4.9 Pairs and Lists

+[Pairs and Lists](https://docs.racket-lang.org/guide/pairs.html) in [**The Racket Guide**](https://docs.racket-lang.org/guide/index.html)introduces pairs and lists.

A *pair* combines exactly two values. The first value is accessed with the [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) procedure, and the second value is accessed with the [cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) procedure. Pairs are not mutable (but see [Mutable Pairs and Lists](https://docs.racket-lang.org/reference/mpairs.html)).

A *list* is recursively defined: it is either the constant [null](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null%29%29), or it is a pair whose second value is a list.

A list can be used as a single-valued sequence (see [Sequences](https://docs.racket-lang.org/reference/sequences.html)). The elements of the list serve as elements of the sequence. See also [in-list](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._in-list%29%29).

Cyclic data structures can be created using only immutable pairs via [read](https://docs.racket-lang.org/reference/Reading.html#%28def._%28%28quote._~23~25kernel%29._read%29%29) or [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29). If starting with a pair and using some number of [cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29)s returns to the starting pair, then the pair is not a list.

See [Reading Pairs and Lists](https://docs.racket-lang.org/reference/reader.html#%28part._parse-pair%29) for information on [read](https://docs.racket-lang.org/reference/Reading.html#%28def._%28%28quote._~23~25kernel%29._read%29%29)ing pairs and lists and [Printing Pairs and Lists](https://docs.racket-lang.org/reference/printing.html#%28part._print-pairs%29) for information on [print](https://docs.racket-lang.org/reference/Writing.html#%28def._%28%28quote._~23~25kernel%29._print%29%29)ing pairs and lists.

4.9.1 Pair Constructors and Selectors

|  |
| --- |
| procedure  (**[pair?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)** *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns #t if *v* is a pair, #f otherwise.

Examples:

|  |
| --- |
| > ([pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) 1) |
| #f |
| > ([pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2)) |
| #t |
| > ([pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2)) |
| #t |
| > ([pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) '(1 2)) |
| #t |
| > ([pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) '()) |
| #f |
| procedure  (**[null?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._null~3f%29%29)** *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns #t if *v* is the empty list, #f otherwise.

Examples:

|  |
| --- |
| > ([null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) 1) |
| #f |
| > ([null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) '(1 2)) |
| #f |
| > ([null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) '()) |
| #t |
| > ([null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1))) |
| #t |
| procedure  (**[cons](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cons%29%29)** *a* *d*) → [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) |
| *a* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *d* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns a newly allocated pair whose first element is *a* and second element is *d*.

Examples:

|  |
| --- |
| > ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2) |
| '(1 . 2) |
| > ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 '()) |
| '(1) |
| procedure  (**[car](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._car%29%29)** *p*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *p* : [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) |

Returns the first element of the pair *p*.

Examples:

|  |
| --- |
| > ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) '(1 2)) |
| 1 |
| > ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 2 3)) |
| 2 |
| procedure  (**[cdr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdr%29%29)** *p*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *p* : [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) |

Returns the second element of the pair *p*.

Examples:

|  |
| --- |
| > ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) '(1 2)) |
| '(2) |
| > ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) '(1)) |
| '() |
| value  **[null](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._null%29%29)** : [null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) |

The empty list.

Examples:

|  |
| --- |
| > [null](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null%29%29) |
| '() |
| > '() |
| '() |
| > ([eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29) '() [null](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null%29%29)) |
| #t |
| procedure  (**[list?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._list~3f%29%29)** *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns #t if *v* is a list: either the empty list, or a pair whose second element is a list. This procedure effectively takes constant time due to internal caching (so that any necessary traversals of pairs can in principle count as an extra cost of allocating the pairs).

Examples:

|  |
| --- |
| > ([list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) '(1 2)) |
| #t |
| > ([list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 2 '()))) |
| #t |
| > ([list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2)) |
| #f |
| procedure  (**[list](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._list%29%29)** *v* ...) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns a newly allocated list containing the *v*s as its elements.

Examples:

|  |
| --- |
| > ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4) |
| '(1 2 3 4) |
| > ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4)) |
| '((1 2) (3 4)) |
| procedure  (**[list\*](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._list%2A%29%29)** *v* ... *tail*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *tail* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Like [list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29), but the last argument is used as the tail of the result, instead of the final element. The result is a list only if the last argument is a list.

Examples:

|  |
| --- |
| > ([list\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%2A%29%29) 1 2) |
| '(1 . 2) |
| > ([list\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%2A%29%29) 1 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4)) |
| '(1 2 3 4) |
| procedure  ([**build-list**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._build-list%29%29) *n* *proc*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *n* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |
| *proc* : ([exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29)) |

Creates a list of *n* elements by applying *proc* to the integers from 0 to ([sub1](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._sub1%29%29) *n*) in order. If*lst* is the resulting list, then ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) *lst* *i*) is the value produced by (*proc* *i*).

Examples:

|  |
| --- |
| > ([build-list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._build-list%29%29) 10 [values](https://docs.racket-lang.org/reference/values.html#%28def._%28%28quote._~23~25kernel%29._values%29%29)) |
| '(0 1 2 3 4 5 6 7 8 9) |
| > ([build-list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._build-list%29%29) 5 ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) ([\*](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2A%29%29) x x))) |
| '(0 1 4 9 16) |

4.9.2 List Operations

|  |
| --- |
| procedure  (**[length](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._length%29%29)** *lst*) → [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the number of elements in *lst*.

Examples:

|  |
| --- |
| > ([length](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._length%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| 4 |
| > ([length](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._length%29%29) '()) |
| 0 |
| procedure  (**[list-ref](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._list-ref%29%29)** *lst* *pos*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns the element of *lst* at position *pos*, where the list’s first element is position 0. If the list has *pos* or fewer elements, then the [exn:fail:contract](https://docs.racket-lang.org/reference/exns.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._exn~3afail~3acontract%29%29) exception is raised.

The *lst* argument need not actually be a list; *lst* must merely start with a chain of at least ([add1](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._add1%29%29) *pos*) pairs.

Examples:

|  |
| --- |
| > ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 'a 'b 'c) 0) |
| 'a |
| > ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 'a 'b 'c) 1) |
| 'b |
| > ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 'a 'b 'c) 2) |
| 'c |
| > ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2) 0) |
| 1 |
| > ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2) 1) |
| *list-ref: index reaches a non-pair* |
| *index: 1* |
| *in: '(1 . 2)* |
| procedure  (**[list-tail](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._list-tail%29%29)** *lst* *pos*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns the list after the first *pos* elements of *lst*. If the list has fewer than *pos* elements, then the [exn:fail:contract](https://docs.racket-lang.org/reference/exns.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._exn~3afail~3acontract%29%29) exception is raised.

The *lst* argument need not actually be a list; *lst* must merely start with a chain of at least *pos* pairs.

Examples:

|  |
| --- |
| > ([list-tail](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-tail%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4) 2) |
| '(3 4) |
| > ([list-tail](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-tail%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2) 1) |
| 2 |
| > ([list-tail](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-tail%29%29) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 2) 2) |
| *list-tail: index reaches a non-pair* |
| *index: 2* |
| *in: '(1 . 2)* |
| > ([list-tail](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-tail%29%29) 'not-a-pair 0) |
| 'not-a-pair |
| procedure  (**[append](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._append%29%29)** *lst* ...) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| ([**append**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29) *lst* ... *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

When given all list arguments, the result is a list that contains all of the elements of the given lists in order. The last argument is used directly in the tail of the result.

The last argument need not be a list, in which case the result is an “improper list.”

Examples:

|  |
| --- |
| > ([append](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4)) |
| '(1 2 3 4) |
| > ([append](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 5 6) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 7 8)) |
| '(1 2 3 4 5 6 7 8) |
| procedure  ([**reverse**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._reverse%29%29) *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns a list that has the same elements as *lst*, but in reverse order.

Example:

|  |
| --- |
| > ([reverse](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._reverse%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| '(4 3 2 1) |

4.9.3 List Iteration

|  |
| --- |
| procedure  (**[map](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29)** *proc* *lst* ...+) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Applies *proc* to the elements of the *lst*s from the first elements to the last. The *proc*argument must accept the same number of arguments as the number of supplied *lst*s, and all *lst*s must have the same number of elements. The result is a list containing each result of *proc* in order.

