3.6 Syntax matters

Functions are defined by one or more equations. In the simplest case, each function is defined by one (possibly conditional) equation:

\[ f \ x_1 \ldots x_n \]
\[ \text{if } \text{test}_1 = e_1 \\
\vdots \\
\text{if } \text{test}_n = e_n \]

Each right-hand side \( e_j \) is an expression.

Note: \( \text{otherwise} = \text{True} \)
3.6 Syntax matters

Functions are defined by one or more equations. In the simplest case, each function is defined by one (possibly conditional) equation:

\[
\begin{align*}
f & \ x_1 \ \ldots \ \ x_n \\
\mid & \ \ \text{test}_1 \ = \ e_1 \\
\vdots \\
\mid & \ \ \text{test}_n \ = \ e_n
\end{align*}
\]

Each right-hand side \(e_i\) is an expression.

Note: otherwise = True

Function and parameter names must begin with a lower-case letter.

An expression can be

- a literal like 0 or "xyz",
- or an identifier like True or x,
An expression can be
  • a literal like 0 or "xyz",
  • or an identifier like True or x,
  • or a function application \( f e_1 \ldots e_n \)
    where \( f \) is a function and \( e_1 \ldots e_n \) are expressions,
  • or a parenthesized expression (\( e \))

Local definitions: where

A defining equation can be followed by one or more local definitions.

\[\text{pow4 } x = x \times x \times x \times x \quad \text{where } x^2 = x \times x\]
Local definitions: where

A defining equation can be followed by one or more local definitions.

\[
\text{pow4 } x \quad = \quad x2 \times x2 \text{ where } x2 = x \times x
\]

\[
\text{pow4 } x \quad = \quad \text{sq } (\text{sq } x) \text{ where } \text{sq } x = x \times x
\]

Local definitions: let

Local definitions: let
Local definitions: let

```
let x = e₁ in e₂
```
defines \(x\) locally in \(e₂\)

Example:
```
let x = 2+3 in x^2 + 2*x
= 35
```

Local definitions: let

```
let x = e₁ in e₂
```
defines \(x\) locally in \(e₂\)

Example: concat

```
concat xss = [x | xs <- xss, x <- xs]

concat [[1,2], [4,5,6]]
= [x | xs <- [[1,2], [4,5,6]], x <- xs]
= [x | x <- [1,2]] ++ [x | x <- [4,5,6]]
= [1,2] ++ [4,5,6]
= [1,2,4,5,6]
```

What is the type of concat?
```
[[a]] -> [a]
```