COMP1100: Programming as Problem Solving

Slides 2: Functions and Sets

Dr. Liam O'Connor

(based on material from Ranald Clouston, Yun Kuen Cheung, and Michael Norrish)

School of Computing, Australian National University

Semester 2 2025

Programming with Functions

Functional Programming

This course teaches a style of programming called functional programming.

Programming with Functions

Functional Programming

This course teaches a style of programming called functional programming.

Functions

Functions are mappings between sets.

Sets

Sets

A set is a collection of things, called its elements.

- Not in any particular order
$$\{3, 4, 8\} = \{8, 3, 4\}$$
- Each element appears only once
$$\{1, 1, 2\} = \{1, 2\}$$

Sets

Sets

A set is a collection of things, called its elements.

- Not in any particular order

$$\{3,4,8\} = \{8,3,4\}$$

- Each element appears only once

$$\{1,1,2\} = \{1,2\}$$

Finite Sets

Some sets have only finitely many elements, so can be listed:

- A singleton, e.g. $\{3\}$ or $\{\text{True}\}$.
- The booleans $\mathbb{B}$: $\{\text{True}, \text{False}\}$

Large finite sets may not be practical to list out fully, so we use ranges:

- A set of characters: $\{\texttt{'A'}, \texttt{'B'}, \ldots, \texttt{'Z'}\}$.
- A bigger set of numbers: $\{0, 1, 2, \ldots, 100\}$.

Sets

Sets

A set is a collection of things, called its elements.

- Not in any particular order

$$\{3, 4, 8\} = \{8, 3, 4\}$$

- Each element appears only once

$$\{1, 1, 2\} = \{1, 2\}$$

Finite Sets

Some sets have only finitely many elements, so can be listed:

- A singleton, e.g. $\{3\}$ or $\{\text{True}\}$.
- The booleans $\mathbb{B}$: $\{\text{True}, \text{False}\}$

Large finite sets may not be practical to list out fully, so we use ranges:

- A set of characters: $\{\texttt{'A'}, \texttt{'B'}, \dots, \texttt{'Z'}\}$.
- A bigger set of numbers: $\{0, 1, 2, \dots, 100\}$.

Infinite Sets

- The natural numbers $\mathbb{N}$: $\{0, 1, 2, \dots\}$
- The integers $\mathbb{Z}$: $\{\dots, -1, 0, 1, 2, \dots\}$
- The real numbers $\mathbb{R}$
- The strings: lists of characters of any finite length.

Combining Sets: Product

Definition

Given two sets A and B, we can form a new set $A \times B$, called the product (or cartesian product), by including every pair (a, b) where a is an element of A and b is an element of B.

Combining Sets: Product

Definition

Given two sets A and B, we can form a new set $A \times B$, called the product (or cartesian product), by including every pair (a, b) where a is an element of A and b is an element of B.

Example

$\mathbb{B} \times \{\text{Red}, \text{Green}, \text{Blue}\}$ is:

$$\{(\text{False}, \text{Red}), (\text{False}, \text{Green}), (\text{False}, \text{Blue}), \\ (\text{True}, \text{Red}), (\text{True}, \text{Green}), (\text{True}, \text{Blue})\}$$

Combining Sets: Product

Definition

Given two sets A and B, we can form a new set $A \times B$, called the product (or cartesian product), by including every pair (a, b) where a is an element of A and b is an element of B.

Example

$\mathbb{B} \times \{\text{Red}, \text{Green}, \text{Blue}\}$ is:

$$\{(\text{False}, \text{Red}), (\text{False}, \text{Green}), (\text{False}, \text{Blue}), \\ (\text{True}, \text{Red}), (\text{True}, \text{Green}), (\text{True}, \text{Blue})\}$$

Products can extend to more than two sets:

$$A \times B \times C \times D$$

whose members have the form (a, b, c, d).

e.g. $\mathbb{B} \times \{\text{Red}, \text{Green}, \text{Blue}\} \times \mathbb{B}$?

