

Package ‘wakefield’

August 29, 2016

Title Generate Random Data Sets

Version 0.3.0

Maintainer Tyler Rinker <tyler.rinker@gmail.com>

Description Generates random data sets including: data.frames, lists,
and vectors.

Depends R (>= 3.1.2)

Imports chron, ggplot2, dplyr, stringi

Suggests testthat

Date 2016-06-15

License GPL-2

LazyData TRUE

URL <https://github.com/trinker/wakefield>

BugReports <https://github.com/trinker/wakefield/issues>

Collate 'utils.R' 'r_sample.R' 'age.R' 'r_sample_factor.R' 'animal.R'
'r_sample_binary.R' 'answer.R' 'area.R' 'as_integer.R' 'car.R'
'children.R' 'coin.R' 'color.R' 'date_stamp.R'
'r_sample_logical.R' 'death.R' 'dice.R' 'dna.R' 'dob.R'
'dummy.R' 'education.R' 'employment.R' 'eye.R' 'grade.R'
'grade_level.R' 'group.R' 'hair.R' 'normal.R' 'height.R'
'hour.R' 'id.R' 'income.R' 'internet_browser.R' 'interval.R'
'iq.R' 'language.R' 'level.R' 'r_sample_ordered.R' 'likert.R'
'lorem_ipsum.R' 'marital.R' 'military.R' 'minute.R' 'month.R'
'r_sample_replace.R' 'wakefield-package.R' 'name.R' 'peek.R'
'political.R' 'probs.R' 'r_data.R' 'r_data_frame.R' 'r_dummy.R'
'seriesname.R' 'r_insert.R' 'r_list.R' 'r_na.R'
'r_sample_integer.R' 'r_series.R' 'race.R' 'relate.R'
'religion.R' 'sat.R' 'second.R' 'sentence.R' 'sex.R'
'sex_inclusive.R' 'smokes.R' 'speed.R' 'state.R' 'string.R'
'table_heat.R' 'time_stamp.R' 'upper.R' 'valid.R' 'variables.R'
'varname.R' 'year.R' 'zip_code.R'

RoxygenNote 5.0.1

NeedsCompilation no

Author Tyler Rinker [aut, cre],
 Josh O'Brien [ctb],
 Ananda Mahto [ctb],
 Matthew Sigal [ctb]

Repository CRAN

Date/Publication 2016-06-16 06:46:28

R topics documented:

age	4
animal	5
animal_list	6
answer	6
area	7
as_integer	8
car	9
children	10
coin	11
color	12
date_stamp	13
death	14
dice	15
dna	16
dob	17
dummy	18
education	19
employment	20
eye	21
grade	22
grade_level	24
grady_augmented	25
group	25
hair	26
height	27
hour	29
id	30
income	31
internet_browser	32
interval	33
iq	34
language	35
languages	36
level	37
likert	38
lorem_ipsum	39
marital	40

military	41
minute	42
month	43
name	44
name_neutral	45
normal	45
peek	46
plot.tbl_df	47
political	48
presidential_debates_2012	49
print.available	49
print.variable	50
probs	50
race	51
relate	52
religion	53
r_data	54
r_data_frame	56
r_dummy	58
r_insert	59
r_list	60
r_na	61
r_sample	62
r_sample_binary	63
r_sample_factor	64
r_sample_integer	64
r_sample_logical	65
r_sample_ordered	66
r_sample_replace	67
r_series	68
sat	69
second	71
sentence	72
seriesname	73
sex	73
sex_inclusive	74
smokes	76
speed	77
state	78
state_populations	80
string	81
table_heat	82
time_stamp	83
upper	84
valid	85
variables	86
varname	87
wakefield	87

year	88
zip_code	89

Index	90
--------------	-----------

age	<i>Generate Random Vector of Ages</i>
-----	---------------------------------------

Description

Generate a random vector of ages.

Usage

```
age(n, x = 20:35, prob = NULL, name = "Age")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of gender elements.

See Also

Other variable.functions: [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
age(10)
hist(age(10000))
interval(age, 3, n = 1000)
```

animal	<i>Generate Random Vector of animals</i>
--------	--

Description

animal - Generate a random vector of animals.

pet - Generate a random vector of pets.

Usage

```
animal(n, k = 10, x = wakefield::animal_list, prob = NULL,  
       name = "Animal")
```

```
pet(n, x = c("Dog", "Cat", "None", "Bird", "Horse"), prob = c(0.365, 0.304,  
0.258, 0.031, 0.015), name = "Pet")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
k	The number of the elements of <code>x</code> to sample from (uses <code>sample(x, k)</code>).
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The household pets and probabilities:

Dog	36.5 %
Cat	30.4 %
None	25.8 %
Bird	3.1 %
Horse	1.5 %

Value

Returns a random factor vector of animal elements.

See Also

Other variable.functions: [age](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#),

income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, race, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```
animal(10)
pie(table(animal(10000)))
```

```
pet(10)
pie(table(pet(10000)))
```

animal_list	<i>Animal List</i>
-------------	--------------------

Description

A dataset containing a character vector animals

Usage

```
data(animal_list)
```

Format

A character vector with 591 elements

References

<http://a-z-animals.com/animals>

answer	<i>Generate Random Vector of Answers (Yes/No)</i>
--------	---

Description

Generate a random vector of answers (yes/no).

Usage

```
answer(n, x = c("No", "Yes"), prob = NULL, name = "Answer")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of answers to sample from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of answers (yes/no) outcome elements.

See Also

Other variable.functions: [age](#), [animal](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
answer(10)
100*table(answer(n <- 10000))/n
```

area	<i>Generate Random Vector of Areas</i>
------	--

Description

Generate a random vector of areas ("Suburban", "Urban", "Rural").