Examples:

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | > ([map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (number) | | ([+](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2B%29%29) 1 number)) | | '(1 2 3 4)) | |
| '(2 3 4 5) |
| |  | | --- | | > ([map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (number1 number2) | | ([+](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2B%29%29) number1 number2)) | | '(1 2 3 4) | | '(10 100 1000 10000)) | |
| '(11 102 1003 10004) |
| procedure  (**[andmap](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29)** *proc* *lst* ...+) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Similar to [map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) in the sense that *proc* is applied to each element of *lst*, but

The [andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) function is actually closer to [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) than [map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29), since [andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29)doesn’t produce a list. Still, ([andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29)f ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) x y z)) is equivalent to ([and](https://docs.racket-lang.org/reference/if.html#%28form._%28%28lib._racket%2Fprivate%2Fletstx-scheme..rkt%29._and%29%29) (f x) (f y) (f z)) in the same way that ([map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) f ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) x yz)) is equivalent to ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) (f x) (fy) (f z)).

* the result is #f if any application of *proc* produces #f, in which case *proc* is not applied to later elements of the *lst*s; and
* the result is that of *proc* applied to the last elements of the *lst*s; more specifically, the application of *proc* to the last elements in the *lst*s is in tail position with respect to the [andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) call.

If the *lst*s are empty, then #t is returned.

Examples:

|  |
| --- |
| > ([andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) [positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) '(1 2 3)) |
| #t |
| > ([andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) [positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) '(1 2 a)) |
| *positive?: contract violation* |
| *expected: real?* |
| *given: 'a* |
| > ([andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) [positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) '(1 -2 a)) |
| #f |
| > ([andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) [+](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2B%29%29) '(1 2 3) '(4 5 6)) |
| 9 |
| procedure  (**[ormap](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._ormap%29%29)** *proc* *lst* ...+) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Similar to [map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) in the sense that *proc* is applied to each element of *lst*, but

To continue the [andmap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._andmap%29%29) note above,([ormap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._ormap%29%29) f ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) x y z)) is equivalent to ([or](https://docs.racket-lang.org/reference/if.html#%28form._%28%28lib._racket%2Fprivate%2Fletstx-scheme..rkt%29._or%29%29) (f x) (f y) (fz)).

* the result is #f if every application of *proc* produces #f; and
* the result is that of the first application of *proc* producing a value other than #f, in which case *proc* is not applied to later elements of the *lst*s; the application of *proc* to the last elements of the *lst*s is in tail position with respect to the [ormap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._ormap%29%29) call.

If the *lst*s are empty, then #f is returned.

Examples:

|  |
| --- |
| > ([ormap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._ormap%29%29) [eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29) '(a b c) '(a b c)) |
| #t |
| > ([ormap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._ormap%29%29) [positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) '(1 2 a)) |
| #t |
| > ([ormap](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._ormap%29%29) [+](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2B%29%29) '(1 2 3) '(4 5 6)) |
| 5 |
| procedure  (**[for-each](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._for-each%29%29)** *proc* *lst* ...+) → [void?](https://docs.racket-lang.org/reference/void.html#%28def._%28%28quote._~23~25kernel%29._void~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Similar to [map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29), but *proc* is called only for its effect, and its result (which can be any number of values) is ignored.

Example:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | > ([for-each](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._for-each%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (arg) | | ([printf](https://docs.racket-lang.org/reference/Writing.html#%28def._%28%28quote._~23~25kernel%29._printf%29%29) "Got ~a\n" arg) | | 23) | | '(1 2 3 4)) | |
| |  | | --- | | Got 1 | | Got 2 | | Got 3 | | Got 4 | |
| procedure  ([**foldl**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) *proc* *init* *lst* ...+) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *init* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29), [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) applies a procedure to the elements of one or more lists. Whereas [map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29)combines the return values into a list, [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) combines the return values in an arbitrary way that is determined by *proc*.

If [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) is called with *n* lists, then *proc* must take *n+*1 arguments. The extra argument is the combined return values so far. The *proc* is initially invoked with the first item of each list, and the final argument is *init*. In subsequent invocations of *proc*, the last argument is the return value from the previous invocation of *proc*. The input *lst*s are traversed from left to right, and the result of the whole [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) application is the result of the last application of *proc*. If the *lst*s are empty, the result is *init*.

Unlike [foldr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldr%29%29), [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) processes the *lst*s in constant space (plus the space for each call to *proc*).

Examples:

|  |
| --- |
| > ([foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) [cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) '() '(1 2 3 4)) |
| '(4 3 2 1) |
| > ([foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) [+](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2B%29%29) 0 '(1 2 3 4)) |
| 10 |
| |  | | --- | | > ([foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (a b result) | | ([\*](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._%2A%29%29) result ([-](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._-%29%29) a b))) | | 1 | | '(1 2 3) | | '(4 5 6)) | |
| -27 |
| procedure  ([**foldr**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldr%29%29) *proc* *init* *lst* ...+) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *init* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29), but the lists are traversed from right to left. Unlike [foldl](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldl%29%29), [foldr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldr%29%29) processes the *lst*s in space proportional to the length of *lst*s (plus the space for each call to *proc*).

Examples:

|  |
| --- |
| > ([foldr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldr%29%29) [cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) '() '(1 2 3 4)) |
| '(1 2 3 4) |
| > ([foldr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._foldr%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (v l) ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) ([add1](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._add1%29%29) v) l)) '() '(1 2 3 4)) |
| '(2 3 4 5) |

4.9.4 List Filtering

|  |
| --- |
| procedure  ([**filter**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29) *pred* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns a list with the elements of *lst* for which *pred* produces a true value. The *pred*procedure is applied to each element from first to last.

Example:

|  |
| --- |
| > ([filter](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29) [positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) '(1 -2 3 4 -5)) |
| '(1 3 4) |
| procedure  ([**remove**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) *v* *lst* [*proc*]) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Returns a list that is like *lst*, omitting the first element of *lst* that is equal to *v* using the comparison procedure *proc* (which must accept two arguments).

Examples:

|  |
| --- |
| > ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 2 4)) |
| '(1 3 2 4) |
| > ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 2 4) [=](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3d%29%29)) |
| '(1 3 2 4) |
| > ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) '(2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) '(1) '(2) '(3))) |
| '((1) (3)) |
| > ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) "2" ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) "1" "2" "3")) |
| '("1" "3") |
| > ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) #\c ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) #\a #\b #\c)) |
| '(#\a #\b) |
| procedure  ([**remq**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%29%29) *v* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) *v* *lst* [eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29)).

Examples:

|  |
| --- |
| > ([remq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4 5)) |
| '(1 3 4 5) |
| > ([remq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%29%29) '(2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) '(1) '(2) '(3))) |
| '((1) (2) (3)) |
| > ([remq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%29%29) "2" ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) "1" "2" "3")) |
| '("1" "3") |
| > ([remq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%29%29) #\c ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) #\a #\b #\c)) |
| '(#\a #\b) |
| procedure  ([**remv**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%29%29) *v* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29) *v* *lst* [eqv?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eqv~3f%29%29)).

Examples:

|  |
| --- |
| > ([remv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4 5)) |
| '(1 3 4 5) |
| > ([remv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%29%29) '(2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) '(1) '(2) '(3))) |
| '((1) (2) (3)) |
| > ([remv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%29%29) "2" ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) "1" "2" "3")) |
| '("1" "3") |
| > ([remv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%29%29) #\c ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) #\a #\b #\c)) |
| '(#\a #\b) |
| procedure  ([**remove\***](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%2A%29%29) *v-lst* *lst* [*proc*]) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v-lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Like [remove](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%29%29), but removes from *lst* every instance of every element of *v-lst*.

Example:

|  |
| --- |
| > ([remove\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%2A%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 2 4 5 2)) |
| '(3 4 5) |
| procedure  ([**remq\***](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%2A%29%29) *v-lst* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v-lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([remove\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%2A%29%29) *v-lst* *lst* [eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29)).

Example:

|  |
| --- |
| > ([remq\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remq%2A%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 2 4 5 2)) |
| '(3 4 5) |
| procedure  ([**remv\***](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%2A%29%29) *v-lst* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v-lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([remove\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remove%2A%29%29) *v-lst* *lst* [eqv?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eqv~3f%29%29)).

