|  |  |
| --- | --- |
| sample {base} | R Documentation |

**Random Samples and Permutations**

**Description**

sample takes a sample of the specified size from the elements of x using either with or without replacement.

**Usage**

sample(x, size, replace = FALSE, prob = NULL)

sample.int(n, size = n, replace = FALSE, prob = NULL)

**Arguments**

|  |  |
| --- | --- |
| x | Either a vector of one or more elements from which to choose, or a positive integer. See ‘Details.’ |
| n | a positive number, the number of items to choose from. See ‘Details.’ |
| size | a non-negative integer giving the number of items to choose. |
| replace | Should sampling be with replacement? |
| prob | A vector of probability weights for obtaining the elements of the vector being sampled. |

**Details**

If x has length 1, is numeric (in the sense of [is.numeric](http://127.0.0.1:14695/library/base/help/is.numeric)) and x >= 1, sampling *via* sample takes place from 1:x. *Note* that this convenience feature may lead to undesired behaviour when x is of varying length in calls such as sample(x). See the examples.

Otherwise x can be any **R** object for which length and subsetting by integers make sense: S3 or S4 methods for these operations will be dispatched as appropriate.

For sample the default for size is the number of items inferred from the first argument, so that sample(x) generates a random permutation of the elements of x (or 1:x).

It is allowed to ask for size = 0 samples with n = 0 or a length-zero x, but otherwise n > 0 or positive length(x) is required.

Non-integer positive numerical values of n or x will be truncated to the next smallest integer, which has to be no larger than [.Machine](http://127.0.0.1:14695/library/base/help/.Machine)$integer.max.

The optional prob argument can be used to give a vector of weights for obtaining the elements of the vector being sampled. They need not sum to one, but they should be non-negative and not all zero. If replace is true, Walker's alias method (Ripley, 1987) is used when there are more than 200 reasonably probable values: this gives results incompatible with those from **R** < 2.2.0.

If replace is false, these probabilities are applied sequentially, that is the probability of choosing the next item is proportional to the weights amongst the remaining items. The number of nonzero weights must be at least size in this case.

sample.int is a bare interface in which both n and size must be supplied as integers.

As from **R** 3.0.0, n can be larger than the largest integer of type integer, up to the largest representable integer in type double. Only uniform sampling is supported. Two random numbers are used to ensure uniform sampling of large integers.

**Value**

For sample a vector of length size with elements drawn from either x or from the integers 1:x.

For sample.int, an integer vector of length size with elements from 1:n, or a double vector if *n >= 2^31*.

**References**

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

Ripley, B. D. (1987) *Stochastic Simulation*. Wiley.

**See Also**

[RNG](http://127.0.0.1:14695/library/base/help/RNG) about random number generation.

CRAN package [**sampling**](https://cran.r-project.org/package=sampling) for other methods of weighted sampling without replacement.

**Examples**

x <- 1:12

# a random permutation

sample(x)

# bootstrap resampling -- only if length(x) > 1 !

sample(x, replace = TRUE)

# 100 Bernoulli trials

sample(c(0,1), 100, replace = TRUE)

## More careful bootstrapping -- Consider this when using sample()

## programmatically (i.e., in your function or simulation)!

# sample()'s surprise -- example

x <- 1:10

sample(x[x > 8]) # length 2

sample(x[x > 9]) # oops -- length 10!

sample(x[x > 10]) # length 0

resample <- function(x, ...) x[sample.int(length(x), ...)]

resample(x[x > 8]) # length 2

resample(x[x > 9]) # length 1

resample(x[x > 10]) # length 0

## R 3.x.y only

sample.int(1e10, 12, replace = TRUE)

sample.int(1e10, 12) # not that there is much chance of duplicates