Usage

```
area(n, x = c("Suburban", "Urban", "Rural"), prob = NULL, name = "Area")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of area status elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
area(10)
barplot(table(area(10000)))
```

as_integer

Convert a Factor Data Frame to Integer

Description

Converts a [data.frame](#) of [factors](#) to integers.

Usage

```
as_integer(x, cols = NULL, fun = as.integer)
```

Arguments

x	A data.frame of factors .
cols	Numeric indices of the columns to include (use - to exclude as well). Default is to assign random NAs to all columns except the first column.
fun	An as. coercion function to apply to each column. Default is as.integer .

Value

Returns a [data.frame](#) equal to the [class](#) of x with integer columns rather than factor.

See Also

[r_series](#)

Examples

```

as_integer(r_series(likert_7, 5, 10))
as_integer(r_series(likert_7, 5, 10), cols = c(2, 4))

library(dplyr)
r_data_frame(n=100,
  age,
  political,
  sex,
  grade
) %>%
  as_integer(2:3)

```

car	<i>Generate Random Vector of Cars</i>
-----	---------------------------------------

Description

Generate a random vector of cars (see [?mtcars](#)).

Usage

```
car(n, x = rownames(datasets::mtcars), prob = NULL, name = "Car")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of car elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
car(10)
table(car(10000))
```

children

Generate Random Vector of Number of Children

Description

Generate a random vector of number of children.

Usage

```
children(n, x = 0:10, prob = c(0.25, 0.25, 0.15, 0.15, 0.1, 0.02, 0.02,
  0.02, 0.02, 0.01, 0.01), name = "Children")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of number of children elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
children(10)
pie(table(children(100)))
```

coin	<i>Generate Random Vector of Coin Flips</i>
------	---

Description

Generate a random vector of coin flips (heads/tails).

Usage

```
coin(n, x = c("Tails", "Heads"), prob = NULL, name = "Coin")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of coin outcomes to sample from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of coin flip outcome elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
coin(10)
100*table(coin(n <- 10000))/n
```

color *Generate Random Vector of Colors*

Description

color - Generate a random vector of colors (sampled from colors()).

color - Generate a random vector of *psychological primary* colors (sampled from colors()).

Usage

```
color(n, k = 10, x = grDevices::colors(), prob = NULL, name = "Color")
```

```
primary(n, x = c("Red", "Green", "Blue", "Yellow", "Black", "White"),
        prob = NULL, name = "Color")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
k	The number of the elements of <code>x</code> to sample from (uses <code>sample(x, k)</code>).
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of color elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
color(10)
pie(tab <- table(color(10000)), col = names(tab))
```

```
primary(10)
pie(tab <- table(primary(10000)), col = names(tab))
barplot(tab <- table(primary(10000, prob = probs(6))), col = names(tab))
```

date_stamp	<i>Generate Random Vector of Dates</i>
------------	--

Description

Generate a random vector of dates.

Usage

```
date_stamp(n, random = FALSE, x = NULL, start = Sys.Date(), k = 12,  
  by = "-1 months", prob = NULL, name = "Date")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
random	logical. If TRUE the dates are randomized, otherwise the dates are sequential.
x	A vector of elements to chose from. This may be NULL if arguments are supplied to <code>start</code> , <code>k</code> , and <code>by</code> . The <code>x</code> argument takes precedence over the other three if <code>!is.null</code> . Note that <code>start</code> , <code>k</code> , and <code>by</code> work together to make a vector of dates to sample from. See seq.Date for additional information.
start	A date to start the sequence at.
k	The length of the sequence (number of the elements) so build out from <code>start</code> .
by	The interval to use in building the sequence.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of date elements.

See Also

[seq.Date](#)

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```

date_stamp(10)
pie(table(date_stamp(2000, prob = probs(12))))

## Supply dates to `x` to sample from
date_stamp(10, x = seq(as.Date("1980-11-16"), length = 30, by = "1 years"))

```

death

*Generate Random Vector of Deaths Outcomes***Description**

Generate a random logical vector of deaths (TRUE/FALSE).

Usage

```

death(n, prob = NULL, name = "Death")

died(n, prob = NULL, name = "Died")

```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random logical vector of death outcome elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```

death(10)
died(10)
100*table(death(n <- 10000))/n
100*table(death(n <- 10000, prob = c(.3, .7)))/n
r_data_frame(10, died)

```

`dice`*Generate Random Vector of Dice Throws*

Description

Generate a random vector of dice throws.

Usage

```
dice(n, x = 1:6, prob = NULL, name = "Dice")
```

Arguments

<code>n</code>	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
<code>x</code>	A vector of elements to chose from.
<code>prob</code>	A vector of probabilities to chose from.
<code>name</code>	The name to assign to the output vector's <code>vname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of dice throw elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
dice(10)
barplot(table(dice(10000)))
```

`dna`*Generate Random Vector of DNA Nucleobases*

Description

Generate a random vector of DNA nucleobases ("Guanine", "Adenine", "Thymine", "Cytosine").