Example:

|  |
| --- |
| > ([remv\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._remv%2A%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 2 4 5 2)) |
| '(3 4 5) |
| procedure   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | ([**sort**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._sort%29%29) |  | *lst* |  |  |  |  | |  |  | *less-than?* |  |  |  |  | |  | [ | #:key *extract-key* |  |  |  |  | |  |  | #:cache-keys? *cache-keys?*]) |  | → |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *less-than?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |
| *extract-key* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) x) |
| *cache-keys?* : [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) = #f |

Returns a list sorted according to the *less-than?* procedure, which takes two elements of *lst* and returns a true value if the first is less (i.e., should be sorted earlier) than the second.

The sort is stable; if two elements of *lst* are “equal” (i.e., *less-than?* does not return a true value when given the pair in either order), then the elements preserve their relative order from *lst* in the output list. To preserve this guarantee, use [sort](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._sort%29%29) with a strict comparison functions (e.g., [<](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3c%29%29) or [string<?](https://docs.racket-lang.org/reference/strings.html#%28def._%28%28quote._~23~25kernel%29._string~3c~3f%29%29); not [<=](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3c~3d%29%29) or [string<=?](https://docs.racket-lang.org/reference/strings.html#%28def._%28%28quote._~23~25kernel%29._string~3c~3d~3f%29%29)).

Because of the peculiar fact that the IEEE-754 number system specifies that +nan.0 is neither greater nor less than nor equal to any other number, sorting lists containing this value may produce a surprising result.

The #:key argument *extract-key* is used to extract a key value for comparison from each list element. That is, the full comparison procedure is essentially

|  |
| --- |
| ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x y) |
| (*less-than?* (*extract-key* x) (*extract-key* y))) |

By default, *extract-key* is applied to two list elements for every comparison, but if *cache-keys?* is true, then the *extract-key* function is used exactly once for each list item. Supply a true value for *cache-keys?* when *extract-key* is an expensive operation; for example, if [file-or-directory-modify-seconds](https://docs.racket-lang.org/reference/Filesystem.html#%28def._%28%28quote._~23~25kernel%29._file-or-directory-modify-seconds%29%29) is used to extract a timestamp for every file in a list, then *cache-keys?* should be #t to minimize file-system calls, but if *extract-key* is [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29), then *cache-keys?* should be #f. As another example, providing *extract-key* as([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) ([random](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._random%29%29))) and #t for *cache-keys?* effectively shuffles the list.

Examples:

|  |
| --- |
| > ([sort](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._sort%29%29) '(1 3 4 2) [<](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3c%29%29)) |
| '(1 2 3 4) |
| > ([sort](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._sort%29%29) '("aardvark" "dingo" "cow" "bear") [string<?](https://docs.racket-lang.org/reference/strings.html#%28def._%28%28quote._~23~25kernel%29._string~3c~3f%29%29)) |
| '("aardvark" "bear" "cow" "dingo") |
| |  | | --- | | > ([sort](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._sort%29%29) '(("aardvark") ("dingo") ("cow") ("bear")) | | #:key [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) [string<?](https://docs.racket-lang.org/reference/strings.html#%28def._%28%28quote._~23~25kernel%29._string~3c~3f%29%29)) | |
| '(("aardvark") ("bear") ("cow") ("dingo")) |

4.9.5 List Searching

|  |
| --- |
| procedure  ([**member**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29) *v* *lst* [*is-equal?*]) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) #f) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *is-equal?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Locates the first element of *lst* that is [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) to *v*. If such an element exists, the tail of *lst* starting with that element is returned. Otherwise, the result is #f.

Examples:

|  |
| --- |
| > ([member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| '(2 3 4) |
| > ([member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29) 9 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| #f |
| > ([member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29) #'x ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) #'x #'y) [free-identifier=?](https://docs.racket-lang.org/reference/stxcmp.html#%28def._%28%28quote._~23~25kernel%29._free-identifier~3d~3f%29%29)) |
| '(#<syntax:504:0 x> #<syntax:504:0 y>) |
| > ([member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29) #'a ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) #'x #'y) [free-identifier=?](https://docs.racket-lang.org/reference/stxcmp.html#%28def._%28%28quote._~23~25kernel%29._free-identifier~3d~3f%29%29)) |
| #f |
| procedure  ([**memv**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._memv%29%29) *v* *lst*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) #f) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29), but finds an element using [eqv?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eqv~3f%29%29).

Examples:

|  |
| --- |
| > ([memv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._memv%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| '(2 3 4) |
| > ([memv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._memv%29%29) 9 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| #f |
| procedure  ([**memq**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._memq%29%29) *v* *lst*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) #f) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29), but finds an element using [eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29).

Examples:

|  |
| --- |
| > ([memq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._memq%29%29) 2 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| '(2 3 4) |
| > ([memq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._memq%29%29) 9 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2 3 4)) |
| #f |
| procedure  ([**memf**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._memf%29%29) *proc* *lst*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) #f) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29), but finds an element using the predicate *proc*; an element is found when *proc* applied to the element returns a true value.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | > ([memf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._memf%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (arg) | | ([>](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3e%29%29) arg 9)) | | '(7 8 9 10 11)) | |
| '(10 11) |
| procedure  ([**findf**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._findf%29%29) *proc* *lst*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [memf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._memf%29%29), but returns the element or #f instead of a tail of *lst* or #f.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | > ([findf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._findf%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (arg) | | ([>](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3e%29%29) arg 9)) | | '(7 8 9 10 11)) | |
| 10 |
| procedure  ([**assoc**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29) *v* *lst* [*is-equal?*]) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) #f) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |
| *is-equal?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Locates the first element of *lst* whose [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) is equal to *v* according to *is-equal?*. If such an element exists, the pair (i.e., an element of *lst*) is returned. Otherwise, the result is #f.

Examples:

|  |
| --- |
| > ([assoc](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29) 3 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 5 6))) |
| '(3 4) |
| > ([assoc](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29) 9 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 5 6))) |
| #f |
| |  | | --- | | > ([assoc](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29) 3.5 | | ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 5 6)) | | ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (a b) ([<](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3c%29%29) ([abs](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._abs%29%29) ([-](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._-%29%29) a b)) 1))) | |
| '(3 4) |
| procedure  ([**assv**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assv%29%29) *v* *lst*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) #f) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Like [assoc](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29), but finds an element using [eqv?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eqv~3f%29%29).

Example:

|  |
| --- |
| > ([assv](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assv%29%29) 3 ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 5 6))) |
| '(3 4) |
| procedure  ([**assq**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assq%29%29) *v* *lst*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) #f) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Like [assoc](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29), but finds an element using [eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29).

Example:

|  |
| --- |
| > ([assq](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assq%29%29) 'c ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 'a 'b) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 'c 'd) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 'e 'f))) |
| '(c d) |
| procedure  ([**assf**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assf%29%29) *proc* *lst*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) #f) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [assoc](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assoc%29%29), but finds an element using the predicate *proc*; an element is found when *proc*applied to the [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) of an *lst* element returns a true value.

Example:

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | > ([assf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._assf%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (arg) | | ([>](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3e%29%29) arg 2)) | | ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 1 2) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 3 4) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) 5 6))) | |
| '(3 4) |

4.9.6 Pair Accessor Shorthands

|  |
| --- |
| procedure  (**[caar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)).

Example:

|  |
| --- |
| > ([caar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caar%29%29) '((1 2) 3 4)) |
| 1 |
| procedure  (**[cadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)).

Example:

|  |
| --- |
| > ([cadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cadr%29%29) '((1 2) 3 4)) |
| 3 |
| procedure  (**[cdar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)).

Example:

|  |
| --- |
| > ([cdar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdar%29%29) '((7 6 5 4 3 2 1) 8 9)) |
| '(6 5 4 3 2 1) |
| procedure  (**[cddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)).

Example:

|  |
| --- |
| > ([cddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cddr%29%29) '(2 1)) |
| '() |
| procedure  (**[caaar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caaar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*))).

Example:

|  |
| --- |
| > ([caaar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caaar%29%29) '(((6 5 4 3 2 1) 7) 8 9)) |
| 6 |
| procedure  (**[caadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*))).

Example:

|  |
| --- |
| > ([caadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caadr%29%29) '(9 (7 6 5 4 3 2 1) 8)) |
| 7 |
| procedure  (**[cadar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cadar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*))).

Example:

|  |
| --- |
| > ([cadar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cadar%29%29) '((7 6 5 4 3 2 1) 8 9)) |
| 6 |
| procedure  (**[caddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29))) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*))).

