das zeigt, dass Dieters Zweifel berechtigt sind?

Hilfe: Sie sollen einen semantisch zulässigen Let-Ausdruck angeben, dessen Übersetzung gemäß des obigen Schemas scheitert, da Sie keinen passenden Typ t für die Argumentvariable x finden können.

Aufgabe C.21
Dieter Schlau ist ganz begeistert von polymorphen Prozeduren. Er sitzt ~~*im iCoffee*~~ zuhause am Schreibtisch und deklariert die Prozedur

```
1 fun α pif (x : bool, y : α, z : α) = if x then y else z
```

Er behauptet, dass man statt eines Konditionals stets die Prozedur `pif` verwenden kann:

$$\text{if } e_1 \text{ then } e_2 \text{ else } e_3 \quad \rightsquigarrow \quad \mathit{pif}\ (e_1, e_2, e_3)$$

Was übersieht Dieter?

kNobelpreis

Retten Sie Dieter Schlau! Bei dieser Aufgabe brauchen Sie all Ihren Mut, um Ihrem Freund Dieter in einem gruseligen Hörspiel aus der Patsche zu helfen – bewaffnet nur mit einer einzigen rekursiven Prozedur. Die Aufgabe finden Sie unter folgendem Link: https://knobel.prog1.party

Ihre Lösung kann von einer Person pro Gruppe auf der persönlichen Statusseite im CMS abgegeben werden. Die Abgabefrist für die kNobelaufgabe ist **Dienstag, 24.11.2020 um 23:59**. Teilen Sie uns mit der ersten Abgabe bitte auch mit, welche der Gruppenmitglieder namentlich auf der kNobelpreis-Seite im CMS (sichtbar nur für registrierte Nutzer) genannt werden dürfen. Neue Gruppen können noch bis zum 31.12.2020 per Mail an Jana Hofmann (jana.hofmann@cispa.saarland) registriert werden. Weitere Informationen zum kNobelpreis (z.B. zur erforderlichen Dokumentation Ihrer Lösung) finden Sie hier. Wenn Sie Rückfragen zur Aufgabenstellung haben, können Sie sich im Forum an Dieter wenden.