Usage

```
dna(n, x = c("Guanine", "Adenine", "Thymine", "Cytosine"), prob = NULL,
    name = "DNA")
```

Arguments

<code>n</code>	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
<code>x</code>	A vector of elements to chose from.
<code>prob</code>	A vector of probabilities to chose from.
<code>name</code>	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of DNA nucleobase elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
dna(10)
barplot(table(dna(10000)))
```

`dob` *Generate Random Vector of Birth Dates*

Description

Generate a random vector of birth dates.

Usage

```
dob(n, random = TRUE, x = NULL, start = Sys.Date() - 365 * 15, k = 365 *  
  2, by = "1 days", prob = NULL, name = "DOB")
```

```
birth(n, random = TRUE, x = NULL, start = Sys.Date() - 365 * 15, k = 365  
  * 2, by = "1 days", prob = NULL, name = "Birth")
```

Arguments

<code>n</code>	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
<code>random</code>	logical. If TRUE the dates are randomized, otherwise the dates are sequential.
<code>x</code>	A vector of elements to chose from. This may be NULL if arguments are supplied to <code>start</code> , <code>k</code> , and <code>by</code> . The <code>x</code> argument takes precedence over the other three if <code>!is.null</code> . Note that <code>start</code> , <code>k</code> , and <code>by</code> work together to make a vector of dates to sample from. See seq.Date for additional information.
<code>start</code>	A date to start the sequence at.
<code>k</code>	The length of the sequence (number of the elements) so build out from <code>start</code> .
<code>by</code>	The interval to use in building the sequence.
<code>prob</code>	A vector of probabilities to chose from.
<code>name</code>	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of birth date elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
dob(10)
barplot(table(birth(15)))
barplot(table(birth(30)))
```

dummy

Generate Random Dummy Coded Vector

Description

Generate a random dummy coded (0/1) vector.

Usage

```
dummy(n, prob = NULL, name = "Dummy")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random dummy vector of (0/1) elements.

See Also

[sample.int](#)

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
dummy(100, name = "Var")
table(dummy(1000))
```

education

*Generate Random Vector of Educational Attainment Level***Description**

Generate a random vector of educational attainment level.

Usage

```
education(n, x = c("No Schooling Completed", "Nursery School to 8th Grade",
  "9th Grade to 12th Grade, No Diploma", "Regular High School Diploma",
  "GED or Alternative Credential", "Some College, Less than 1 Year",
  "Some College, 1 or More Years, No Degree", "Associate's Degree",
  "Bachelor's Degree", "Master's Degree", "Professional School Degree",
  "Doctorate Degree"), prob = c(0.013, 0.05, 0.085, 0.246, 0.039, 0.064, 0.15,
  0.075, 0.176, 0.072, 0.019, 0.012), name = "Education")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The educational attainments and probabilities used match approximate U.S. educational attainment make-up (<http://www.census.gov>):

Highest Attainment	Percent
No Schooling Completed	1.3 %
Nursery School to 8th Grade	5 %
9th Grade to 12th Grade, No Diploma	8.5 %
Regular High School Diploma	24.6 %
GED or Alternative Credential	3.9 %
Some College, Less than 1 Year	6.4 %
Some College, 1 or More Years, No Degree	15 %
Associate's Degree	7.5 %
Bachelor's Degree	17.6 %
Master's Degree	7.2 %
Professional School Degree	1.9 %
Doctorate Degree	1.2 %

Value

Returns a random vector of educational attainment level elements.

References

<http://www.census.gov>

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
education(10)
pie(table(education(10000)))
```

employment

Generate Random Vector of Employment Statuses

Description

Generate a random vector of employment statuses.

Usage

```
employment(n, x = c("Full Time", "Part Time", "Unemployed", "Retired",
  "Student"), prob = c(0.6, 0.1, 0.1, 0.1, 0.1), name = "Employment")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The following arbitrary probabilities are used:

Employment Status	Percent
Full Time	60%
Part Time	10%
Unemployed	10%
Retired	10%
Student	10%

Value

Returns a random vector of employment status elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
employment(10)
pie(table(employment(10000)))
barplot(table(employment(10000)))
```

eye

Generate Random Vector of Eye Colors

Description

Generate a random vector of eye colors.

Usage

```
eye(n, x = c("Brown", "Blue", "Green", "Hazel", "Gray"), prob = c(0.44, 0.3,
  0.13, 0.09, 0.04), name = "Eye")
```

Arguments

n The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.

x A vector of elements to chose from.

prob A vector of probabilities to chose from.

name The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Details

The eye colors and probabilities:

Color	Percent
Brown	44 %
Blue	30 %
Green	13 %
Hazel	9 %
Gray	4 %

Value

Returns a random vector of eye color elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
eye(10)
barplot(v <- table(eye(10000)), col = replace(names(v), 4, "yellowgreen"))
```

grade *Generate Random Vector of Grades*

Description

grade - Generate a random normal vector of percent grades.
 grade - Generate a random normal vector of letter grades.
 grade - Generate a random normal vector of grade point averages (GPA; 0.0 - 4.0 scale).

Usage

```
grade(n, mean = 88, sd = 4, name = "Grade", digits = 1)

grade_letter(n, mean = 88, sd = 4, name = "Grade_Letter")

gpa(n, mean = 88, sd = 4, name = "GPA")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
mean	The mean value for the normal distribution to be drawn from.
sd	The standard deviation of the normal distribution to draw from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .
digits	Integer indicating the number of decimal places to be used. Negative values are allowed (see round).

