Ada Programming/All Operators

Aus Wikibooks
< Ada Programming

All bookshelves > Computer science > Programming languages > **Ada Programming**

This Wikibook was voted **Book of the Month** for **September 2005**!

Contents

Augusta Ada King, Countess of Lovelace.

1 Operators

1.1 Standard operators

Ada allows operator overloading for all standard operators and so the following summaries can only describe the suggested standard operations for each operator. It is quite possible to misuse any standard operator to perform something unusual.

Each operator is either a keyword or a delimiter -- hence all operator pages are redirects to the appropiate keyword or delimiter.

The list is sorted from lowest precedence to highest precedence.

1.1.1 Logical operators

and
: and $x \wedge y$, (also keyword and)

or
: or $x \vee y$, (also keyword or)

xor
: exclusive or $(x \wedge \bar{y}) \vee (\bar{x} \wedge y)$, (also keyword xor)

1.1.2 Relational operators

/=
: Not Equal $x \neq y$, (also special character /=)

=

Equal $x = y$, (also special character =)

<

Less than $x < y$, (also special character <)

<=

Less than or equal to ($x \leq y$), (also special character <=)

>

Greater than ($x > y$), (also special character >)

>=

Greater than or equal to ($x \geq y$), (also special character >=)

1.1.3 Binary adding operators

+

Add $x + y$, (also special character +)

-

Subtract $x - y$, (also special character -)

&

Concatenate , x & y, (also special character &)

1.1.4 Unary adding operators

+

Plus sign + x, (also special character +)

-

Minus sign - x, (also special character -)

1.1.5 Multiplying operator

*

Multiply, $x \times y$, (also special character *)

/

Divide x / y, (also special character /)

mod

modulus (also keyword mod)

rem

remainder (also keyword rem)

1.1.6 Highest precedence operator

**

Power x^y, (also special character **)

not

logical not $\neg x$, (also keyword not)

abs

absolute value $|x|$ (also keyword abs)

1.2 Shortcut operators

The shortcut operators cannot be overloaded.

and then

e.g. `if Y /= 0 and then X/Y > Limit then ...`

or else

e.g. `if Ptr = null or else Ptr.I = 0 then ...`

1.3 Membership tests

The Membership Tests also cannot be overloaded.

in
: element of, $var \in type$, *e.g.* `if I in Positive then`, (also keyword **in**)

not in
: not element of, $var \notin type$, *e.g.* `if I not in Positive then`, (also keywords **not in**)

1.4 See also

1.4.1 Wikibook

- Ada Programming

1.4.2 Ada 95 Reference Manual

- 4.5 Operators and Expression Evaluation (http://www.adaic.com/standards/95lrm/html/RM-4-5.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5.html))

1.4.3 Ada 2005 Reference Manual

- 4.5 Operators and Expression Evaluation (http://www.adaic.com/standards/05rm/html/RM-4-5.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5.html))

1.4.4 Ada Quality and Style Guide

- 2.1.3 Alignment of Operators (http://www.adaic.org/docs/95style/html/sec_2/2-1-3.html)
- 5.7.4 Overloaded Operators (http://www.adaic.org/docs/95style/html/sec_5/5-7-4.html)
- 5.7.5 Overloading the Equality Operator (http://www.adaic.org/docs/95style/html/sec_5/5-7-5.html)

Ada Operators

`and`	`and then`	`>`	`+`	`abs`	`&`
`or`	`or else`	`>=`	`-`	`mod`	
`xor`	`=`	`<`	`*`	`rem`	`in`
`not`	`/=`	`<=`	`**`	`/`	`not in`

2 Operators: &

2.1 As operator

2.1.1 Concatenating arrays

```
function "&" (Left, Right : T) return T;
```

Any array type (including fixed Strings) can be concatenated using the **&** operator. You can also append a single element to an array.

2.2 Common non-standard operations

2.2.1 Concatenating strings

The **&** operator is also defined for Bounded_String and Unbounded_String.