Example:

|  |
| --- |
| > ([caddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caddr%29%29) '(3 2 1)) |
| 1 |
| procedure  (**[cdaar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdaar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*))).

Example:

|  |
| --- |
| > ([cdaar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdaar%29%29) '(((6 5 4 3 2 1) 7) 8 9)) |
| '(5 4 3 2 1) |
| procedure  (**[cdadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*))).

Example:

|  |
| --- |
| > ([cdadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdadr%29%29) '(9 (7 6 5 4 3 2 1) 8)) |
| '(6 5 4 3 2 1) |
| procedure  (**[cddar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cddar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*))).

Example:

|  |
| --- |
| > ([cddar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cddar%29%29) '((7 6 5 4 3 2 1) 8 9)) |
| '(5 4 3 2 1) |
| procedure  (**[cdddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29))) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*))).

Example:

|  |
| --- |
| > ([cdddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdddr%29%29) '(3 2 1)) |
| '() |
| procedure  (**[caaaar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caaaar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([caaaar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caaaar%29%29) '((((5 4 3 2 1) 6) 7) 8 9)) |
| 5 |
| procedure  (**[caaadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caaadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([caaadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caaadr%29%29) '(9 ((6 5 4 3 2 1) 7) 8)) |
| 6 |
| procedure  (**[caadar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caadar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([caadar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caadar%29%29) '((7 (5 4 3 2 1) 6) 8 9)) |
| 5 |
| procedure  (**[caaddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caaddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)))) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([caaddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caaddr%29%29) '(9 8 (6 5 4 3 2 1) 7)) |
| 6 |
| procedure  (**[cadaar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cadaar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cadaar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cadaar%29%29) '(((6 5 4 3 2 1) 7) 8 9)) |
| 5 |
| procedure  (**[cadadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cadadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cadadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cadadr%29%29) '(9 (7 6 5 4 3 2 1) 8)) |
| 6 |
| procedure  (**[caddar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._caddar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29))) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([caddar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._caddar%29%29) '((7 6 5 4 3 2 1) 8 9)) |
| 5 |
| procedure  (**[cadddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cadddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)))) |

Returns ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cadddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cadddr%29%29) '(4 3 2 1)) |
| 1 |
| procedure  (**[cdaaar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdaaar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cdaaar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdaaar%29%29) '((((5 4 3 2 1) 6) 7) 8 9)) |
| '(4 3 2 1) |
| procedure  (**[cdaadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdaadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cdaadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdaadr%29%29) '(9 ((6 5 4 3 2 1) 7) 8)) |
| '(5 4 3 2 1) |
| procedure  (**[cdadar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdadar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cdadar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdadar%29%29) '((7 (5 4 3 2 1) 6) 8 9)) |
| '(4 3 2 1) |
| procedure  (**[cdaddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdaddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)))) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cdaddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdaddr%29%29) '(9 8 (6 5 4 3 2 1) 7)) |
| '(5 4 3 2 1) |
| procedure  (**[cddaar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cddaar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cddaar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cddaar%29%29) '(((6 5 4 3 2 1) 7) 8 9)) |
| '(4 3 2 1) |
| procedure  (**[cddadr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cddadr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29))) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cddadr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cddadr%29%29) '(9 (7 6 5 4 3 2 1) 8)) |
| '(5 4 3 2 1) |
| procedure  (**[cdddar](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cdddar%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29))) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cdddar](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdddar%29%29) '((7 6 5 4 3 2 1) 8 9)) |
| '(4 3 2 1) |
| procedure  (**[cddddr](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._cddddr%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) ([cons/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._cons%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)))) |

Returns ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *v*)))).

Example:

|  |
| --- |
| > ([cddddr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cddddr%29%29) '(4 3 2 1)) |
| '() |

4.9.7 Additional List Functions and Synonyms

|  |  |
| --- | --- |
| ([require](https://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._require%29%29) [racket/list](https://docs.racket-lang.org/reference/pairs.html#%28mod-path._racket%2Flist%29)) | package: base |

The bindings documented in this section are provided by the [racket/list](https://docs.racket-lang.org/reference/pairs.html#%28mod-path._racket%2Flist%29) and [racket](https://docs.racket-lang.org/reference/index.html)libraries, but not [racket/base](https://docs.racket-lang.org/reference/index.html).

|  |
| --- |
| value  **[empty](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._empty%29%29)** : [null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) |

The empty list.

Examples:

|  |
| --- |
| > [empty](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._empty%29%29) |
| '() |
| > ([eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29) [empty](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._empty%29%29) [null](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null%29%29)) |
| #t |
| procedure  (**[cons?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._cons~3f%29%29)** *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

The same as ([pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) *v*).

Example:

|  |
| --- |
| > ([cons?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._cons~3f%29%29) '(1 2)) |
| #t |
| procedure  (**[empty?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._empty~3f%29%29)** *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

The same as ([null?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null~3f%29%29) *v*).

Examples:

|  |
| --- |
| > ([empty?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._empty~3f%29%29) '(1 2)) |
| #f |
| > ([empty?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._empty~3f%29%29) '()) |
| #t |
| procedure  (**[first](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._first%29%29)** *lst*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

The same as ([car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) *lst*), but only for lists (that are not empty).

Example:

|  |
| --- |
| > ([first](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._first%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 1 |
| procedure  (**[rest](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._rest%29%29)** *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

The same as ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) *lst*), but only for lists (that are not empty).

Example:

|  |
| --- |
| > ([rest](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._rest%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| '(2 3 4 5 6 7 8 9 10) |
| procedure  (**[second](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._second%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the second element of the list.

Example:

|  |
| --- |
| > ([second](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._second%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 2 |
| procedure  (**[third](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._third%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the third element of the list.

Example:

|  |
| --- |
| > ([third](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._third%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 3 |
| procedure  (**[fourth](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._fourth%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the fourth element of the list.

Example:

|  |
| --- |
| > ([fourth](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._fourth%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 4 |
| procedure  (**[fifth](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._fifth%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the fifth element of the list.

Example:

|  |
| --- |
| > ([fifth](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._fifth%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 5 |
| procedure  (**[sixth](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._sixth%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the sixth element of the list.

Example:

|  |
| --- |
| > ([sixth](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._sixth%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 6 |
| procedure  (**[seventh](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._seventh%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the seventh element of the list.

Example:

|  |
| --- |
| > ([seventh](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._seventh%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 7 |
| procedure  (**[eighth](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._eighth%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the eighth element of the list.

Example:

|  |
| --- |
| > ([eighth](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._eighth%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 8 |
| procedure  (**[ninth](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._ninth%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the ninth element of the list.

Example:

|  |
| --- |
| > ([ninth](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._ninth%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 9 |
| procedure  (**[tenth](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._tenth%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the tenth element of the list.

Example:

|  |
| --- |
| > ([tenth](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._tenth%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 10 |
| procedure  (**[last](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._last%29%29)** *lst*) → [any](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns the last element of the list.

Example:

|  |
| --- |
| > ([last](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._last%29%29) '(1 2 3 4 5 6 7 8 9 10)) |
| 10 |
| procedure  (**[last-pair](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._last-pair%29%29)** *p*) → [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) |
| *p* : [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) |

Returns the last pair of a (possibly improper) list.

Example:

|  |
| --- |
| > ([last-pair](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._last-pair%29%29) '(1 2 3 4)) |
| '(4) |
| procedure  (**[make-list](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._make-list%29%29)** *k* *v*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *k* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns a newly constructed list of length *k*, holding *v* in all positions.

Example:

|  |
| --- |
| > ([make-list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._make-list%29%29) 7 'foo) |
| '(foo foo foo foo foo foo foo) |
| procedure  (**[list-update](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._list-update%29%29)** *lst* *pos* *updater*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *pos* : ([and/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._and%2Fc%29%29) ([>=/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._~3e~3d%2Fc%29%29) 0) ([</c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._~3c%2Fc%29%29) ([length](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._length%29%29) *lst*))) |
| *updater* : ([->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Returns a list that is the same as *lst* except at the specified index. The element at the specified index is (*updater* ([list-ref](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-ref%29%29) *lst* *pos*)).

