Chapter 15 Functional Programming

Values Functions Extensionality Currying Definitions

Values

- An expression is used to describe (or denote) a value.
 - Among the kinds of value are: numbers of various kinds, truth values, characters, tuples, functions, and lists.
 - New kinds of value can be introduced.
- The evaluator prints a value by printing its canonical representation.
 - Some values have no canonical representation (i.e. function values).
 - Other values are not finite (i.e. Π)
 Chapter 15: Functional Programming

Values

Chapter 15: Functional Programming

- For some expressions the process of reduction never stops and never produces any result (i.e. the expression infinity).
- Some expressions do not denote well-defined values in the normal mathematical sense (i.e. the expression 1/0).
- Every syntactically well-formed expression denotes a value.
 - A special symbol ⊥ (bottom) stands for the undefined value of a particular type Chapter 15: Functional Programming

Values

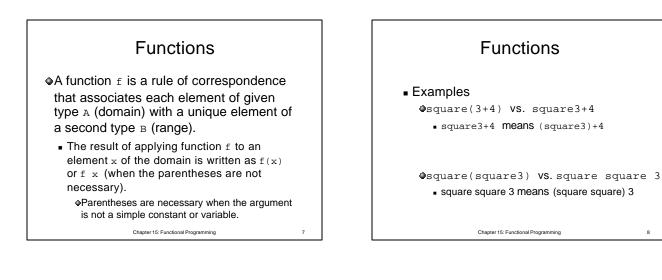
- The value of infinity is the undefined value \perp or type Integer.
- 1/0 is the undefined value \perp or type Float •1/0 = \perp
- The computer is not able to produce the value ⊥.
 If generates an error message or it remains perpetually silent.
- ⊥ is a special value that can be added to the universe of values only if its properties and its relationship with other values are precisely stated.

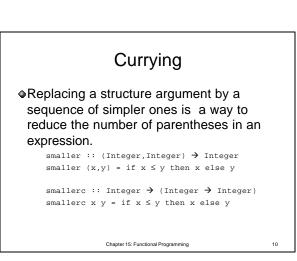
Chapter 15: Functional Programming

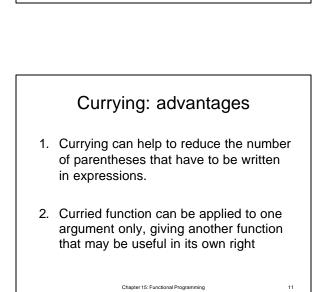
Values

- If $f \perp = \perp$, then f is *strict*, otherwise it is *nonstrict*
- square is a strict function because the evaluation of the undefined value goes into an infinite reduction (i.e. ? square infinity)
- three is nonstrict because the evaluation of the undefined value is 3 (i.e. ? three infinity)

Chapter 15: Functional Programming







Extensionality

Two functions are equal if they give equal

• This is called the principle of *extensionality*.

double, double' :: Integer \rightarrow Integer

double and double' defines the same

Chapter 15: Functional Programming

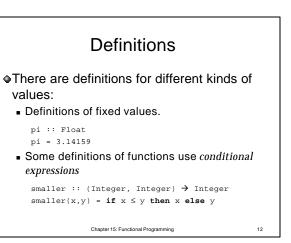
functional value, double = double'

results for equal arguments.

Example:

double x = x + xdouble' x = 2 * x

• f = g if an only if f x = g x for all x



Definitions

• The same expressions can be defined using *guarded equations.*

The main advantage of guarded expressions is when there are three or more clauses in a definition. Chapter 15: Functional Programming

13