Details

The conversion between percent range, letter grade, and GPA is:

Percent	Letter	GPA
97-100	A+	4.00
93-96	A	4.00
90-92	A-	3.67
87-89	B+	3.33
83-86	B	3.00
80-82	B-	2.67
77-79	C+	2.33
73-76	C	2.00
70-72	C-	1.67
67-69	D+	1.33
63-66	D	1.00
60-62	D-	0.67
< 60	F	0.00

Value

Returns a random normal vector of grade elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
grade(10)
hist(grade(10000))
interval(grade, 5, n = 1000)
```

```
grade_letter(10)
barplot(table(grade_letter(10000)))

gpa(10)
hist(gpa(10000))
```

grade_level	<i>Generate Random Vector of Grade Levels</i>
-------------	---

Description

Generate a random vector of grade levels.

Usage

```
grade_level(n, x = c("K", "1", "2", "3", "4", "5", "6", "7", "8", "9", "10",
  "11", "12"), prob = NULL, name = "Grade_Level")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of grade level elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
grade_level(10)
barplot(table(grade_level(10000)))
```

grady_augmented	<i>Augmented List of Grady Ward's English Words and Mark Kantrowitz's Names List</i>
-----------------	--

Description

A dataset containing a vector of Grady Ward's English words augmented with **qdapDictionaries's** DICTIONARY, **Mark Kantrowitz's names list**, other proper nouns, and contractions.

Usage

```
data(grady_augmented)
```

Format

A character vector with 122806 elements

Details

A dataset containing a vector of Grady Ward's English words augmented with proper nouns (U.S. States, Countries, Mark Kantrowitz's Names List, and months) and contractions. That dataset is augmented to increase the data set size.

References

Moby Thesaurus List by Grady Ward <http://www.gutenberg.org>

List of names from Mark Kantrowitz <http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/corpora/names/>. A copy of the **README** is available [here](#) per the author's request.

group	<i>Generate Random Vector of Control/Treatment Groups</i>
-------	---

Description

Generate a random vector of binary groups (e.g., control/treatment).

Usage

```
group(n, x = c("Control", "Treatment"), prob = NULL, name = "Group")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of groups to sample from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of group (control/treatment) elements.

Note

If you want > 2 groups see `'r_sample_factor'`.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
group(10)
100*table(group(n <- 10000))/n
100*table(group(n <- 10000, prob = c(.3, .7)))/n
```

hair

Generate Random Vector of Hair Colors

Description

Generate a random vector of hair colors.

Usage

```
hair(n, x = c("Brown", "Black", "Blonde", "Red"), prob = c(0.35, 0.28, 0.26,
  0.11), name = "hair")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The hair colors and probabilities:

Color	Percent
Brown	35 %
Black	28 %
Blonde	26 %
Red	11 %

Value

Returns a random vector of hair color elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
hair(10)
v <- table(hair(10000))
lbs <- paste0(names(v), "\n", round(100*v/sum(v), 1), "%")
pie(v, col = replace(names(v), 3, "yellow"), labels = lbs)
```

height

Generate Random Vector of Heights

Description

`height` and `height_in` - Generate a random normal vector of heights in inches.

`height_cm` - Generate a random normal vector of heights in centimeters.

Usage

```
height(n, mean = 69, sd = 3.75, min = 1, max = NULL, digits = 0,  
       name = "Height")
```

```
height_in(n, mean = 69, sd = 3.75, min = 1, max = NULL, digits = 1,  
          name = "Height(in)")
```

```
height_cm(n, mean = 175.26, sd = 9.525, min = 1, max = NULL,  
          digits = 1, name = "Height(cm)")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
mean	The mean value for the normal distribution to be drawn from.
sd	The standard deviation of the normal distribution to draw from.
min	A numeric lower boundary cutoff. Results less than this value will be replaced with min.
max	A numeric upper boundary cutoff. Results greater than this value will be replaced with max.
digits	Integer indicating the number of decimal places to be used. Negative values are allowed (see round).
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random normal vector of height elements.

Note

`height` rounds to nearest whole number. `height_in` & `height_cm` round to the nearest tenths.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
height(10)  
hist(height(10000))  
interval(height, 5, n = 1000)
```

hour	<i>Generate a Random Sequence of H:M:S Times</i>
------	--

Description

Generate a random vector of H:M:S times.

Usage

```
hour(n, x = seq(0, 23.5, by = 0.5), prob = NULL, random = FALSE,  
     name = "Hour")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
random	logical. If TRUE the times are randomized, otherwise the times are sequential.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of H:M:S time elements.

See Also

[times](#)

Examples

```
hour(20)  
hour(20, random=TRUE)
```

id	<i>Identification Numbers</i>
----	-------------------------------

Description

id - Generate a sequential [character](#) vector of zero-padded identification numbers (IDs).

id_factor - Generate a sequential [factor](#) vector of zero-padded identification numbers (IDs).

Usage

```
id(n, random = FALSE, name = "ID")
```

```
id_factor(n, random = FALSE, name = "ID")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
random	logical. If TRUE the IDs are randomized, otherwise the IDs are sequential.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a (optionally random) vector of [character/factor](#) observations ID numbers.

Warning

id uses [sprintf](#) to generate the padded ID. Per [sprintf](#)'s documentation: "The format string is passed down the OS's `sprintf` function...The behaviour on inputs not documented here is 'undefined', which means it is allowed to differ by platform." See [sprintf](#) for details.

Note

id is faster than `id_factor`, as the later coerces the vector to a [factor](#).

See Also

[sprintf](#)

Examples

```
id(1000)
r_data_frame(n=21, id)
```

income	<i>Generate Random Gamma Vector of Incomes</i>
--------	--

Description

Generate a random gamma vector of incomes.

Usage

```
income(n, digits = 2, name = "Income")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
digits	Integer indicating the number of decimal places to be used. Negative values are allowed (see round).
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

Incomes are generated using: $\text{rgamma}(n, 2) * 2000$.