2.3 See also

2.3.1 Wikibook

- Ada Programming/Delimiters
- Ada Programming/Operators

2.3.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.3 Binary Adding Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-3.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-3.html))
- A.4.4 Bounded-Length String Handling (http://www.adaic.com/standards/95lrm/html/RM-A-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-A-4-4.html))
- A.4.5 Unbounded-Length String Handling (http://www.adaic.com/standards/95lrm/html/RM-A-4-5.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-A-4-5.html))

2.3.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.3 Binary Adding Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-3.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-3.html))
- A.4.4 Bounded-Length String Handling (http://www.adaic.com/standards/05rm/html/RM-A-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-A-4-4.html))
- A.4.5 Unbounded-Length String Handling (http://www.adaic.com/standards/05rm/html/RM-A-4-5.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-A-4-5.html))

3 Operators: **

3.1 Operator

3.1.1 Standard Operations

3.1.1.1 Arithmetic Power of

The "**" operator is defined as arithmetic power of for all numeric types.

```
function "**" (Left : T; Right : Integer) return T;
```

3.1.1.1.1 Usage

```
A : constant Float   := 5.0 ** 2;  -- A is now 25.0
B : constant Integer := 5 ** 2;    -- B is also 25
```

3.1.1.1.2 Working Example

File: operator_power.adb (view (http://svn.sourceforge.net/viewcvs.cgi/wikibook-ada/trunk/demos/Source/operator_power.adb?view=markup) , plain text (http://svn.sourceforge.net/viewcvs.cgi/*checkout*/wikibook-ada/trunk/demos/Source/operator_power.adb) , download page (https://sourceforge.net/project/showfiles.php?group_id=124904) , browse all (http://wikibook-ada.sourceforge.net/html/index.html))

```
with Ada.Text_IO;

procedure Operator_Power is
   A : constant Float   := 5.0 ** 2;  -- A is now 25.0
   B : constant Integer := 5 ** 2;    -- B is also 25

   package T_IO renames Ada.Text_IO;
   package F_IO is new  Ada.Text_IO.Float_IO (Float);
   package I_IO is new  Ada.Text_IO.Integer_IO (Integer);

begin
   T_IO.Put ("A = ");
   F_IO.Put (
      Item => A,
      Fore => 3,
      Aft  => 1,
      Exp  => 0);
   T_IO.New_Line;
   T_IO.Put ("B = ");
   I_IO.Put (
      Item  => B,
      Width => 3,
      Base  => 10);
   T_IO.New_Line;
end Operator_Power;
```

3.2 See also

3.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters
- Ada Programming/Operators

3.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.6 Highest Precedence Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-6.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-6.html))

3.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.6 Highest Precedence Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-6.html) (Annotated

(http://www.adaic.com/standards/05aarm/html/AA-4-5-6.html))

4 Operators: *

4.1 Operator

4.1.1 Standard Operations

4.1.1.1 Arithmetic Multiplication

The "*" operator is defined as arithmetic multiplication for all numeric types.

```
function "*" (Left, Right : T) return T;
```

4.1.1.1.1 Usage

```
A : constant Float   := 5.0 * 2.0;  -- A is now 10.0
B : constant Integer := 5 * 2;      -- B is also 10
```

4.1.1.1.2 Working Example

File: operator_multiply.adb (view (http://svn.sourceforge.net/viewcvs.cgi/wikibook-ada/trunk/demos/Source/operator_multiply.adb?view=mar , plain text (http://svn.sourceforge.net/viewcvs.cgi/*checkout*/wikibook-ada/trunk/demos/Source/operator_multiply.adb , download page (https://sourceforge.net/project/showfiles.php?group_id=124904) , browse all (http://wikibook-ada.sourceforge.net/html/index.html))

```
with Ada.Text_IO;

procedure Operator_Multiply is
   A : constant Float   := 5.0 * 2.0;  -- A is now 10.0
   B : constant Integer := 5 * 2;      -- B is also 10

   package T_IO renames Ada.Text_IO;
   package F_IO is new  Ada.Text_IO.Float_IO (Float);
   package I_IO is new  Ada.Text_IO.Integer_IO (Integer);

begin
   T_IO.Put ("A = ");
   F_IO.Put (
      Item => A,
      Fore => 3,
      Aft  => 1,
      Exp  => 0);
   T_IO.New_Line;
   T_IO.Put ("B = ");
   I_IO.Put (
      Item  => B,
      Width => 3,
      Base  => 10);
   T_IO.New_Line;
end Operator_Multiply;
```

4.1.2 Common Non-Standard Operations

4.1.2.1 Character replication

A String is created where a single character is replicated n-times.

```
function "*" (Left : Natural; Right : Character) return String;
```

In addition to standard Strings this operator is also defined for Bounded_String and Unbounded_String.