Combining Sets: Sum

Definition

Given two sets A and B, we can form a new set $A + B$, called the sum, by combining:

- Every element a of A, along with a 'tag' to remind us that it came from the left:

$$(a, \text{Left})$$

- Every element b of B, tagged right:

$$(a, \text{Right})$$

Combining Sets: Sum

Definition

Given two sets A and B, we can form a new set $A + B$, called the sum, by combining:

- Every element a of A, along with a 'tag' to remind us that it came from the left:

$$(a, \mathrm{Left})$$

- Every element b of B, tagged right:

$$(a, \mathrm{Right})$$

Example: $\mathbb{B} + \{\textsf{Red}, \textsf{Green}, \textsf{Blue}\}$

$$\{(\textsf{False}, \mathrm{Left}), (\textsf{True}, \mathrm{Left}), \\ (\textsf{Red}, \mathrm{Right}), (\textsf{Green}, \mathrm{Right}), (\textsf{Blue}, \mathrm{Right})\}$$

Combining Sets: Sum

Definition

Given two sets A and B, we can form a new set $A + B$, called the sum, by combining:

- Every element a of A, along with a 'tag' to remind us that it came from the left:

$$(a, \mathrm{Left})$$

- Every element b of B, tagged right:

$$(a, \mathrm{Right})$$

Question

Why do we need the tags?

Example: $\mathbb{B} + \{\textsf{Red}, \textsf{Green}, \textsf{Blue}\}$

$$\{(\textsf{False}, \mathrm{Left}), (\textsf{True}, \mathrm{Left}), (\textsf{Red}, \mathrm{Right}), (\textsf{Green}, \mathrm{Right}), (\textsf{Blue}, \mathrm{Right})\}$$

Combining Sets: Sum

Definition

Given two sets A and B, we can form a new set $A + B$, called the sum, by combining:

- Every element a of A, along with a 'tag' to remind us that it came from the left:
$$(a, \text{Left})$$
- Every element b of B, tagged right:
$$(a, \text{Right})$$

Question

Why do we need the tags?

We want our sets to remain entirely disjoint. Consider $\mathbb{B} + \mathbb{B}$.

Note: $A + B$ is therefore different from set union $A \cup B$!

Example: $\mathbb{B} + \{\text{Red}, \text{Green}, \text{Blue}\}$

$$\{(\text{False}, \text{Left}), (\text{True}, \text{Left}),\\ (\text{Red}, \text{Right}), (\text{Green}, \text{Right}), (\text{Blue}, \text{Right})\}$$

Combining Sets: Sum

Definition

Given two sets A and B, we can form a new set $A + B$, called the sum, by combining:

- Every element a of A, along with a 'tag' to remind us that it came from the left:
$$(a, \mathrm{Left})$$
- Every element b of B, tagged right:
$$(a, \mathrm{Right})$$

Example: $\mathbb{B} + \{\text{Red}, \text{Green}, \text{Blue}\}$

$$\{(\text{False}, \mathrm{Left}), (\text{True}, \mathrm{Left}),$$
$$(\text{Red}, \mathrm{Right}), (\text{Green}, \mathrm{Right}), (\text{Blue}, \mathrm{Right})\}$$

Question

Why do we need the tags?

We want our sets to remain entirely disjoint. Consider $\mathbb{B} + \mathbb{B}$.

Note: $A + B$ is therefore different from set union $A \cup B$!

Sums can extend to more than two sets:
$$A + B + C + D$$

Although we had better pick better names for tags than Left and Right!

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

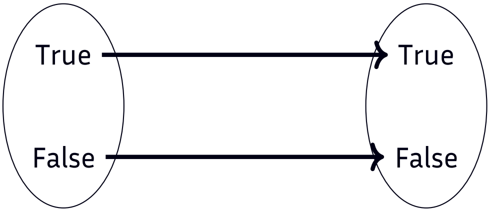
$$f(a) = b$$

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).



Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

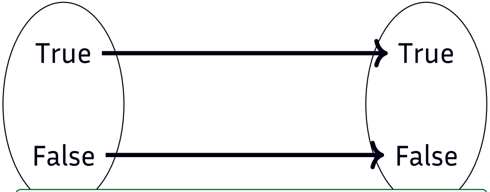
Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$



This is the identity function $\mathbb{B} \longrightarrow \mathbb{B}$.

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

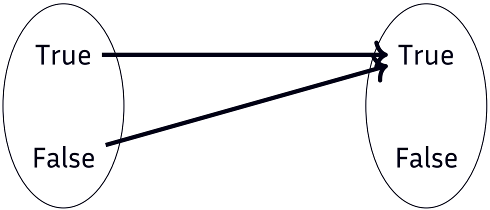
Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$



Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

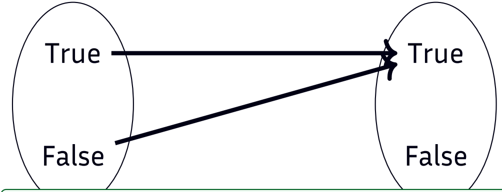
Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$



The constant function returning True.

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$

This is not a total function — it is partial

Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

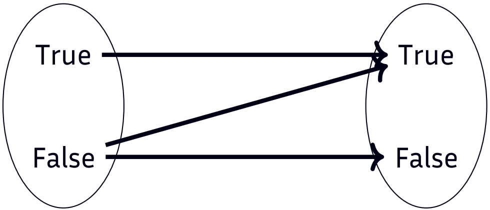
Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$



Functions

Definition

A function is defined by:

- A set A, called the domain;
- A set B, called the codomain (or range);
- An assignment from *each* element of A (input) to *one* element of B (output).

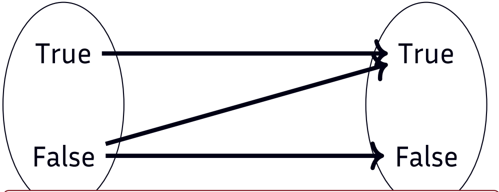
Notation

Giving such a function a name f, we write:

$$f :: A \to B$$

If f assigns output b to input a, we write:

$$f(a) = b$$



This is not a function (but a relation)

Defining Functions

Finite Domains

If the domain is finite, we can define a function by listing its output for each possible input.

For example, define:

$$
\begin{aligned}
&f :: \mathbb{B} \to \mathbb{Z} \\
&f(\mathrm{False}) = -6 \\
&f(\mathrm{True}) = 26
\end{aligned}
$$

Defining Functions

Finite Domains

If the domain is finite, we can define a function by listing its output for each possible input.

For example, define:

$$\begin{aligned} f &:: \mathbb{B} \to \mathbb{Z} \\ f(\text{False}) &= -6 \\ f(\text{True}) &= 26 \end{aligned}$$

Infinite (or larger finite) Domains

For larger domains, we must explain how to compute the output for all inputs:

$$\begin{aligned} &minus :: \mathbb{Z} \to \mathbb{Z} \\ &minus(x) = -x \end{aligned}$$

$$isPos :: \mathbb{Z} \to \mathbb{B}$$

$$isPos(y) = \begin{cases} \text{True} & \text{if } y \text{ is positive} \\ \text{False} & \text{if } y \text{ is negative} \\ \text{False} & \text{if } y = 0 \end{cases}$$

Here x and y are variables, which stand for any element of $\mathbb{Z}$.

Polymorphic Functions

Definition

A function that can be defined *simultaneously* for many different sets is called polymorphic.

- The identity function:

$$\begin{aligned} id &:: A \to A \\ id(a) &= a \end{aligned}$$

- The constant zero function:

$$\begin{aligned} cz &:: B \to \mathbb{Z} \\ cz(b) &= 0 \end{aligned}$$

Here A and B are variables which stand for any *set* — and a and b are variables which stand for arbitrary elements of those sets.