Example:

|  |
| --- |
| > ([list-update](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._list-update%29%29) '(zero one two) 1 [symbol->string](https://docs.racket-lang.org/reference/symbols.html#%28def._%28%28quote._~23~25kernel%29._symbol-~3estring%29%29)) |
| '(zero "one" two) |

Added in version 6.3 of package base.

|  |
| --- |
| procedure  (**[list-set](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._list-set%29%29)** *lst* *pos* *value*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *pos* : ([and/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._and%2Fc%29%29) ([>=/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._~3e~3d%2Fc%29%29) 0) ([</c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._~3c%2Fc%29%29) ([length](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._length%29%29) *lst*))) |
| *value* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns a list that is the same as *lst* except at the specified index. The element at the specified index is *value*.

Example:

|  |
| --- |
| > ([list-set](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._list-set%29%29) '(zero one two) 2 "two") |
| '(zero one "two") |

Added in version 6.3 of package base.

|  |
| --- |
| procedure  (**[index-of](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._index-of%29%29)** *lst* *v* [*is-equal?*]) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) #f) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *is-equal?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Like [member](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._member%29%29), but returns the index of the first element found instead of the tail of the list.

Example:

|  |
| --- |
| > ([index-of](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._index-of%29%29) '(1 2 3 4) 3) |
| 2 |

Added in version 6.7.0.3 of package base.

|  |
| --- |
| procedure  (**[index-where](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._index-where%29%29)** *lst* *proc*) → ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) #f) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Like [index-of](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._index-of%29%29) but with the predicate-searching behavior of [memf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._memf%29%29).

Example:

|  |
| --- |
| > ([index-where](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._index-where%29%29) '(1 2 3 4) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29)) |
| 1 |

Added in version 6.7.0.3 of package base.

|  |
| --- |
| procedure  (**[indexes-of](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._indexes-of%29%29)** *lst* *v* [*is-equal?*]) |
| → ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29)) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *is-equal?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Like [index-of](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._index-of%29%29), but returns the a list of all the indexes where the element occurs in the list instead of just the first one.

Example:

|  |
| --- |
| > ([indexes-of](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._indexes-of%29%29) '(1 2 1 2 1) 2) |
| '(1 3) |

Added in version 6.7.0.3 of package base.

|  |
| --- |
| procedure  ([**indexes-where**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._indexes-where%29%29) *lst* *proc*) → ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29)) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |

Like [indexes-of](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._indexes-of%29%29) but with the predicate-searching behavior of [index-where](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._index-where%29%29).

Example:

|  |
| --- |
| > ([indexes-where](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._indexes-where%29%29) '(1 2 3 4) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29)) |
| '(1 3) |

Added in version 6.7.0.3 of package base.

|  |
| --- |
| procedure  (**[take](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._take%29%29)** *lst* *pos*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns a fresh list whose elements are the first *pos* elements of *lst*. If *lst* has fewer than *pos* elements, the [exn:fail:contract](https://docs.racket-lang.org/reference/exns.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._exn~3afail~3acontract%29%29) exception is raised.

The *lst* argument need not actually be a list; *lst* must merely start with a chain of at least *pos* pairs.

Examples:

|  |
| --- |
| > ([take](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take%29%29) '(1 2 3 4) 2) |
| '(1 2) |
| > ([take](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take%29%29) 'non-list 0) |
| '() |
| procedure  (**[drop](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._drop%29%29)** *lst* *pos*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Just like [list-tail](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list-tail%29%29).

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| procedure   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | (**[split-at](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._split-at%29%29)** *lst* *pos*) |  | → |  | |  |  |  | | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns the same result as

([values](https://docs.racket-lang.org/reference/values.html#%28def._%28%28quote._~23~25kernel%29._values%29%29) ([take](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take%29%29) *lst* *pos*) ([drop](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._drop%29%29) *lst* *pos*))

except that it can be faster.

|  |
| --- |
| procedure  (**[takef](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._takef%29%29)** *lst* *pred*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |

Returns a fresh list whose elements are taken successively from *lst* as long as they satisfy *pred*. The returned list includes up to, but not including, the first element in *lst* for which *pred* returns #f.

The *lst* argument need not actually be a list; the chain of pairs in *lst* will be traversed until a non-pair is encountered.

Examples:

|  |
| --- |
| > ([takef](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._takef%29%29) '(2 4 5 8) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29)) |
| '(2 4) |
| > ([takef](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._takef%29%29) '(2 4 6 8) [odd?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._odd~3f%29%29)) |
| '() |
| > ([takef](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._takef%29%29) '(2 4 . 6) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29)) |
| '(2 4) |
| procedure  (**[dropf](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._dropf%29%29)** *lst* *pred*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |

Drops elements from the front of *lst* as long as they satisfy *pred*.

Examples:

|  |
| --- |
| > ([dropf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._dropf%29%29) '(2 4 5 8) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29)) |
| '(5 8) |
| > ([dropf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._dropf%29%29) '(2 4 6 8) [odd?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._odd~3f%29%29)) |
| '(2 4 6 8) |
| procedure   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | (**[splitf-at](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._splitf-at%29%29)** *lst* *pred*) |  | → |  | |  |  |  | | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |

Returns the same result as

([values](https://docs.racket-lang.org/reference/values.html#%28def._%28%28quote._~23~25kernel%29._values%29%29) ([takef](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._takef%29%29) *lst* *pred*) ([dropf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._dropf%29%29) *lst* *pred*))

except that it can be faster.

|  |
| --- |
| procedure  (**[take-right](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._take-right%29%29)** *lst* *pos*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns the [list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29)’s *pos*-length tail. If *lst* has fewer than *pos* elements, then the[exn:fail:contract](https://docs.racket-lang.org/reference/exns.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._exn~3afail~3acontract%29%29) exception is raised.

The *lst* argument need not actually be a list; *lst* must merely end with a chain of at least *pos* pairs.

Examples:

|  |
| --- |
| > ([take-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take-right%29%29) '(1 2 3 4) 2) |
| '(3 4) |
| > ([take-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take-right%29%29) 'non-list 0) |
| 'non-list |
| procedure  (**[drop-right](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._drop-right%29%29)** *lst* *pos*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns a fresh list whose elements are the prefix of *lst*, dropping its *pos*-length tail. If *lst* has fewer than *pos* elements, then the [exn:fail:contract](https://docs.racket-lang.org/reference/exns.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._exn~3afail~3acontract%29%29) exception is raised.

The *lst* argument need not actually be a list; *lst* must merely end with a chain of at least *pos* pairs.

Examples:

|  |
| --- |
| > ([drop-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._drop-right%29%29) '(1 2 3 4) 2) |
| '(1 2) |
| > ([drop-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._drop-right%29%29) 'non-list 0) |
| '() |
| procedure   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | ([**split-at-right**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._split-at-right%29%29) *lst* *pos*) |  | → |  | |  |  |  | | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | |
| *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *pos* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns the same result as

([values](https://docs.racket-lang.org/reference/values.html#%28def._%28%28quote._~23~25kernel%29._values%29%29) ([drop-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._drop-right%29%29) *lst* *pos*) ([take-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take-right%29%29) *lst* *pos*))

except that it can be faster.

Examples:

|  |
| --- |
| > ([split-at-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._split-at-right%29%29) '(1 2 3 4 5 6) 3) |
| |  | | --- | | '(1 2 3) | | '(4 5 6) | |
| > ([split-at-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._split-at-right%29%29) '(1 2 3 4 5 6) 4) |
| |  | | --- | | '(1 2) | | '(3 4 5 6) | |
| |  | | --- | | procedure  (**[takef-right](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._takef-right%29%29)** *lst* *pred*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | | *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) | |
| |  | | --- | | procedure  (**[dropf-right](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._dropf-right%29%29)** *lst* *pred*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) | |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | procedure   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | ([**splitf-at-right**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._splitf-at-right%29%29) *lst* *pred*) |  | → |  | |  |  |  | | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | | | *lst* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) | | *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) | |

Like [takef](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._takef%29%29), [dropf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._dropf%29%29), and [splitf-at](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._splitf-at%29%29), but combined with the from-right functionality of [take-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take-right%29%29), [drop-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._drop-right%29%29), and [split-at-right](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._split-at-right%29%29).

|  |
| --- |
| procedure  (**[list-prefix?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._list-prefix~3f%29%29)** *l* *r* [*same?*]) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *l* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *r* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

True if *l* is a prefix of *r*.

Example:

|  |
| --- |
| > ([list-prefix?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._list-prefix~3f%29%29) '(1 2) '(1 2 3 4 5)) |
| #t |

Added in version 6.3 of package base.

|  |
| --- |
| procedure  (**[take-common-prefix](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._take-common-prefix%29%29)** *l* *r* [*same?*]) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *l* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *r* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Returns the longest common prefix of *l* and *r*.