Value

Returns a random gamma vector of income elements.

See Also

[gamma](#)

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
income(10)
hist(income(10000))
pie(table(cut(income(10000), 10)))
```

internet_browser *Generate Random Vector of Internet Browsers*

Description

Generate a random vector of Internet browser.

Usage

```
internet_browser(n, x = c("Chrome", "IE", "Firefox", "Safari", "Opera",
  "Android"), prob = c(0.5027, 0.175, 0.1689, 0.0994, 0.017, 0.0132),
  name = "Browser")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The browser use and probabilities (from <http://gs.statcounter.com>):

Browser	Percent
Chrome	50.27 %
IE	17.50 %
Firefox	16.89 %
Safari	9.94 %
Opera	1.70 %
Android	1.32 %

Value

Returns a random factor vector of Internet browser elements.

References

<http://www.pewforum.org/2012/12/18/table-religious-composition-by-country-in-numbers>

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
internet_browser(20)
barplot(table(internet_browser(10000)))
pie(table(internet_browser(10000)))
```

interval	<i>Cut Numeric Into Factor</i>
----------	--------------------------------

Description

A wrapper for [cut](#) that cuts the vector and then adds the varname produced by the original function.

Usage

```
interval(fun, breaks, ..., labels = NULL, include.lowest = FALSE,
         right = TRUE, dig.lab = 3, ordered_result = FALSE, n)
```

Arguments

<code>fun</code>	A vector producing function.
<code>breaks</code>	Either a numeric vector of two or more unique cut points or a single number (greater than or equal to 2) giving the number of intervals into which the vector produced from <code>fun</code> is to be cut.
<code>labels</code>	Labels for the levels of the resulting category. By default, labels are constructed using "(a,b]" interval notation. If <code>labels = FALSE</code> , simple integer codes are returned instead of a factor.
<code>include.lowest</code>	logical. If TRUE an 'x[i]' equal to the lowest (or highest, for <code>right = FALSE</code>) 'breaks' value should be included.
<code>right</code>	logical. If TRUE the intervals will be closed on the right (and open on the left).
<code>dig.lab</code>	An integer which is used when labels are not given. It determines the number of digits used in formatting the break numbers.
<code>ordered_result</code>	logical. If TRUE the result be an ordered factor.
<code>n</code>	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
<code>...</code>	Other arguments passed to <code>fun</code> .

Value

Returns a `cut` factor vector.

See Also

`cut`

Examples

```
interval(normal, 4, n=100)
attributes(interval(normal, 4, n=100))
interval(age, 3, n = 1000)
```

 iq

Generate Random Vector of Intelligence Quotients (IQs)

Description

Generate a random normal vector of intelligence quotients (IQs).

Usage

```
iq(n, mean = 100, sd = 10, min = 0, max = NULL, digits = 0,
   name = "IQ")
```

Arguments

<code>n</code>	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
<code>mean</code>	The mean value for the normal distribution to be drawn from.
<code>sd</code>	The standard deviation of the normal distribution to draw from.
<code>min</code>	A numeric lower boundary cutoff. Results less than this value will be replaced with <code>min</code> .
<code>max</code>	A numeric upper boundary cutoff. Results greater than this value will be replaced with <code>max</code> .
<code>digits</code>	Integer indicating the number of decimal places to be used. Negative values are allowed (see <code>round</code>).
<code>name</code>	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random normal vector of IQ elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
iq(10)
hist(iq(10000))
interval(iq, 5, n = 1000)
```

language

Generate Random Vector of Languages

Description

Generate a random vector of languages from the [presidential_debates_2012](#).

Usage

```
language(n, x = wakefield::languages[["Language"]],
         prob = wakefield::languages[["Proportion"]], name = "Language")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random character vector of language elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
language(10)
pie(table(language(10000)))

lang <- wakefield::languages[sample(1:99, 6), ]
lang["prop"] <- lang[["N"]]/sum(lang[["N"]])
labs <- round(100 * lang[["prop"]], 1)
pie(lang[["prop"]], paste0(lang[["Language"]], "\n", labs, "%"))
```

languages

Languages of the World

Description

A dataset containing native language use statistics taken from: http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers.

Usage

```
data(languages)
```

Format

A data frame with 99 rows and 4 variables

Details

- Language. The language spoken.
- N. The number of speakers world-wide.
- Proportion. The proportion of speakers.
- Percent. The percentage of speakers.

References

http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers

level	<i>Generate Random Vector of Levels</i>
-------	---

Description

level - Generate a random vector of integer levels (1-4).

math - Generate a random vector of integer mathematics levels (1-4) similar to New York State grades 3-8 assessment results.

ela - Generate a random vector of integer English language arts (ELA) levels (1-4) similar to New York State grades 3-8 assessment results.

Usage

```
level(n, x = 1:4, prob = NULL, name = "Level")
```

```
math(n, x = 1:4, prob = c(0.29829, 0.33332, 0.22797, 0.14042),
     name = "Math")
```

```
ela(n, x = 1:4, prob = c(0.3161, 0.37257, 0.2233, 0.08803), name = "ELA")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

Distribution of levels (used in `prob`) were taken from New York State' s 2014 assessment report: <http://www.p12.nysed.gov/irs/>

Level	ELA	Math
1	31.6%	29.8%
2	37.3%	33.3%
3	22.3%	22.8%
4	8.8%	14.0%

Value

Returns a random vector of integer levels (1-4) elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
level(10)
barplot(table(level(10000, prob = probs(4))))
```

```
math(10)
barplot(table(math(10000)))
```

```
ela(10)
barplot(table(ela(10000)))
```

likert

Generate Random Vector of Likert-Type Responses

Description

Generate a random vector of Likert-type responses.