Polymorphic Functions

Definition

A function that can be defined *simultaneously* for many different sets is called polymorphic.

- The identity function:

$$id :: A \to A$$
$$id(a) = a$$

- The constant zero function:

$$cz :: B \to \mathbb{Z}$$
$$cz(b) = 0$$

Here A and B are variables which stand for any *set* — and a and b are variables which stand for arbitrary elements of those sets.

Examples

Let's define these together:

- $proj_L :: A \times B \to A$
- $proj_R :: A \times B \to B$
- $inj_L :: A \to A + B$
- $inj_R :: B \to A + B$

Polymorphic Functions

Definition

A function that can be defined *simultaneously* for many different sets is called polymorphic.

- The identity function:

$$id :: A \to A$$
$$id(a) = a$$

- The constant zero function:

$$cz :: B \to \mathbb{Z}$$
$$cz(b) = 0$$

Here A and B are variables which stand for any *set* — and a and b are variables which stand for arbitrary elements of those sets.

Bounded Polymorphism

Is this function polymorphic?

$$minus :: A \to A$$
$$minus(x) = -x$$

Polymorphic Functions

Definition

A function that can be defined *simultaneously* for many different sets is called polymorphic.

- The identity function:

$$id :: A \to A$$
$$id(a) = a$$

- The constant zero function:

$$cz :: B \to \mathbb{Z}$$
$$cz(b) = 0$$

Here A and B are variables which stand for any *set* — and a and b are variables which stand for arbitrary elements of those sets.

Bounded Polymorphism

Is this function polymorphic?

$$minus :: A \to A$$
$$minus(x) = -x$$

Defined for more than one set, e.g. $\mathbb{Z}, \mathbb{R}$, but not defined for others, e.g. $\mathbb{N}, \mathbb{B}$.

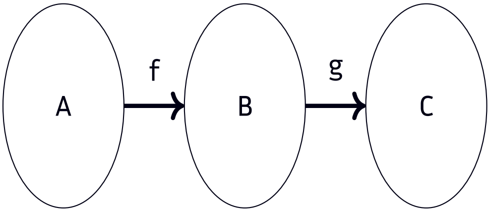
This function is polymorphic *only for sets where* $(-)$ is defined.

Combining Functions: Composition

Definition

If we have functions $f :: A \to B$ and $g :: B \to C$, we define a new function $g \circ f :: A \to C$ by

$$(g \circ f)(a) = g(f(a))$$

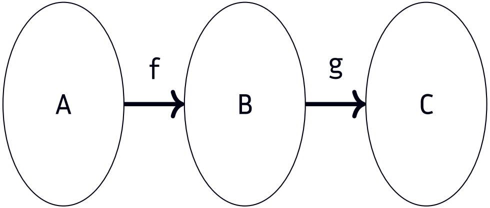


Combining Functions: Composition

Definition

If we have functions $f :: A \to B$ and $g :: B \to C$, we define a new function $g \circ f :: A \to C$ by

$$(g \circ f)(a) = g(f(a))$$



Examples

Earlier we defined $minus :: \mathbb{Z} \to \mathbb{Z}$ and $isPos :: \mathbb{Z} \to \mathbb{B}$.

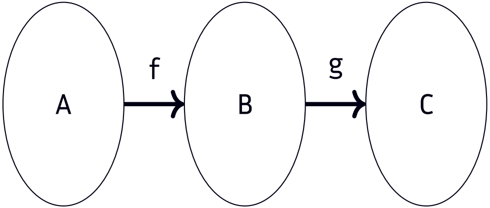
- Can we define $minus \circ isPos$ or $isPos \circ minus$?

Combining Functions: Composition

Definition

If we have functions $f :: A \to B$ and $g :: B \to C$, we define a new function $g \circ f :: A \to C$ by

$$(g \circ f)(a) = g(f(a))$$



Examples

Earlier we defined $minus :: \mathbb{Z} \to \mathbb{Z}$ and $isPos :: \mathbb{Z} \to \mathbb{B}$.