Example:

|  |
| --- |
| > ([take-common-prefix](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._take-common-prefix%29%29) '(a b c d) '(a b x y z)) |
| '(a b) |

Added in version 6.3 of package base.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| procedure   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | (**[drop-common-prefix](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._drop-common-prefix%29%29)** *l* *r* [*same?*]) |  | → |  | |  |  |  | | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | | |
| *l* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *r* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Returns the tails of *l* and *r* with the common prefix removed.

Example:

|  |
| --- |
| > ([drop-common-prefix](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._drop-common-prefix%29%29) '(a b c d) '(a b x y z)) |
| |  | | --- | | '(c d) | | '(x y z) | |

Added in version 6.3 of package base.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| procedure   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | (**[split-common-prefix](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._split-common-prefix%29%29)** *l* *r* [*same?*]) |  | → |  | |  |  |  |  |  | | --- | --- | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | | |
| *l* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *r* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Returns the longest common prefix together with the tails of *l* and *r* with the common prefix removed.

Example:

|  |
| --- |
| > ([split-common-prefix](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._split-common-prefix%29%29) '(a b c d) '(a b x y z)) |
| |  | | --- | | '(a b) | | '(c d) | | '(x y z) | |

Added in version 6.3 of package base.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| procedure   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | (**[add-between](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._add-between%29%29)** |  | *lst* |  |  |  |  | |  |  | *v* |  |  |  |  | |  | [ | #:before-first *before-first* |  |  |  |  | |  |  | #:before-last *before-last* |  |  |  |  | |  |  | #:after-last *after-last* |  |  |  |  | |  |  | #:splice? *splice?*]) |  | → |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *before-first* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) = '() |
| *before-last* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) = *v* |
| *after-last* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) = '() |
| *splice?* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) = #f |

Returns a list with the same elements as *lst*, but with *v* between each pair of elements in *lst*; the last pair of elements will have *before-last* between them, instead of *v* (but *before-last* defaults to *v*).

If *splice?* is true, then *v* and *before-last* should be lists, and the list elements are spliced into the result. In addition, when *splice?* is true, *before-first* and *after-last*are inserted before the first element and after the last element respectively.

Examples:

|  |
| --- |
| > ([add-between](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._add-between%29%29) '(x y z) 'and) |
| '(x and y and z) |
| > ([add-between](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._add-between%29%29) '(x) 'and) |
| '(x) |
| > ([add-between](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._add-between%29%29) '("a" "b" "c" "d") "," #:before-last "and") |
| '("a" "," "b" "," "c" "and" "d") |
| |  | | --- | | > ([add-between](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._add-between%29%29) '(x y z) '(-) #:before-last '(- -) | | #:before-first '(begin) #:after-last '(end LF) | | #:splice? #t) | |
| '(begin x - y - - z end LF) |
| procedure  (**[append\*](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29)** *lst* ... *lsts*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lsts* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29)) |
| ([**append\***](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29) *lst* ... *lsts*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lsts* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [append](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29), but the last argument is used as a list of arguments for [append](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29), so ([append\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29)*lst* [...](https://docs.racket-lang.org/reference/stx-patterns.html#%28form._%28%28lib._racket%2Fprivate%2Fstxcase-scheme..rkt%29._......%29%29) *lsts*) is the same as ([apply](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._apply%29%29) [append](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29) *lst* [...](https://docs.racket-lang.org/reference/stx-patterns.html#%28form._%28%28lib._racket%2Fprivate%2Fstxcase-scheme..rkt%29._......%29%29) *lsts*). In other words, the relationship between [append](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._append%29%29) and [append\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29) is similar to the one between [list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) and [list\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%2A%29%29).

Examples:

|  |
| --- |
| > ([append\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29) '(a) '(b) '((c) (d))) |
| '(a b c d) |
| |  | | --- | | > ([cdr](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cdr%29%29) ([append\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29) ([map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) ([list](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list%29%29) ", " x)) | | '("Alpha" "Beta" "Gamma")))) | |
| '("Alpha" ", " "Beta" ", " "Gamma") |
| procedure  (**[flatten](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._flatten%29%29)** *v*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Flattens an arbitrary S-expression structure of pairs into a list. More precisely, *v* is treated as a binary tree where pairs are interior nodes, and the resulting list contains all of the non-[null](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._null%29%29) leaves of the tree in the same order as an inorder traversal.

Examples:

|  |
| --- |
| > ([flatten](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._flatten%29%29) '((a) b (c (d) . e) ())) |
| '(a b c d e) |
| > ([flatten](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._flatten%29%29) 'a) |
| '(a) |
| procedure   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | (**[check-duplicates](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._check-duplicates%29%29)** |  | *lst* |  |  |  |  | |  | [ | *same?* |  |  |  |  | |  |  | #:key *extract-key*]) |  | → |  | ([or/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._or%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) #f) | |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |
| *extract-key* : ([->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) x) |

Returns the first duplicate item in *lst*. More precisely, it returns the first *x* such that there was a previous *y* where (*same?* (*extract-key* *x*) (*extract-key* *y*)).

The *same?* argument should be an equivalence predicate such as [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) or [eqv?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eqv~3f%29%29) or a dictionary. The procedures [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29), [eqv?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eqv~3f%29%29), and [eq?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._eq~3f%29%29) automatically use a dictionary for speed.

Examples:

|  |
| --- |
| > ([check-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._check-duplicates%29%29) '(1 2 3 4)) |
| #f |
| > ([check-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._check-duplicates%29%29) '(1 2 3 2 1)) |
| 2 |
| > ([check-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._check-duplicates%29%29) '((a 1) (b 2) (a 3)) #:key [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29)) |
| '(a 3) |
| |  | | --- | | > ([check-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._check-duplicates%29%29) '(1 2 3 4 5 6) | | ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x y) ([equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) ([modulo](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._modulo%29%29) x 3) ([modulo](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._modulo%29%29) y 3)))) | |
| 4 |

Added in version 6.3 of package base.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| procedure   |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | (**[remove-duplicates](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._remove-duplicates%29%29)** |  | *lst* |  |  |  |  | |  | [ | *same?* |  |  |  |  | |  |  | #:key *extract-key*]) |  | → |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |
| *extract-key* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) x) |

Returns a list that has all items in *lst*, but without duplicate items, where *same?*determines whether two elements of the list are equivalent. The resulting list is in the same order as *lst*, and for any item that occurs multiple times, the first one is kept.

The #:key argument *extract-key* is used to extract a key value from each list element, so two items are considered equal if (*same?* (*extract-key* x) (*extract-key* y)) is true.

Examples:

|  |
| --- |
| > ([remove-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._remove-duplicates%29%29) '(a b b a)) |
| '(a b) |
| > ([remove-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._remove-duplicates%29%29) '(1 2 1.0 0)) |
| '(1 2 1.0 0) |
| > ([remove-duplicates](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._remove-duplicates%29%29) '(1 2 1.0 0) [=](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._~3d%29%29)) |
| '(1 2 0) |
| procedure  (**[filter-map](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._filter-map%29%29)** *proc* *lst* ...+) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([filter](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) x) ([map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) *proc* *lst* [...](https://docs.racket-lang.org/reference/stx-patterns.html#%28form._%28%28lib._racket%2Fprivate%2Fstxcase-scheme..rkt%29._......%29%29))), but without building the intermediate list.

Example:

|  |
| --- |
| > ([filter-map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._filter-map%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) ([and](https://docs.racket-lang.org/reference/if.html#%28form._%28%28lib._racket%2Fprivate%2Fletstx-scheme..rkt%29._and%29%29) ([positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) x) x)) '(1 2 3 -2 8)) |
| '(1 2 3 8) |
| procedure  (**[count](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._count%29%29)** *proc* *lst* ...+) → [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([length](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._length%29%29) ([filter-map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._filter-map%29%29) *proc* *lst* [...](https://docs.racket-lang.org/reference/stx-patterns.html#%28form._%28%28lib._racket%2Fprivate%2Fstxcase-scheme..rkt%29._......%29%29))), but without building the intermediate list.

