# Package 'index0'

May 8, 2026

**Title** Zero-Based Indexing in R

**Version** 0.0.1

**Description**
Extract and replace elements using indices that start from zero (rather than one), as is common in mathematical notation and other programming languages.

**License** MIT + file LICENSE

**Language** en-GB

**Encoding** UTF-8

**RoxygenNote** 7.1.2

**Suggests** testthat (>= 3.0.0)

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** David Antony Selby [aut, cre] (ORCID:
<https://orcid.org/0000-0001-8026-5663>)

**Maintainer** David Antony Selby <david.selby@manchester.ac.uk>

**Repository** CRAN

**Date/Publication** 2021-12-03 08:20:05 UTC

## Contents

## head.index0 *Return the First or Last Parts of a Zero-Indexed Object*

### Description

Works like `utils::head()` and `utils::tail()`.

### Usage

```
## S3 method for class 'index0'
head(x, ...)

## S3 method for class 'index0'
tail(x, ...)
```

### Arguments

| | |
|---|---|
| `x` | An `index0` object |
| `...` | Other arguments, passed to generic function |

### Details

Just because an object is zero-indexed, doesn't mean that the definition of, for example, "the first 5 elements" or "the last two elements" has changed. Thus we add methods `head()` and `tail()` to ensure they behave as normal.

### Value

An `index0` object

## index0 *Zero-based indexing of vectors*

### Description

Normally R is indexed from 1, but with the special `index0` class, you can have vectors that are indexed from zero. Works both for subsetting (extraction) and (sub-)assignment. An `index0` object is just like a normal vector or matrix, but `x[i]` returns or replaces the `(i+1)`th index.

## Usage

```
## S3 method for class 'index0'
x[i, j, ...]

## S3 replacement method for class 'index0'
x[i, j, ...] <- value

as.index0(x)

as.index1(x)

is.index0(x)

index_from_0(x)

## S3 method for class 'index0'
print(x, ...)
```

## Arguments

| | |
|---|---|
| `x` | object from which to extract element(s) or in which to replace element(s) |
| `i, j` | indices specifying elements to extract or replace. Starting from 1. |
| `...` | other arguments passed to generic methods. |
| `value` | typically an array-like R object of a similar class as `x`. |

## Details

Assign the class `index0` to a vector, using `as.index0()` or `index_from_0()`, then use the subset operators normally and they will be indexed from zero. You can reverse the operation (reset to indexing from 1) with `as.index1()` or by manually removing the `index0` class. Character indices *seem* to be unaffected. Be cautious with logical indices. See examples.

## Value

`as.index0` returns the input (typically a vector or matrix) unchanged except for the addition of an `index0` class attribute, which enables the zero-based indexing behaviour. Use `as.index1` to remove this class again, if present.

If `x` is a zero-indexed object with class `index0`, then `x[i]` returns an appropriate subset of `x`. The returned subset is also zero-indexed. `x[i] <- value` changes the `i`th element (effectively `(i+1)`th element in ordinary R code) in place.

`is.index0(x)` returns `TRUE` if `x` is indexed from zero, otherwise `FALSE`.

## Source

Partially inspired by this Stack Overflow answer: Zero based arrays/vectors in R

## Examples

```
# Vectors
v <- as.index0(letters)
v[0:3]
v[c(0, 2)] <- c('zeroth', 'second')
v

# Matrices and arrays
m <- index_from_0(matrix(1:4, 2))
m[0, 1]
m[0, 1] <- 99
m
```

# Index