Usage

```
likert(n, x = c("Strongly Agree", "Agree", "Neutral", "Disagree",
  "Strongly Disagree"), prob = NULL, name = "Likert")
```

```
likert_5(n, x = c("Strongly Agree", "Agree", "Neutral", "Disagree",
  "Strongly Disagree"), prob = NULL, name = "Likert")
```

```
likert_7(n, x = c("Strongly Agree", "Agree", "Somewhat Agree", "Neutral",
  "Somewhat Disagree", "Disagree", "Strongly Disagree"), prob = NULL,
  name = "Likert")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of Likert-type response elements.

Note

likert & likert_5 are identical outputs, sampling from a 5-point response scale. likert_7 samples from a 7-point response scale.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
dice(10)
barplot(table(dice(10000)))
```

lorem_ipsum

Generate Random Lorem Ipsum Strings

Description

Generates (pseudo)random *lorem ipsum* text.

Usage

```
lorem_ipsum(n, ..., name = "Lorem_Ipsum")
paragraph(n, ..., name = "Paragraph")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .
...	Other arguments passed to stri_rand_lipsum .

Value

Returns a random character vector of string elements.

Note

lorem_ipsum and paragraph produce identical strings but will produce different vector/column names when used inside of r_data_frame or r_list.

See Also

[stri_rand_lipsum](#)

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
lorem_ipsum(10)
paragraph(10)

lorem_ipsum(10, start_lipsum = FALSE)
```

marital

Generate Random Vector of Marital Statuses

Description

Generate a random vector of marital statuses.

Usage

```
marital(n, x = c("Married", "Divorced", "Widowed", "Separated",
  "Never Married"), prob = NULL, name = "Marital")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random vector of marital status elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
marital(10)
barplot(table(marital(10000)))
```

military

Generate Random Vector of Military Branches

Description

Generate a random vector of military branches.

Usage

```
military(n, x = c("Army", "Air Force", "Navy", "Marine Corps", "Coast Guard"),
  prob = c(0.3785, 0.2334, 0.2218, 0.1366, 0.0296), name = "Military")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The military branches and probabilities used match approximate U.S. military make-up:

Branch	N	Percent
Army	541,291	37.9%
Air Force	333,772	23.3%
Navy	317,237	22.2%
Marine Corps	195,338	13.7%
Coast Guard	42,357	3.0%

Value

Returns a random factor vector of military branch elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
military(10)
barplot(table(military(10000)))
pie(table(military(10000)))
```

minute

Generate a Random Sequence of Minutes in H:M:S Format

Description

Generate a random vector of minutes in H:M:S format.

Usage

```
minute(n, x = seq(0, 59, by = 1)/60, prob = NULL, random = FALSE,
       name = "Minute")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
random	logical. If TRUE the times are randomized, otherwise the times are sequential.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of minute time elements in H:M:S format.

See Also

[times](#)

Examples

```
minute(20)
minute(20, random=TRUE)
pie(table(minute(2000, x = seq(0, 59, by = 10)/60, prob = probs(6))))
```

month

Generate Random Vector of Months

Description

Generate a random factor vector of months.

Usage

```
month(n, x = month.name, prob = NULL, name = "Month")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random character vector of month elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
month(10)
pie(table(month(10000, prob = probs(12))))
```

name	<i>Generate Random Vector of Names</i>
------	--

Description

Generate a random vector of first names (Gender Neutral-ish).

Usage

```
name(n, x = wakefield::name_neutral, prob = NULL, replace = FALSE,  
     name = "Name")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
replace	logical. If TRUE sampling is done with replacement. Default is without replacement.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of name elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
name(10)  
name(100)  
name(1000, replace = TRUE)
```

name_neutral	<i>Gender Neutral Names</i>
--------------	-----------------------------

Description

A dataset containing a character vector gender neutral names according to the U.S. Census.

Usage

```
data(name_neutral)
```

Format

A character vector with 662 elements

References

<http://www.census.gov>

normal	<i>Generate Random Normal Vector</i>
--------	--------------------------------------

Description

normal - A wrapper for `rnorm` that generate a random normal vector.

normal_round - A wrapper for `rnorm` that generate a rounded random normal vector.

Usage

```
normal(n, mean = 0, sd = 1, min = NULL, max = NULL, name = "Normal")
```

```
normal_round(n, mean = 0, sd = 1, min = NULL, max = NULL, digits = 2,
  name = "Normal")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
mean	The mean value for the normal distribution to be drawn from.
sd	The standard deviation of the normal distribution to draw from.
min	A numeric lower boundary cutoff. Results less than this value will be replaced with min.
max	A numeric upper boundary cutoff. Results greater than this value will be replaced with max.

name	The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .
digits	Integer indicating the number of decimal places to be used. Negative values are allowed (see round).

Value

Returns a random vector of elements.

