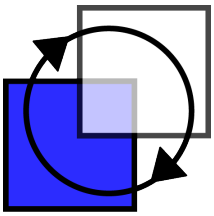
CISPA HELMHOLTZ CENTER FOR INFORMATION SECURITY

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



Programmierung 1 (WS 2020/21)
Zusatzerklärung zum Thema
;

Diese Zusatzerklärung soll sich um die Aufgabe des ; drehen. Das ; zwischen Deklarationen ist **nicht** Teil der Programmiersprache SML.

> "Damit der Interpreter mit der Bearbeitung des Programms beginnt, muss nach dem Programm zunächst ein Semikolon ; eingegeben und direkt danach die Eingabetaste gedrückt werden. Das Hilfszeichen ; ist erforderlich, damit mehrere Zeilen am Stück übergeben werden können."
>
> *Seite 4, Programmierung – eine Einführung in die Informatik mit Standard ML*

1 Aufbau eines Programms

Ein SML Programm ist eine Folge von Deklarationen. Whitespace (Leerzeichen, Zeilenumbrüche, etc.) zwischen Deklarationen wird ignoriert. Die folgenden SML-Programme sind also alle gültig:

(a) Zwei `val`-Deklarationen, wie wir sie sonst auch kennen:

```
1 val a = (1, 3)
2 val b = #1 a
```

(b) Zwei `val`-Deklarationen, jedoch dieses Mal in einer Zeile:

```
1 val a = (1, 3) val b = #1 a
```

(c) Zwei Prozedurdeklarationen, wie wir sie sonst auch kennen:

```
1 fun plus (x: int, y: int) : int = x + y
2 fun minus (x: int, y: int) : int = x - y
```

(d) Zwei Prozedurdeklarationen, jedoch dieses Mal in einer Zeile:

```
1 fun f (x: int) : int = x fun g (y: int) : int = y + y
```

(e) Das leere Programm:

```
1
```

(f) Ein etwas komplizierteres Programm:

```
1 fun fac (n: int) : int =
2    let
3       (* Prozedurdeklaration in einem let *)
4       fun faci (n: int, a: int) : int =
5          if n < 1 then a else faci(n - 1, a * n)
6    in
7       faci(n, 1)
8    end
```

2 Aufgabe des Semikolons

Bei genauem Hinschauen sieht man, dass in keinem dieser Programme ein `;` vorkommt. Das liegt daran, dass ein `;` zwischen Deklarationen nicht benötigt wird. In der Programmierung 1 ist für uns ein `;` zwischen Deklarationen noch nicht einmal Teil des Programms. Das `;` in einem Interpreter dient nur dazu, zu signalisieren, dass der davorstehende Code interpretiert werden soll.

Achtung: Das folgende ist kein valides SML-Programm:

```
fun f (n: int): int = if n < 1 then 1 else n * f (n - 1)
f 4
```

Auch wenn hier der Aufruf von `f` in einer neuen Zeile steht, müssen wir bedenken, dass der Interpreter Zeilenumbrüche ignoriert. Hierbei handelt es sich nicht um eine Folge von Deklarationen. Um dies zu ändern sollte der Trick nicht sein, ein `;` einzufügen. Stattdessen können wir es wie folgt zu einem gültigen Programm umwandlen:

```
fun f (n: int): int = if n < 1 then 1 else n * f (n - 1)
val it = f 4
```

3 Verständnischeck

- Welche der folgenden Semikolons werden benötigt, damit der Interpreter keinen Fehler wirft?
- Wie könnte man den Code umschreiben, sodass kein Semikolon mehr vorkommt?

(a)

```
fun f (x: int) = x + 2 * x;

val a = f 42
val b = f 96


;
```

(b)

```
fun f (x: int): int = if x < 1 then x else x * f (x - 1);
val a = 42;
val b = f a;
;;; val c = 4;;;;
f 4;
it + it;
val a = it;
```

(c)

```
val a = let
            fun sq (x: int) = x * x;
            val it = sq 42;
        in
            it
        end;

val b = a
```

Testen Sie Ihre Lösungen zur Überprüfung in SOSML, diskutieren Sie im Forum oder fragen Sie in der Office Hour eine(n) Tutor:in Ihrer Wahl.