Package ‘inops’

October 13, 2022

Title Infix Operators for Detection, Subsetting and Replacement

Version 0.0.1

Description Infix operators to detect, subset, and replace the elements matched by a given condition.

The functions have several variants of operator types, including subsets, ranges, regular expressions and others.

Implemented operators work on vectors, matrices, and lists.

Depends R (>= 3.1.0)

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests testthat (>= 2.1.0), purrr, knitr, rmarkdown, dplyr, nycflights13

URL https://github.com/moodymudskipper/inops

BugReports https://github.com/moodymudskipper/inops/issues

VignetteBuilder knitr

NeedsCompilation no

Author Antoine Fabri [aut, cre], Karolis Koncevičius [aut]

Maintainer Antoine Fabri <antoine.fabri@gmail.com>

Repository CRAN

Date/Publication 2019-11-19 12:40:03 UTC

R topics documented:

- comparison_replace .................................................. 2
- comparison_subset .................................................. 3
- in_detect .................................................................. 4
- in_replace ................................................................. 6
- in_subset ................................................................. 8
- out .......................................................................... 10
Description

Operators for replacing values using the standard comparison operators.

Usage

\[ x \geq y \leftarrow value \]
\[ x > y \leftarrow value \]
\[ x \leq y \leftarrow value \]
\[ x < y \leftarrow value \]
\[ x == y \leftarrow value \]
\[ x != y \leftarrow value \]

Arguments

\[ x \] first element of the operation.
\[ y \] second element of the operation.
\[ value \] replacement value.

Details

Thanks to these operators:

- \[ x == y \leftarrow value \] is equivalent to \[ x[x == y] \leftarrow value \]
- \[ x != y \leftarrow value \] is equivalent to \[ x[x != y] \leftarrow value \]
- \[ x <= y \leftarrow value \] is equivalent to \[ x[x <= y] \leftarrow value \]
- \[ x >= y \leftarrow value \] is equivalent to \[ x[x >= y] \leftarrow value \]
- \[ x < y \leftarrow value \] is equivalent to \[ x[x < y] \leftarrow value \]
- \[ x > y \leftarrow value \] is equivalent to \[ x[x > y] \leftarrow value \]

Value

\[ x \] with values for which the comparisons evaluate to TRUE replaced with \[ value \].

See Also

`==`
**Examples**

```r
ages <- c(130, 10, 1996, 21, 39, 74, -2, 0)

ages == 1996 <- as.numeric(format(Sys.Date(), "%Y")) - 1986
ages

ages > 100 <- NA
ages

ages <= 0 <- NA
ages
```

---

**Description**

Operators for subsetting values using the standard comparison operators.

**Usage**

```r
x %[>=% y
x %[>% y
x %[<=% y
x %[<% y
x %[==% y
x %[!=% y
```

**Arguments**

- `x` first element of the operation.
- `y` second element of the operation.

**Value**

Elements of `x` matched by the used comparison.

**See Also**

`'=='`
Examples

ages <- c(130, 10, 21, 39, 74, -2, 0)
ages %<% 5

letters %==% "a"
letters %!=% "a"

---

in_detect  Matching Values and Intervals

Description

Operators for detecting which values are within a given interval or set.

Usage

x %in{}% table
x %out{}% table
x %in[]% interval
x %out[]% interval
x %in()% interval
x %out()% interval
x %in()% interval
x %out()% interval
x %in~% pattern
x %out~% pattern
x %in~p% pattern
x %out~p% pattern
x %in~f% pattern
in_detect

x %out-f% pattern
x %in#% count
x %out#% count

Arguments

- x: vector or array of values to be matched.
- table: vector or list to be matched against.
- interval: numeric vector defining a range to be matched against.
- pattern: pattern to be matched against.
- count: numeric vector defining counts for count-based selection.

Details

Compared with default %in% implementation in R the operators implemented here try to be more consistent with other default infix operators like == and <. In particular they preserve the dimensions and the missing values (see examples).