4.1.2.1.1 Usage

```
A : constant String := 10 * 'X';  -- A is filled with 10 X
```

4.1.2.1.2 Working Example

The character replication operator is part of the *Ada.Strings.Fixed* package. You need to **with** and **use** the package to make the operator visible.

File: operator_multiply_2.adb (view (http://svn.sourceforge.net/viewcvs.cgi/wikibook-ada/trunk/demos/Source/operator_multiply_2.adb?view=m , plain text (http://svn.sourceforge.net/viewcvs.cgi/*checkout*/wikibook-ada/trunk/demos/Source/operator_multiply_2.a , download page (https://sourceforge.net/project/showfiles.php?group_id=124904) , browse all (http://wikibook-ada.sourceforge.net/html/index.html))

```
with Ada.Text_IO;
with Ada.Strings.Fixed;

procedure Operator_Multiply_2 is
   use Ada.Strings.Fixed;

   A : constant String := 10 * 'X';  -- A is filled with 10 X

   package T_IO renames Ada.Text_IO;

begin
   T_IO.Put_Line ("A = " & A);
end Operator_Multiply_2;
```

4.1.2.2 String replication

A String is created where a source string is replicated n-times.

```
function "*" (Left : Natural; Right : String) return String;
```

In addition to standard fixed strings this operator is also defined for Bounded_String and Unbounded_String.

4.1.2.2.1 Usage

```
A : constant String := 3 * "Hello ";  -- A is filled with 3 Hello
```

4.1.2.2.2 Working Example

The string replication operator is part of the Ada.Strings.Fixed package. You need to **with** and **use** the package to make the operator visible.

File: operator_multiply_3.adb (view (http://svn.sourceforge.net/viewcvs.cgi/wikibook-ada/trunk/demos/Source/operator_multiply_3.adb?view=m , plain text (http://svn.sourceforge.net/viewcvs.cgi/*checkout*/wikibook-ada/trunk/demos/Source/operator_multiply_3.a , download page (https://sourceforge.net/project/showfiles.php?group_id=124904) , browse all (http://wikibook-ada.sourceforge.net/html/index.html))

```
with Ada.Text_IO;
with Ada.Strings.Fixed;

procedure Operator_Multiply_3 is
   use Ada.Strings.Fixed;

   A : constant String := 3 * "Hello ";  -- A is filled with 3 Hello.

   package T_IO renames Ada.Text_IO;

begin
   T_IO.Put_Line ("A = " & A);
end Operator_Multiply_3;
```

4.2 See also

4.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters
- Ada Programming/Operators

4.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.5 Multiplying Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-5.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-5.html))
- A.4.3 Fixed-Length String Handling (http://www.adaic.com/standards/95lrm/html/RM-A-4-3.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-A-4-3.html))
- A.4.4 Bounded-Length String Handling (http://www.adaic.com/standards/95lrm/html/RM-A-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-A-4-4.html))
- A.4.5 Unbounded-Length String Handling (http://www.adaic.com/standards/95lrm/html/RM-A-4-5.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-A-4-5.html))

4.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.5 Multiplying Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-5.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-5.html))
- A.4.3 Fixed-Length String Handling (http://www.adaic.com/standards/05rm/html/RM-A-4-3.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-A-4-3.html))
- A.4.4 Bounded-Length String Handling (http://www.adaic.com/standards/05rm/html/RM-A-4-4.html) (Annotated

(http://www.adaic.com/standards/05aarm/html/AA-A-4-4.html))
- A.4.5 Unbounded-Length String Handling (http://www.adaic.com/standards/05rm/html/RM-A-4-5.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-A-4-5.html))

5 Operators: -

5.1 Operator

5.1.1 Standard Operations

5.1.1.1 Arithmetic Subtraction

The "-" operator is defined as arithmetic subtraction for all numeric types.

```
function "-" (Left, Right : T) return T;
```

5.1.1.1.1 Usage

```
A : constant Float   := 5.0 - 2.0;  -- A is now 3.0
B : constant Integer := 5 - 2;      -- B is also 3
```

5.1.1.2 Minus sign

The "-" unary operator is defined as arithmetic negative sign for all numeric types.

```
function "-" (Left : T) return T;
```

5.1.1.2.1 Usage

```
A : constant Float   := -5.0;
B : constant Integer := -5;
C : constant Integer := -B;  -- C is now 5
```

5.2 See also

5.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters
- Ada Programming/Operators
- Ada Programming/Mathematical calculations

5.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.3 Binary Adding Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-3.html) (Annotated

(http://www.adaic.com/standards/95aarm/html/AA-4-5-3.html))
- 4.5.4 Unary Adding Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-4.html))

5.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.3 Binary Adding Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-3.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-3.html))
- 4.5.4 Unary Adding Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-4.html))

6 Operators: /=

6.1 Operator

The operator **/=** compares two values on inequality. It is predefined for all non limited types. The operator will also be defined if a suitable operator = is available.

