

Numbers

<code>parse-integer</code>	Parses an integer from a string
<code>floor</code> , <code>ceiling</code>	Truncates toward negative/positive infinity
<code>round</code>	Rounds to the nearest integer
<code>mod</code> , <code>rem</code>	Modulus/remainder of a truncating division
<code>1+</code> , <code>1-</code>	Returns a new value (inc/dec)rement by a number
<code>incf</code> , <code>decf</code>	ⓘ Macros to (inc/dec)rement in place
<code>=</code> , <code>/=</code>	Mathematical value equality (types ignored)
<code>></code> , <code>>=</code> , <code><</code> , <code><=</code>	Order comparison

Characters

Strings

<code>format</code>	Returns a formatted output string when the <i>destination</i> is <code>nil</code>
---------------------	---

Sequences

<code>elt</code>	Access elements with an integer index
<code>length</code>	
<code>count</code>	Count the appearances of an element
<code>find</code>	Finds an item or returns <code>nil</code>
<code>position</code>	The first index of an element
<code>search</code>	Returns the start position of a sequence in another sequence
<code>remove</code>	Removes all the occurrences of an item
<code>substitute</code>	Replaces all occurrences of an item

Lists

<code>mapcar</code>	Successively applies a function to each element of a list
<code>maplist</code>	Like <code>mapcar</code> , but its function receives a cons cell instead of its element (therefore, the function has access to the rest of the list)
<code>mapcan</code>	Like <code>mapcar</code> , but the results are combined into a list
<code>mapcon</code>	Like <code>mapcan</code> , but applied to <code>maplist</code>

Vectors

Hash Tables

Things to remember

Object Oriented	All values in a Common Lisp program are instances of some class . Furthermore, all classes are organized into a single hierarchy rooted at the class <code>T</code> .
Starting a project	SBCL and Quicklisp installation and a project template

Equality

<code>eq</code>	Object identity (don't use with numbers or chars)
<code>eql</code>	👍 Considers equivalent two objects of the same class with the same numeric or char value
<code>equal</code>	Lists with the same structure and contents, strings with the same characters
<code>equalp</code>	Ignores differences in case or in numerical type (<code>1 == 1 .0</code>)

Functions

<code>defun</code>	
<code>lambda</code>	Returns an anonymous function
<code>return -from</code>	Immediately returns a value from a function (or a block)
<code>funcall</code>	Invokes a function from a function object
<code>apply</code>	Works like <code>funcall</code> , but receives the arguments as a list
<code>&optional</code> , <code>&key</code> , <code>&rest</code> , <code>&allow-other-keys</code>	Different ways to capture function arguments

Standard Control Constructs

<code>if</code>	The <i>else</i> form is optional
<code>when</code>	Like <code>if</code> , but returns <code>nil</code> if the condition is <i>falsey</i> and evaluates multiple body forms
<code>unless</code>	Like <code>when</code> , but executes its body only when the condition is <i>falsey</i>



By Andre Boechat
(boechat107)

cheatography.com/boechat107/
boechat107.github.io/

Published 30th January, 2018.
Last updated 10th June, 2024.
Page 1 of 2.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>

IO

`format` Produces formatted output to *stdout* when the *destination* is `t`

`format language`

`a, s,` Interpolate argument in human readable format; interpolate as *Lisp* readable

`%`, `&` Newline; newline if not at the beginning of a line

Object-Oriented

`defgeneric` Defines an abstract operation (*polymorphism*)

`defmethod` Defines an implementation of a generic function

`call-n ext -method` Similar to an invocation to a super-class method

`defclass` New named class; some slot options are: `:reader`, `:writer`, `:accessor`, `:initarg`, `:initform`, `:documentation`, `:allocation`

`slot-value` Returns the value of slot in the object (`setf`-able)

`with-slots, with-accessors` Binds a slot/accessor to a symbol that can be used in its body



By Andre Boechat
(boechat107)

Published 30th January, 2018.
Last updated 10th June, 2024.
Page 2 of 2.

Sponsored by **Readable.com**
Measure your website readability!
<https://readable.com>

cheatography.com/boechat107/
boechat107.github.io/