Style of parentheses define the type of matching template:

- %in{}% detects which elements of x are present in the set given by the table argument.
- %in(), %in[], %in() and %in[] detect the elements of x included in a range of interval argument, using range(interval). This range being closed, open on both sides, open on the left, or open on the right, respectively.
- %in~%, %in~p% and %in~f% detect the elements of x that match the regular expression given by pattern. They wrap grepl() with the default parameters of perl = TRUE, and with fixed = TRUE, respectively.
- %in#% detects the elements that occur a specified number of times. Operators of the form %out<suffix>% return the negation of %in<suffix>%

Value

a logical vector or an array of the same dimensions as x indicating if each value of x is within the defined subset.

See Also

%in%

Examples

# difference in behaviour with dimensions when compared to %in%
iris[1:10,] %in% "setosa"
iris[1:10,] == "setosa"
iris[1:10,] %in()% "setosa"
# difference in behaviour with missing values when compared to %in%

```r
x <- c(1,2,3,NA,4)
x %in% c(1,2,3)
x %in{}% c(1,2,3)
```

# other interval operators

```r
x <- 1:10
x %in{[]}% c(3,7)
x %in()% c(3,7)
x %in()% c(3,7)
x %in[])% c(3,7)
x %out[])% c(3,7)
```

# when more than 2 numbers are provided for the interval - range is used

```r
x <- 1:10
all.equal(x %in[]% c(2,4), x %in[]% c(2,3,4))
all.equal(x %in[]% c(2,4), x %in[]% range(c(2,3,4)))
```

# matching according to regular expressions

```r
iris$Species %in~% "^v"
iris$Species %in-f% "^v"
iris$Species %in-f% "versicolor"
iris$Species %in-f% c("versicolor", "virginica")
```

# selecting by number of occurances

```r
mtcars$gear %in#% 1:5
mtcars$gear %out#% 1:5
```
Arguments

- **x**: vector or array of values to be matched.
- **table**: vector or list to be matched against.
- **value**: replacement value.
- **interval**: numeric vector defining a range to be matched against.
- **pattern**: pattern to be matched against.
- **count**: numeric vector defining counts for count-based selection.

Details

For each `%*%<-` operator of this package `x %*% y <- value` is a shorthand for `x[x %*% y] <- value`.

Value

- `x` with specified values replaced with `value`.

See Also

- `%in{}`
Examples

# interval replacement operators
x <- 1:10
x %in% c(3,7) <- 0
x

x <- 1:10
x %in% c(3,7) <- NA
x

x <- 1:10
x %out% c(3,7) <- x
x

# regular expression replacement operators
region <- as.character(state.region)
table(region)
region %in% "^North" <- "North"
table(region)

# count based replacement operators
carb <- mtcars$carb
table(carb, useNA="always")
carb %in#% 1 <- NA
table(carb, useNA="always")

in_subset

Subsetting Values and Intervals

Description

Operators for subsetting values within a given interval or set.

Usage

x %in% table
x %out% table
x %in% interval
x %out% interval
x %in()% interval
x %out()% interval
Arguments

x vector or array of values to be matched.

table vector or list to be matched against.

interval numeric vector defining a range to be matched against.

pattern pattern to be matched against.

count numeric vector defining counts for count-based selection.

Details

For each %[*% operator of this package x %[*% y is a shorthand for x[x %*[y].

Value

elements of x matched by the used infix operator type.

See Also

%in{}%
Examples

# interval subsetting operators
x <- 1:10
x %in% c(3,7)
x %in% c(3,7)
x %out% c(3,7)

# regular expression subsetting operators
carnames <- rownames(mtcars)
carnames %in% "Mazda"
carnames %in% c("Mazda", "Merc")
carnames %in% c("\w{10,100}$") # long car names

# count-based subsetting operators
mtcars$cyl %in% 1:10
mtcars$cyl %out% 1:10

Description

%out% is the negation of %in%, so x %out% y is equivalent to ! x %in% y.

Usage

x %out% table

Arguments

x vector of values to be matched.

Value

a logical vector or of the same length as x indicating if each value of x is within the defined subset.

See Also

%in%

Examples

iris$Species %in% c("setosa", "versicolor")
iris$Species %out% c("setosa", "versicolor")