Note that in Ada the representation for this operator was chosen for resembling the mathematical symbol ≠, in the same way than <= resembles ≤, or >= resembles ≥.

6.2 See also

6.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters

6.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/95lrm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-2.html))

6.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/05rm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-2.html))

7 Operators: /

7.1 Operator

7.1.1 Standard operations

7.1.1.1 Arithmetic division

The "/" operator is defined as arithmetic division for all numeric types.

```
function "/" (Left, Right : T) return T;
```

7.1.1.1.1 Usage

```
A : constant Float   := 5.0 / 2.0;  -- A is now 2.5
B : constant Integer := 5 / 2;      -- B is also 2
```

7.2 See also

7.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters
- Ada Programming/Operators
- Ada Programming/Mathematical calculations

7.2.2 Ada Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.5 Multiplying Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-5.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-5.html))

8 Operators: =

8.1 Operator

The operator **=** compares two values on equality. It is predefined for all non limited types.

8.2 See also

8.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters

8.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/95lrm/html/RM-4-5-2.html) (Annotated

(http://www.adaic.com/standards/95aarm/html/AA-4-5-2.html))

8.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/05rm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-2.html))

9 Operators: abs

This keyword is used for the operator that gets the absolute value of an integer number.

```
y := abs (x);
```

9.1 See also

9.1.1 Wikibook

- Ada Programming
- Ada Programming/Keywords

9.1.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.6 Highest Precedence Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-6.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-6.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

9.1.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.6 Highest Precedence Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-6.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-6.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))

9.1.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)
- 5.5.3 Parenthetical Expressions (http://www.adaic.org/docs/95style/html/sec_5/5-5-3.html)

10 Operators: and

10.1 Logical operator

10.1.1 Boolean operator

```
X : Boolean := A > 10 and A < 20;
```

10.1.2 Boolean shortcut operator

Shortcut operators are used to make the evaluation of parts of boolean expressions conditional: *and then*, *or else*. This should never be done to speed up the evaluation (with modern optimizing compilers, it will possibly not have that effect). The correct use is to prevent the evaluation of expressions known to raise an exception.

```
if Dog /= null and then G (Dog) then
   Walk (Dog);
end if;
```

In the example above, `G (Dog)` is only called when the pointer `Dog` is not `null`, i.e. it actually points to something.

Actually *and then* and *or else* are not operators in the sense of the reference manual, they are called 'Short-circuit Control Forms'. The difference is that (true) operators can be redefined (i.e. overloaded), whereas these cannot. They are however defined for any boolean type.

Since Ada allows parallel evaluation of the arguments for an expression, shortcut operators are not the standard way of evaluating boolean expressions. In case the final result of the evaluation is guaranteed to be the same, the compiler is allowed to use a shortcut evaluation.

10.1.3 Boolean operator on arrays

The and operator is applied to each pair of boolean elements from the left and right arrays. The result has the same bounds than the left operand.

```
type Day_Of_Month is range 1 .. 31;
type Month_Array is array (Day_Of_Month) of Boolean;

X : Month_Array := Function_1;
Y : Month_Array := Function_2;
Z : Month_Array := X and Y;
```

10.1.4 Bitwise operator

The operator **and** could be used with modular types to perform bitwise operations.

10.2 Adding interfaces to tagged types

This language feature will be made available in the forthcoming Ada 2005 standard.

```
type Programmer is new Person
                   and Printable
with
   record
     Skilled_In : Language_List;
   end record;
```

10.3 See also

10.3.1 Wikibook

- Ada Programming
- Ada Programming/Keywords
- Ada Programming/Operators
- Ada Programming/Keywords/interface

10.3.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

10.3.2.1 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- 3.9.4 Interface Types (http://www.adaic.com/standards/05rm/html/RM-3-9-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-3-9-4.html))
- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))

10.3.3 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)

11 Operators: >=

11.1 Operator

The operator **>=** compares two values on greater than or equal to. It is predefined for all discrete types.

