# 1 Basic Theories

## 1.1 Boolean Theory

**Operators** Some boolean operators are supported by LaTeX, but they have names suggesting shape rather than content, e.g., `a \Rightarrow b`. It would be nice if they were given informative, short names without clashing with existing LaTeX commands.

| | |
|---|---|
| $a \Rightarrow b$ | `a \imp b` |
| $a \implies b$ | `a \Imp b` |
| $a \Leftarrow b$ | `a \pmi b` or `a \impby b` |
| $a \impliedby b$ | `a \Pmi b` or `a \Impby b` |
| $a = b$ | `a \Eq b` |
| **if** $c$ **then** $x$ **else** $y$ **fi** | `\cond{c}{x}{y}` |

Other boolean symbols ($=$, `\bot` for $\perp$, `\lor` for $\vee$, etc.) have reasonable names in LaTeX, and I will not show them.

Two more variants of if-then-else-fi:

- `\condb{c}{x}{y}`:

  **if** $c$ **then** $x$
  **else** $y$ **fi**

- `\condbb{c}{x}{y}`:

  **if** $c$ **then**
  $x$
  **else**
  $y$
  **fi**

**Proof format** Using the `align*` environment provided by $\mathcal{AMS}$-LaTeX(package name amsmath), a calculational proof with hints can be typeset easily. The first proof in the textbook:

$$
\begin{aligned}
& a \wedge b \Rightarrow c && \text{Material Implication} \\
= \;& \neg(a \wedge b) \vee c && \text{Duality} \\
= \;& \neg a \vee \neg b \vee c && \text{Material Implication} \\
= \;& a \Rightarrow \neg b \vee c && \text{Material Implication} \\
= \;& a \Rightarrow (b \Rightarrow c)
\end{aligned}
$$

Its code:

```
\begin{align*}
&\Blank a \et b \imp c          && \text{Material Implication} \\
&\Eq  \neg(a \et b) \vel c      && \text{Duality} \\
&\Eq  \neg a \vel \neg b \vel c && \text{Material Implication} \\
```

```
&\Eq  a \imp \neg b \vel c      && \text{Material Implication} \\
&\Eq  a \imp (b \imp c)
\end{align*}
```

The command `\Blank` is a blank relation symbol I invented; it is necessary in that position to keep `align*` happy. Its definition is simply: `\mathrel{\phantom{\Eq}}`.

# 2 Basic Data Structures

**Bunch Theory**

| | |
|---|---|
| $A, B$ | `A,B` |
| $A$‘$B$ | ``A`B`` |
| $null$ | `\nul` |
| ¢$A$ | `\card A` |
| $0,..10$ | `0 \bto 10` |
| $nat, xnat$ | `\nat, \xnat` |
| $int, xint$ | `\int, \xint` |
| $rat, xrat$ | `\rat, \xrat` |

**String Theory**

| | |
|---|---|
| $nil$ | `\nil` |
| $n^*S$ | `n^*S` |
| ${}^*S$ | `{}^*S` |
| $0;..10$ | `0 \sto 10` |

**List Theory**

| | |
|---|---|
| $L^+M$ | `L^+M` |
| $n \to i \mid L$ | `n \to i \ow L` |
| $L\,n$ | `L \ap n` |

# 3 Function Theory

The `\fun` and `\fn` commands produce functions; `\fun` requires a domain and `\fn` omits the domain.

| | |
|---|---|
| $\langle x\colon nat \to x+1\rangle$ | `\fun{x}{\nat}{x+1}` |
| $\langle x \to x+1\rangle$ | `\fn{x}{x+1}` |

The `\bind` and `\bnd` commands help you produce quantified expressions. They just have the quantifier missing, and you just put it back. `\bind` requires a domain and `\bnd` omits the domain. Some examples:

| | |
|---|---|
| $\forall x \cdot x = x$ | `\forall\bnd{x}{x=x}` |
| $\Sigma i\colon 0,..10 \cdot i^2$ | `\Sigma\bind{i}{0 \bto 10}{i^2}` |
| $\S x\colon nat \cdot x/2 : nat$ | `\S\bind{x}{\nat}{x/2:\nat}` |

Two quantifiers are not already available in LaTeX: *MAX* and *MIN*. I have defined them as `\MAX` and `\MIN`, respectively.

Both application and composition are `\ap`. You can think of it as standing for "apposition". Selective union is `\ow`, standing for "otherwise". You have seen them in List Theory. More examples:

| | |
|---|---|
| $MAX\, v\colon x \cdot n$ | `\MAX\bind{v}{x}{n}` |
| $MIN\, v\colon x \cdot n$ | `\MIN\bind{v}{x}{n}` |
| $f \mid g$ | `f \ow g` |
| $h\, f\, x\, g\, y$ | `h \ap f \ap x \ap g \ap y` |

# 4 Program Theory

| | |
|---|---|
| $ok$ | `\ok` |
| $S \,.\, R$ | `S \dc R` |
| $x{:=}\,e$ | `x \get e` |

# 5 Programming Language

Two forms of while-do-od:

- `\while{c}{P}`:

  **while** $c$ **do** $P$ **od**

- `\whileb{c}{P}`:

  **while** $c$
  **do** $P$ **od**