Example:

|  |
| --- |
| > ([count](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._count%29%29) [positive?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._positive~3f%29%29) '(1 -1 2 3 -2 5)) |
| 4 |
| procedure   |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | | (**[partition](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._partition%29%29)** *pred* *lst*) |  | → |  | |  |  |  | | --- | --- | --- | | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |  | [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) | | |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Similar to [filter](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29), except that two values are returned: the items for which *pred* returns a true value, and the items for which *pred* returns #f.

The result is the same as

([values](https://docs.racket-lang.org/reference/values.html#%28def._%28%28quote._~23~25kernel%29._values%29%29) ([filter](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29) *pred* *lst*) ([filter](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29) ([negate](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28lib._racket%2Ffunction..rkt%29._negate%29%29) *pred*) *lst*))

but *pred* is applied to each item in *lst* only once.

Example:

|  |
| --- |
| > ([partition](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._partition%29%29) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29) '(1 2 3 4 5 6)) |
| |  | | --- | | '(2 4 6) | | '(1 3 5) | |
| procedure  (**[range](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29)** *end*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *end* : [real?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._real~3f%29%29) |
| ([**range**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) *start* *end* [*step*]) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *start* : [real?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._real~3f%29%29) |
| *end* : [real?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._real~3f%29%29) |
| *step* : [real?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._real~3f%29%29) = 1 |

Similar to [in-range](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._in-range%29%29), but returns lists.

The resulting list holds numbers starting at *start* and whose successive elements are computed by adding *step* to their predecessor until *end* (excluded) is reached. If no starting point is provided, 0 is used. If no *step* argument is provided, 1 is used.

Like [in-range](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._in-range%29%29), a [range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) application can provide better performance when it appears directly in a [for](https://docs.racket-lang.org/reference/for.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._for%29%29) clause.

Examples:

|  |
| --- |
| > ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 10) |
| '(0 1 2 3 4 5 6 7 8 9) |
| > ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 10 20) |
| '(10 11 12 13 14 15 16 17 18 19) |
| > ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 20 40 2) |
| '(20 22 24 26 28 30 32 34 36 38) |
| > ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 20 10 -1) |
| '(20 19 18 17 16 15 14 13 12 11) |
| > ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 10 15 1.5) |
| '(10 11.5 13.0 14.5) |

Changed in version 6.7.0.4 of package base: Adjusted to cooperate with [for](https://docs.racket-lang.org/reference/for.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._for%29%29) in the same way that [in-range](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._in-range%29%29) does.

|  |
| --- |
| procedure  (**[append-map](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._append-map%29%29)** *proc* *lst* ...+) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *proc* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns ([append\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append%2A%29%29) ([map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Fmap..rkt%29._map%29%29) *proc* *lst* [...](https://docs.racket-lang.org/reference/stx-patterns.html#%28form._%28%28lib._racket%2Fprivate%2Fstxcase-scheme..rkt%29._......%29%29))).

Example:

|  |
| --- |
| > ([append-map](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._append-map%29%29) [vector->list](https://docs.racket-lang.org/reference/vectors.html#%28def._%28%28quote._~23~25kernel%29._vector-~3elist%29%29) '(#(1) #(2 3) #(4))) |
| '(1 2 3 4) |
| procedure  (**[filter-not](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._filter-not%29%29)** *pred* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *pred* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [filter](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Fprivate%2Flist..rkt%29._filter%29%29), but the meaning of the *pred* predicate is reversed: the result is a list of all items for which *pred* returns #f.

Example:

|  |
| --- |
| > ([filter-not](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._filter-not%29%29) [even?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._even~3f%29%29) '(1 2 3 4 5 6)) |
| '(1 3 5) |
| procedure  (**[shuffle](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._shuffle%29%29)** *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns a list with all elements from *lst*, randomly shuffled.

Examples:

|  |
| --- |
| > ([shuffle](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._shuffle%29%29) '(1 2 3 4 5 6)) |
| '(3 6 5 4 2 1) |
| > ([shuffle](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._shuffle%29%29) '(1 2 3 4 5 6)) |
| '(2 6 4 1 3 5) |
| > ([shuffle](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._shuffle%29%29) '(1 2 3 4 5 6)) |
| '(2 3 5 6 4 1) |
| procedure  (**[combinations](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._combinations%29%29)** *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| ([**combinations**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._combinations%29%29) *lst* *size*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *size* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Wikipedia [combinations](https://en.wikipedia.org/wiki/Combination)

Return a list of all combinations of elements in the input list (aka the powerset of *lst*). If *size* is given, limit results to combinations of *size* elements.

Examples:

|  |
| --- |
| > ([combinations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._combinations%29%29) '(1 2 3)) |
| '(() (1) (2) (1 2) (3) (1 3) (2 3) (1 2 3)) |
| > ([combinations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._combinations%29%29) '(1 2 3) 2) |
| '((1 2) (1 3) (2 3)) |
| procedure  (**[in-combinations](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._in-combinations%29%29)** *lst*) → [sequence?](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._sequence~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| ([**in-combinations**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._in-combinations%29%29) *lst* *size*) → [sequence?](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._sequence~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *size* : [exact-nonnegative-integer?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._exact-nonnegative-integer~3f%29%29) |

Returns a sequence of all combinations of elements in the input list, or all combinations of length *size* if *size* is given. Builds combinations one-by-one instead of all at once.

Examples:

|  |
| --- |
| > ([time](https://docs.racket-lang.org/reference/time.html#%28form._%28%28lib._racket%2Fprivate%2Fmore-scheme..rkt%29._time%29%29) ([begin](https://docs.racket-lang.org/reference/begin.html#%28form._%28%28quote._~23~25kernel%29._begin%29%29) ([combinations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._combinations%29%29) ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 15)) ([void](https://docs.racket-lang.org/reference/void.html#%28def._%28%28quote._~23~25kernel%29._void%29%29)))) |
| cpu time: 61 real time: 16 gc time: 0 |
| > ([time](https://docs.racket-lang.org/reference/time.html#%28form._%28%28lib._racket%2Fprivate%2Fmore-scheme..rkt%29._time%29%29) ([begin](https://docs.racket-lang.org/reference/begin.html#%28form._%28%28quote._~23~25kernel%29._begin%29%29) ([in-combinations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._in-combinations%29%29) ([range](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._range%29%29) 15)) ([void](https://docs.racket-lang.org/reference/void.html#%28def._%28%28quote._~23~25kernel%29._void%29%29)))) |
| cpu time: 0 real time: 1 gc time: 0 |
| procedure  (**[permutations](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._permutations%29%29)** *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns a list of all permutations of the input list. Note that this function works without inspecting the elements, and therefore it ignores repeated elements (which will result in repeated permutations).

Examples:

|  |
| --- |
| > ([permutations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._permutations%29%29) '(1 2 3)) |
| '((1 2 3) (2 1 3) (1 3 2) (3 1 2) (2 3 1) (3 2 1)) |
| > ([permutations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._permutations%29%29) '(x x)) |
| '((x x) (x x)) |
| procedure  (**[in-permutations](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._in-permutations%29%29)** *lst*) → [sequence?](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._sequence~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns a sequence of all permutations of the input list. It is equivalent to ([in-list](https://docs.racket-lang.org/reference/sequences.html#%28def._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._in-list%29%29)([permutations](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._permutations%29%29) l)) but much faster since it builds the permutations one-by-one on each iteration

|  |
| --- |
| procedure  (**[argmin](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._argmin%29%29)** *proc* *lst*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *proc* : ([->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [real?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._real~3f%29%29)) |
| *lst* : ([and/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._and%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29)) |

Returns the first element in the list *lst* that minimizes the result of *proc*. Signals an error on an empty list. See also [min](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._min%29%29).

Examples:

|  |
| --- |
| > ([argmin](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._argmin%29%29) [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) '((3 pears) (1 banana) (2 apples))) |
| '(1 banana) |
| > ([argmin](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._argmin%29%29) [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) '((1 banana) (1 orange))) |
| '(1 banana) |
| procedure  (**[argmax](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._argmax%29%29)** *proc* *lst*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *proc* : ([->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [real?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._real~3f%29%29)) |
| *lst* : ([and/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._and%2Fc%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29)) |

Returns the first element in the list *lst* that maximizes the result of *proc*. Signals an error on an empty list. See also [max](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._max%29%29).