11.2 See also

11.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters

11.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/95lrm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-2.html))

11.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/05rm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-2.html))

12 Operators: >

12.1 Operator

The operator **>** compares two values on being greater. It is predefined for all discrete types.

12.2 See also

12.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters

12.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/95lrm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-2.html))

12.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/05rm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-2.html))

13 Operators: in

This keyword is used in:

- The `in` and `in out` mode of subprograms parameters.
- The membership test.

13.1 See also

13.1.1 Wikibook

- Ada Programming
- Ada Programming/Keywords

13.1.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 6.1 Subprogram Declarations (http://www.adaic.com/standards/95lrm/html/RM-6-1.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-6-1.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

13.1.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 6.1 Subprogram Declarations (http://www.adaic.com/standards/05rm/html/RM-6-1.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-6-1.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))

13.1.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)

14 Operators: <=

14.1 Operator

The operator **<=** compares two values on less than or equal to. It is predefined for all discreed types.

14.2 See also

14.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters

14.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/95lrm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-2.html))

14.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/05rm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-2.html))

15 Operators: <

15.1 Operator

The operator **<** compares two values on less than. It is predefined for all discrete types.

15.2 See also

15.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters

15.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/95lrm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-2.html))

15.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.2 Relational Operators and Membership Tests (http://www.adaic.com/standards/05rm/html/RM-4-5-2.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-2.html))

16 Operators: mod

This keyword is used in the mod operator and in the declaration of modular types.

16.1 See also

16.1.1 Wikibook

- Ada Programming
- Ada Programming/Keywords

16.1.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

16.1.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))

16.1.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)

17 Operators: not

This keyword is used in:

- Logical negation operator
- Negative membership test: `not in`

17.1 See also

17.1.1 Wikibook

- Ada Programming
- Ada Programming/Keywords

17.1.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.6 Highest Precedence Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-6.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-6.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

17.1.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated

(http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.6 Highest Precedence Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-6.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-6.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))

17.1.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)

18 Operators: or

18.1 Logical operator

18.1.1 Boolean operator

```
X : Boolean := A < 10 or A > 20;
```

18.1.2 Boolean shortcut operator

In the below example the function `G` is only called when `F(X)` returns the value *False*.

```
if F(X) or else G(Y) then

   Walk_The_Dog;

end if;
```

This shortcut operator is sometimes used to speed up the evaluation of boolean expressions, but the Ada Style Guide recommends to compare the performance of both forms before switching one to the other. In general, it is good idea to use **or else** in sake of performance only when the second expression involves a function call.

The **or else** form is also used when the second expression is known to raise an exception unless the first expression is *False*.

Unlike C/C++, Ada short-cut operators are not the standard way to evaluate boolean expressions. This is because Ada is designed to do by default what is generally safer, but lets the programmer request a different behaviour.

18.1.3 Boolean operator on arrays

The or operator is applied to each pair of boolean elements from the left and right arrays. The result has the same bounds than the left operand.

```
type Day_Of_Month is range 1 .. 31;
type Month_Array is array (Day_Of_Month) of Boolean;

X : Month_Array := Function_1;
Y : Month_Array := Function_2;
Z : Month_Array := X or Y;
```

18.1.4 Bitwise operator

The operator **or** could be used with modular types to perform bitwise operations.

18.2 Select statement

18.2.1 alternative

See Ada Programming/Tasking#Selective_waiting.

18.2.2 delay

See Ada Programming/Tasking#Timeout.

18.3 See also

18.3.1 Wikibook

- Ada Programming
- Ada Programming/Keywords
- Ada Programming/Operators

18.3.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.1 Logical Operators and Short-circuit Control Forms (http://www.adaic.com/standards/95lrm/html/RM-4-5-1.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-1.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

18.3.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.1 Logical Operators and Short-circuit Control Forms (http://www.adaic.com/standards/95lrm/html/RM-4-5-1.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-1.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

18.3.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)
- 5.5.5 Short Circuit Forms of the Logical Operators (http://www.adaic.org/docs/95style/html/sec_5/5-5-5.html)
- 10.5.2 Short-Circuit Operators (http://www.adaic.org/docs/95style/html/sec_10/10-5-2.html)
- 10.6.3 Bit Operations on Modular Types

(http://www.adaic.org/docs/95style/html/sec_10/10-6-3.html)

