1811/2807/7001ICT
Programming Principles

School of Information and Communication Technology
Griffith University

Trimester 1, 2024

9 Booleans

In this section we will introduce the kinds of symbols used in mathematics and logic to:

- represent yes/no or true/false values;
- compare values and thereby create true/false values;
- combine true/false values.

This will help you to write programs that do more than follow simple sequences.

9.1 Boolean values

We call the logical values True and False *Booleans* after George Boole, who invented *Boolean algebra*, that is reasoning and logic with symbols, not with just words.

symbol	*means*
F	false/no
T	true/yes

9.2 Relational operators

We use relational operators to compare values and thereby create Boolean values.

The values we can compare include all types (for example numbers and Booleans themselves), so long as we compare things of the same type (for example, numbers with numbers, but not a number with a Boolean).

symbol	*means*	*symbol*	*means*
$<$	less than	$>$	greater than
$\leq$	less than or equal to	$\geq$	greater than or equal to
$=$	equal to	$\neq$	not equal to

For example, $x > 0$ is an expression with a Boolean value.

More examples: If x contains 3 and y contains −3...

expression	*value*
$x < 10$	T
$y \leq 10$	T
$x < y$	F
$x + y = 0$	T
$x \neq y$	T
$x \geq y$	T
$x \leq y$	F

9.3 Boolean operators

Boolean operators are symbols used in Boolean algebra to combine Boolean values.

Typically we want to:

- negate a value (change T to F or F to T);
- see if both of two values are T;
- see if at least one of two values are T;
- see if only one of two values are T.

operator	*means*	*examples*
$\neg$	not	$\neg\mathrm{T}$, $\neg p$, $\neg(x < 10)$
$\wedge$	and	$\mathrm{T} \wedge \mathrm{F}$, $p \wedge q$, $x < 10 \wedge y = 0$
$\vee$	or	$\mathrm{T} \vee \mathrm{F}$, $p \vee q$, $x < 10 \vee y = 0$
xor	*exclusive* or	$\mathrm{T}\,\mathrm{xor}\,\mathrm{F}$, $p\,\mathrm{xor}\,q$, $x < 10\,\mathrm{xor}\,y = 0$

These *truth tables* show precisely how these operators combine Boolean values.

p	$\neg p$
F	T
T	F

p	q	$p \wedge q$
F	F	F
F	T	F
T	F	F
T	T	T

p	q	$p \vee q$
F	F	F
F	T	T
T	F	T
T	T	T

p	q	$p \operatorname{xor} q$
F	F	F
F	T	T
T	F	T
T	T	F

Note that when we say "or" in English we most often mean xor, but what we will use most often in programs is $\vee$.

$p \operatorname{xor} q$ is the same as $(p \wedge \neg q) \vee (\neg p \wedge q)$.

9.3.1 Precedence

Relational and Boolean operators can be mixed with arithmetic operators in expressions.

precedence	*operations*
highest	*unary* + and −, ¬
	×, div, ÷, /, mod
	binary + and −
	<, ≤, >, ≥, =, ≠
	∧
lowest	∨, xor

Examples: If x contains 3 and y contains −3...

expression	*value*
$\neg(x < 10)$	F
$x > 10 \wedge y \leq 10$	F
$x > 10 \vee y \leq 10$	T

Section summary

This section covered:

- the Boolean values T and F;
- the relational operators that can create Boolean values;
- the Boolean operators (not, and, or, exclusive or); and
- the precedence of Boolean and relational operators relative to the arithmetic operators.