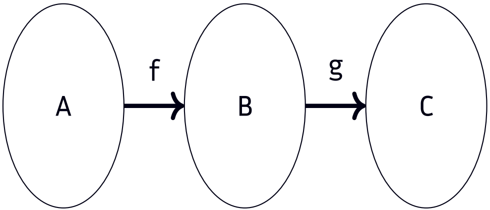
- Can we define $minus \circ isPos$ or $isPos \circ minus$?
- For the one we can define, what is the domain and codomain?

Combining Functions: Composition

Definition

If we have functions $f :: A \to B$ and $g :: B \to C$, we define a new function $g \circ f :: A \to C$ by

$$(g \circ f)(a) = g(f(a))$$



Examples

Earlier we defined $minus :: \mathbb{Z} \to \mathbb{Z}$ and $isPos :: \mathbb{Z} \to \mathbb{B}$.

- Can we define $minus \circ isPos$ or $isPos \circ minus$?
- For the one we can define, what is the domain and codomain?
- Can you describe in words what this combined function does?

Combining Sets: Sets of Functions

The Space of Functions

Until now we have thought of elements of sets, and functions, as different things.

But given sets A and B, we can form a new set $A \to B$ — the set of *all possible functions* from A to B.

There is no difference between writing:

$$f :: A \to B$$

and saying:

"f is an element of the set $A \to B$"

Combining Sets: Sets of Functions

The Space of Functions

Until now we have thought of elements of sets, and functions, as different things.

But given sets A and B, we can form a new set $A \to B$ — the set of *all possible functions* from A to B.

There is no difference between writing:

$$f :: A \to B$$

and saying:

"f is an element of the set $A \to B$"

The codomain of a function can itself consist of functions:

$$pair :: A \to (B \to A \times B)$$
$$(pair(a))(b) = (a, b)$$

And the domain of a function can itself consist of functions:

$$doTwice :: (A \to A) \to (A \to A)$$
$$(doTwice(f))(a) = f(f(a))$$

Combining Sets: Sets of Functions

The Space of Functions

Until now we have thought of elements of sets, and functions, as different things.

But given sets A and B, we can form a new set $A \to B$ — the set of *all possible functions* from A to B.

There is no difference between writing:

$$f :: A \to B$$

and saying:

"f is an element of the set $A \to B$"

The codomain of a function can itself consist of functions:

$$pair :: A \to (B \to A \times B)$$
$$(pair(a))(b) = (a, b)$$

And the domain of a function can itself consist of functions:

$$doTwice :: (A \to A) \to (A \to A)$$
$$(doTwice(f))(a) = f(f(a))$$

Right-associativity

$$A \to B \to C = A \to (B \to C)$$

Which of the above brackets are unneeded?

Combining Sets: Sets of Functions

The Space of Functions

Until now we have thought of elements of sets, and functions, as different things.

But given sets A and B, we can form a new set $A \to B$ — the set of *all possible functions* from A to B.

There is no difference between writing:

$$f :: A \to B$$

and saying:

"f is an element of the set $A \to B$"

The codomain of a function can itself consist of functions:

$$pair :: A \to B \to A \times B$$
$$(pair(a))(b) = (a, b)$$

And the domain of a function can itself consist of functions:

$$doTwice :: (A \to A) \to A \to A$$
$$(doTwice(f))(a) = f(f(a))$$

Right-associativity

$$A \to B \to C = A \to (B \to C)$$

Which of the above brackets are unneeded?

From Mathematics to Programming

The Space of Functions

- Syntax differs between standard mathematics and any given programming language, e.g. `f x` in Haskell instead of $f(x)$.
- Mathematicians are comfortable with infinite constructions, such as real numbers, but programmers are restricted to finite memory.
- Mathematical functions assign an output to every input — programmed functions may crash, or get stuck in a loop.
- Mathematical functions are defined entirely by their association of output to input; in programming one mathematically identical function may be better than another, e.g. it may run faster.

Haskell