19 Operators: +

19.1 Operator

19.1.1 Standard Operations

19.1.1.1 Arithmetic Addition

The "+" operator is defined as arithmetic addition for all numeric types.

```
function "+" (Left, Right : T) return T;
```

19.1.1.2 Plus sign

The "+" operator is defined as arithmetic plus sign for all numeric types.

```
function "+" (Left : T) return T;
```

19.1.1.2.1 Usage

```
A : constant Float   := +5.0;  -- A is now 5.0
B : constant Integer := +5;    -- B is also 5
```

19.1.2 Common Non-Standard Operations

19.1.2.1 Type Convertion

The operator plus sign is often used to create a type conversion operator:

```
function "+" (Left : T1) return T2;
```

19.2 See also

19.2.1 Wikibook

- Ada Programming
- Ada Programming/Delimiters
- Ada Programming/Operators
- Ada Programming/Mathematical calculations

19.2.2 Ada 95 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.3 Binary Adding Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-3.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-3.html))

- 4.5.4 Unary Adding Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-4.html))

19.2.3 Ada 2005 Reference Manual

- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.3 Binary Adding Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-3.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-3.html))
- 4.5.4 Unary Adding Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-4.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-4.html))

20 Operators: rem

20.1 Operator rem

The **rem** keyword is used as the remainder operator, that is, the remainder of the signed integer division. The following formula applies:

```
A = (A / B) * B + (A rem B)
```

20.2 See also

20.2.1 Wikibook

- Ada Programming
- Ada Programming/Keywords

20.2.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))
- 4.5.5 Multiplying Operators (http://www.adaic.com/standards/95lrm/html/RM-4-5-5.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-5.html))

20.2.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))
- 4.5.5 Multiplying Operators (http://www.adaic.com/standards/05rm/html/RM-4-5-5.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-5.html))

20.2.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)

21 Operators: xor

21.1 Logical operator

21.1.1 Boolean operator

```
X : Boolean := A = 10 xor B = 10;
```

21.1.2 Boolean operator on arrays

The xor operation is applied to each boolean inside the array.

```
type Day_Of_Month is range 1 .. 31;
type Month_Array is array (Day_Of_Month) of Boolean;

X : Month_Array := Function_1;
Y : Month_Array := Function_2;
Z : Month_Array := X xor Y;
```

21.1.3 Bitwise operator

The operator **xor** could be used with modular types and also with boolean arrays to perform bitwise operations.

21.2 See also

21.2.1 Wikibook

- Ada Programming
- Ada Programming/Keywords
- Ada Programming/Operators

21.2.2 Ada 95 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/95lrm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/95lrm/html/RM-4-4.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-4.html))
- 4.5.1 Logical Operators and Short-circuit Control Forms (http://www.adaic.com/standards/95lrm/html/RM-4-5-1.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-4-5-1.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/95lrm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/95aarm/html/AA-P.html))

21.2.3 Ada 2005 Reference Manual

- 2.9 Reserved Words (http://www.adaic.com/standards/05rm/html/RM-2-9.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-2-9.html))
- 4.4 Expressions (http://www.adaic.com/standards/05rm/html/RM-4-4.html) (Annotated

(http://www.adaic.com/standards/05aarm/html/AA-4-4.html))
- 4.5.1 Logical Operators and Short-circuit Control Forms (http://www.adaic.com/standards/05rm/html/RM-4-5-1.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-4-5-1.html))
- Annex P (informative) Syntax Summary (http://www.adaic.com/standards/05rm/html/RM-P.html) (Annotated (http://www.adaic.com/standards/05aarm/html/AA-P.html))

21.2.4 Ada Quality and Style Guide

- 3.1.3 Capitalization (http://www.adaic.org/docs/95style/html/sec_3/3-1-3.html)
- 10.6.3 Bit Operations on Modular Types (http://www.adaic.org/docs/95style/html/sec_10/10-6-3.html)

Von „http://en.wikibooks.org/wiki/Ada_Programming/All_Operators“

Kategorien: Books with print version | Ada Programming/Ada 2005 feature | Ada Programming | Ada programming language | Programming

---

- Diese Seite wurde zuletzt am 30. Juni 2006 um 23:35 Uhr geändert.
- All text is available under the terms of the GNU Free Documentation License (see **Copyrights** for details).
  Wikibooks® is a registered trademark of the Wikimedia Foundation, Inc.
- Datenschutz
- Über Wikibooks
- Impressum