See Also

[rnorm](#)

[round](#)

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
normal(100, name = "Var")
hist(normal(10000, 100, 10))
interval(normal, 9, n = 1000)
```

peek

Data Frame Viewing

Description

Convenience function to view all the columns of the head of a truncated `data.frame`. `peek` invisibly returns `x`. This makes its use ideal in a **dplyr/magrittr** pipeline.

Usage

```
peek(x, n = 10, width = 10, ...)
```

Arguments

x	A <code>data.frame</code> object.
n	Number of rows to display.
width	The width of the columns to be displayed.
...	For internal use.

Details

By default **dplyr** does not print all columns of a data frame (`tbl_df`). This makes inspection of data difficult at times, particularly with text string data. `peek` allows the user to see a truncated head for inspection purposes.

Value

Prints a truncated head but invisibly returns `x`.

See Also

[head](#)

Examples

```
(dat1 <- r_data_frame(100, id, sentence, paragraph))
peek(dat1)
peek(dat1, n = 20)
peek(dat1, width = 40)

library(dplyr)

## Use in a dplyr/magrittr pipeline to view the data (silly example)
par(mfrow = c(2, 2))

r_data_frame(1000, id, sex, pet, employment, eye, sentence, paragraph) %>%
  peek %>%
  (function(x, ind = 2:5){ invisible(lapply(ind, function(i) pie(table(x[[i]]))))})

## A wider data set example
dat2 <- r_data_theme()

dat2
peek(dat2)
```

plot.tbl_df

Plots a tbl_df Object

Description

Plots a `tbl_df` object.

Usage

```
## S3 method for class 'tbl_df'
plot(x, ...)
```

Arguments

x The tbl_df object.
 ... Arguments passed to `table_heat`.

 political

Generate Random Vector of Political Parties

Description

Generate a random vector of political parties.

Usage

```
political(n, x = c("Democrat", "Republican", "Constitution", "Libertarian",
  "Green"), prob = c(0.577269133302094, 0.410800432748879,
  0.00491084954793489, 0.00372590303330866, 0.0032936813677832),
  name = "Political")
```

Arguments

n The number elements to generate. This can be globally set within the environ-
 ment of `r_data_frame` or `r_list`.
 x A vector of elements to chose from.
 prob A vector of probabilities to chose from.
 name The name to assign to the output vector's `varname` attribute. This is used to auto
 assign names to the column/vector name when used inside of `r_data_frame` or
`r_list`.

Details

The political parties and probabilities used match approximate U.S. political make-up of registered voters (2014). The default make up is:

Party	N	Percent
Democrat	43,140,758	57.73%
Republican	30,700,138	41.08%
Constitution	367,000	.49%
Libertarian	278,446	.37%
Green	246,145	.33%

Value

Returns a random factor vector of political party elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
political(10)
barplot(table(political(10000)))
```

```
presidential_debates_2012
      2012 U.S. Presidential Debate Dialogue
```

Description

A dataset containing 2911 ordered sentences used by speakers during the three 2012 presidential debates.

Usage

```
data(presidential_debates_2012)
```

Format

A character vector with 2911 elements

```
print.available      Prints an available Object.
```

Description

Prints an available object.

Usage

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

Arguments

x	The available object
...	ignored

<code>print.variable</code>	<i>Prints a variable Object</i>
-----------------------------	---------------------------------

Description

Prints a variable object

Usage

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

Arguments

<code>x</code>	The variable object.
<code>...</code>	Ignored.

<code>probs</code>	<i>Generate a Random Vector of Probabilities.</i>
--------------------	---

Description

Generate a random vector of probabilities that sum to 1.

Usage

```
probs(j, upper = 1e+06)
```

Arguments

<code>j</code>	An integer of number of probability elements (typically performs best at $j < 4000$).
<code>upper</code>	<code>probs</code> works by sampling from <code>1 : upper</code> <code>j</code> times and then dividing each sample by the sum of all samples.

Value

Returns a vector of probabilities summing to 1.

Examples

```
probs(10)  
sum(probs(100))  
pie(table(month(10000, prob = probs(12))))
```

race *Generate Random Vector of Races*

Description

Generate a random vector of races.

Usage

```
race(n, x = c("White", "Hispanic", "Black", "Asian", "Bi-Racial", "Native",
             "Other", "Hawaiian"), prob = c(0.637, 0.163, 0.122, 0.047, 0.019, 0.007,
             0.002, 0.0015), name = "Race")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The races and probabilities used match approximate U.S. racial make-up. The default make up is:

Race	Percent
White	63.70 %
Hispanic	16.30 %
Black	12.20 %
Asian	4.70 %
Bi-Racial	1.90 %
Native	.70 %
Other	.20 %
Hawaiian	.15 %

Value

Returns a random factor vector of elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#),

height, income, internet_browser, iq, language, level, likert, lorem_ipsum, marital, military, month, name, normal, political, religion, sat, sentence, sex_inclusive, sex, smokes, speed, state, string, upper, valid, year, zip_code

Examples

```
race(10)
100*table(race(n <- 10000))/n
```

relate	<i>Create Related Numeric Columns</i>
--------	---------------------------------------

Description

Generate columns that are related.

Usage

```
relate(x, j, name = NULL, operation = "+", mean = 5, sd = 1,
  rep.sep = "_", digits = max(nchar(sub("^[^.]*. ", "", x))))
```

Arguments

x	A starting column.
j	The number of columns to produce.
name	An optional prefix name to give to the columns. If NULL attempts to take from the varname attribute of x. If not found, "Variable" is used.
operation	A operation character vector of length 1; either c("+", "-", "*", "/"). This is the relationship between columns.
mean	Mean is the average value to add, subtract, multiple, or divide by.
sd	The amount of variability to allow in mean. Setting to 0 will constrain the operation between $x_{(n - 1)}$ column and x_n to be exactly the mean value (see Examples for a demonstration).
rep.sep	A separator to use for repeated variable names. For example if the <code>age</code> is used three times (<code>r_data_frame(age, age, age)</code>), the name "Age" will be assigned to all three columns. The results in column names c("Age_1", "Age_2", "Age_3").
digits	The number of digits to round to. Defaults to the max number of significant digits in x.

Value

Returns a `tbl_df`.

See Also

[r_series](#)

Examples