Examples:

|  |
| --- |
| > ([argmax](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._argmax%29%29) [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) '((3 pears) (1 banana) (2 apples))) |
| '(3 pears) |
| > ([argmax](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._argmax%29%29) [car](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._car%29%29) '((3 pears) (3 oranges))) |
| '(3 pears) |
| procedure  (**[group-by](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._group-by%29%29)** *key* *lst* [*same?*]) → ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29)) |
| *key* : ([->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *same?* : ([any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) . [->](https://docs.racket-lang.org/reference/function-contracts.html#%28form._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._-~3e%29%29) . [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29)) = [equal?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._equal~3f%29%29) |

Groups the given list into equivalence classes, with equivalence being determined by *same?*. Within each equivalence class, [group-by](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._group-by%29%29) preserves the ordering of the original list. Equivalence classes themselves are in order of first appearance in the input.

Example:

|  |
| --- |
| > ([group-by](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._group-by%29%29) ([lambda](https://docs.racket-lang.org/reference/lambda.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._lambda%29%29) (x) ([modulo](https://docs.racket-lang.org/reference/generic-numbers.html#%28def._%28%28quote._~23~25kernel%29._modulo%29%29) x 3)) '(1 2 1 2 54 2 5 43 7 2 643 1 2 0)) |
| '((1 1 43 7 643 1) (2 2 2 5 2 2) (54 0)) |

Added in version 6.3 of package base.

|  |
| --- |
| procedure  (**[cartesian-product](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._cartesian-product%29%29)** *lst* ...) → ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29)) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Computes the n-ary cartesian product of the given lists.

Examples:

|  |
| --- |
| > ([cartesian-product](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._cartesian-product%29%29) '(1 2 3) '(a b c)) |
| '((1 a) (1 b) (1 c) (2 a) (2 b) (2 c) (3 a) (3 b) (3 c)) |
| > ([cartesian-product](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._cartesian-product%29%29) '(4 5 6) '(d e f) '(#t #f)) |
| |  | | --- | | '((4 d #t) | | (4 d #f) | | (4 e #t) | | (4 e #f) | | (4 f #t) | | (4 f #f) | | (5 d #t) | | (5 d #f) | | (5 e #t) | | (5 e #f) | | (5 f #t) | | (5 f #f) | | (6 d #t) | | (6 d #f) | | (6 e #t) | | (6 e #f) | | (6 f #t) | | (6 f #f)) | |

Added in version 6.3 of package base.

|  |
| --- |
| procedure  (**[remf](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._remf%29%29)** *pred* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Returns a list that is like *lst*, omitting the first element of *lst* for which *pred* produces a true value.

Example:

|  |
| --- |
| > ([remf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._remf%29%29) [negative?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._negative~3f%29%29) '(1 -2 3 4 -5)) |
| '(1 3 4 -5) |

Added in version 6.3 of package base.

|  |
| --- |
| procedure  (**[remf\*](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28lib._racket%2Flist..rkt%29._remf%2A%29%29)** *pred* *lst*) → [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |
| *pred* : [procedure?](https://docs.racket-lang.org/reference/procedures.html#%28def._%28%28quote._~23~25kernel%29._procedure~3f%29%29) |
| *lst* : [list?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._list~3f%29%29) |

Like [remf](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._remf%29%29), but removes all the elements for which *pred* produces a true value.

Example:

|  |
| --- |
| > ([remf\*](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28lib._racket%2Flist..rkt%29._remf%2A%29%29) [negative?](https://docs.racket-lang.org/reference/number-types.html#%28def._%28%28quote._~23~25kernel%29._negative~3f%29%29) '(1 -2 3 4 -5)) |
| '(1 3 4) |

Added in version 6.3 of package base.

4.9.8 Immutable Cyclic Data

|  |
| --- |
| procedure  (**[make-reader-graph](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29)** *v*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns a value like *v*, with placeholders created by [make-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-placeholder%29%29) replaced with the values that they contain, and with placeholders created by [make-hash-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-hash-placeholder%29%29) with an immutable hash table. No part of *v* is mutated; instead, parts of *v* are copied as necessary to construct the resulting graph, where at most one copy is created for any given value.

Since the copied values can be immutable, and since the copy is also immutable, [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29) can create cycles involving only immutable pairs, vectors, boxes, and hash tables.

Only the following kinds of values are copied and traversed to detect placeholders:

* pairs
* vectors, both mutable and immutable
* boxes, both mutable and immutable
* hash tables, both mutable and immutable
* instances of a [prefab](https://docs.racket-lang.org/reference/structures.html#%28tech._prefab%29) structure type
* placeholders created by [make-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-placeholder%29%29) and [make-hash-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-hash-placeholder%29%29)

Due to these restrictions, [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29) creates exactly the same sort of cyclic values as [read](https://docs.racket-lang.org/reference/Reading.html#%28def._%28%28quote._~23~25kernel%29._read%29%29).

Example:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | > ([let\*](https://docs.racket-lang.org/reference/let.html#%28form._%28%28lib._racket%2Fprivate%2Fletstx-scheme..rkt%29._let%2A%29%29) ([ph ([make-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-placeholder%29%29) #f)] | | [x ([cons](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._cons%29%29) 1 ph)]) | | ([placeholder-set!](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder-set%21%29%29) ph x) | | ([make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29) x)) | |
| #0='(1 . #0#) |
| procedure  ([**placeholder?**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder~3f%29%29) *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns #t if *v* is a placeholder created by [make-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-placeholder%29%29), #f otherwise.

|  |
| --- |
| procedure  (**[make-placeholder](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._make-placeholder%29%29)** *v*) → [placeholder?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns a placeholder for use with [placeholder-set!](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder-set%21%29%29) and [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29). The *v*argument supplies the initial value for the placeholder.

|  |
| --- |
| procedure  ([**placeholder-set!**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder-set%21%29%29) *ph* *datum*) → [void?](https://docs.racket-lang.org/reference/void.html#%28def._%28%28quote._~23~25kernel%29._void~3f%29%29) |
| *ph* : [placeholder?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder~3f%29%29) |
| *datum* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Changes the value of *ph* to v.

|  |
| --- |
| procedure  ([**placeholder-get**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder-get%29%29) *ph*) → [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |
| *ph* : [placeholder?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._placeholder~3f%29%29) |

Returns the value of *ph*.

|  |
| --- |
| procedure  (**[hash-placeholder?](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._hash-placeholder~3f%29%29)** *v*) → [boolean?](https://docs.racket-lang.org/reference/booleans.html#%28def._%28%28quote._~23~25kernel%29._boolean~3f%29%29) |
| *v* : [any/c](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fprivate%2Fmisc..rkt%29._any%2Fc%29%29) |

Returns #t if *v* is a placeholder created by [make-hash-placeholder](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-hash-placeholder%29%29), #f otherwise.

|  |
| --- |
| procedure  (**[make-hash-placeholder](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._make-hash-placeholder%29%29)** *assocs*) → [hash-placeholder?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._hash-placeholder~3f%29%29) |
| *assocs* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Like [make-immutable-hash](https://docs.racket-lang.org/reference/hashtables.html#%28def._%28%28quote._~23~25kernel%29._make-immutable-hash%29%29), but produces a table placeholder for use with [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29).

|  |
| --- |
| procedure  (**[make-hasheq-placeholder](https://docs.racket-lang.org/reference/pairs.html" \l "%28def._%28%28quote._~23~25kernel%29._make-hasheq-placeholder%29%29)** *assocs*) → [hash-placeholder?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._hash-placeholder~3f%29%29) |
| *assocs* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Like [make-immutable-hasheq](https://docs.racket-lang.org/reference/hashtables.html#%28def._%28%28quote._~23~25kernel%29._make-immutable-hasheq%29%29), but produces a table placeholder for use with [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29).

|  |
| --- |
| procedure  ([**make-hasheqv-placeholder**](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-hasheqv-placeholder%29%29) *assocs*) → [hash-placeholder?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._hash-placeholder~3f%29%29) |
| *assocs* : ([listof](https://docs.racket-lang.org/reference/data-structure-contracts.html#%28def._%28%28lib._racket%2Fcontract%2Fbase..rkt%29._listof%29%29) [pair?](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._pair~3f%29%29)) |

Like [make-immutable-hasheqv](https://docs.racket-lang.org/reference/hashtables.html#%28def._%28%28quote._~23~25kernel%29._make-immutable-hasheqv%29%29), but produces a table placeholder for use with [make-reader-graph](https://docs.racket-lang.org/reference/pairs.html#%28def._%28%28quote._~23~25kernel%29._make-reader-graph%29%29).