```

relate(1:10, 10)

(x <- r_data_frame(10, id, relate(1:10, 10, "Time", mean = 2)))
library(ggplot2)

dat <- with(x, data.frame(ID = rep(ID, ncol(x[, -1])), stack(x[, -1])))
dat[["Time"]] <- factor(sub("Time_", "", dat[["ind"]]), levels = 1:10)
ggplot(dat, aes(x = Time, y = values, color = ID, group = ID)) +
  geom_line(size=.8)

relate(1:10, 10, name = "X", operation = "-")
relate(1:10, 10, "X", mean = 1, sd = 0)
relate(1:10, 10, "Var", "*")
relate(1:10, 10, "Var", "/")

relate(gpa(30), 5, mean = .1)
relate(likert(10), 5, mean = .1, sd = .2)
relate(date_stamp(10), 6)
relate(time_stamp(10), 6)
relate(rep(100, 10), 6, "Reaction", "-")

```

religion

*Generate Random Vector of Religions***Description**

Generate a random vector of religion.

Usage

```

religion(n, x = c("Christian", "Muslim", "None", "Hindu", "Buddhist", "Folk",
  "Other", "Jewish"), prob = c(0.31477, 0.23163, 0.16323, 0.14985, 0.07083,
  0.05882, 0.00859, 0.00227), name = "Religion")

```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The religion and probabilities used match approximate world religion make-up (from [Pew Research Center](#)). The default make up is:

Religion	N	Percent
Christian	2,173,260,000	31.48 %
Muslim	1,599,280,000	23.16 %
None	1,127,000,000	16.32 %
Hindu	1,034,620,000	14.99 %
Buddhist	489,030,000	7.08 %
Folk	406,140,000	5.88 %
Other	59,330,000	.86 %
Jewish	15,670,000	.23 %

Value

Returns a random factor vector of religion elements.

References

<http://www.pewforum.org/2012/12/18/table-religious-composition-by-country-in-numbers>

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
religion(10)
barplot(table(religion(10000)))
pie(table(religion(10000)))
```

r_data

Pre-Selected Column Data Set

Description

r_data - Generate a data set with pre-set columns selected.

r_data_theme - Generate a themed data set with pre-set columns.

Usage

```
r_data(n = 500, ...)
```

```
r_data_theme(n = 100, data_theme = "the_works")
```

Arguments

n	The length to pass to the randomly generated vectors (number of rows).
data_theme	A data theme. Currently selections include: the_works all available variable functions survey ID column plus 10 numeric 5-point Likert type response columns survey2 ID column plus 10 5-point Likert type response columns
...	A set of optionally named arguments. Using wakefield variable functions require no name or call parenthesis.

Details

The pre-selected columns include:

- ID
- Race
- Age
- Sex
- Hour
- IQ
- Height
- Died

The user may use ... to add additional columns. r_data is a convenience function to quickly produce a data set. For more specific usage use the more flexible [r_data_frame](#) function.

Value

Returns a [tbl_df](#).

See Also

[r_data_frame](#)

Examples

```
r_data()
r_data(10)
r_data(10, paragraph, Attending = valid)

peek(r_data_theme())
plot(r_data_theme(), flip=TRUE)

r_data_theme("survey")
r_data_theme("survey2")
```

r_data_frame

*Data Frame Production (From Variable Functions)***Description**

Produce a `tbl_df` data frame that allows the user to lazily pass unnamed **wakefield** variable functions (optionally, without call parenthesis).

Usage

```
r_data_frame(n, ..., rep.sep = "_")
```

Arguments

n	The length to pass to the randomly generated vectors.
rep.sep	A separator to use for repeated variable names. For example if the <code>age</code> is used three times (<code>r_data_frame(age, age, age)</code>), the name "Age" will be assigned to all three columns. The results in column names <code>c("Age_1", "Age_2", "Age_3")</code> . To turn of this behavior use <code>rep.sep = NULL</code> . This results in <code>c("Age", "Age.1", "Age.2")</code> column names in the <code>data.frame</code> .
...	A set of optionally named arguments. Using wakefield variable functions require no name or call parenthesis.

Value

Returns a `tbl_df`.

Author(s)

Josh O'Brien and Tyler Rinker <tyler.rinker@gmail.com>.

References

<http://stackoverflow.com/a/29617983/1000343>

See Also

`r_list`, `r_series` `r_dummy`

Examples

```
r_data_frame(n = 30,
             id,
             race,
             age,
             sex,
             hour,
             iq,
```



```
    height,
    died,
    Scoring = rnorm,
    Smoker = valid
  )

r_data_frame(n = 30,
  id,
  race,
  age(x = 8:14),
  Gender = sex,
  Time = hour,
  iq,
  grade, grade, grade, #repeated measures
  height(mean=50, sd = 10),
  died,
  Scoring = rnorm,
  Smoker = valid
)

r_data_frame(n = 500,
  id,
  age, age, age,
  grade, grade, grade
)

## Repeated Measures/Time Series
r_data_frame(n=100,
  id,
  age,
  sex,
  r_series(likert, 3),
  r_series(likert, 4, name = "Item", integer = TRUE)
)

## Expanded Dummy Coded Variables
r_data_frame(n=100,
  id,
  age,
  r_dummy(sex, prefix=TRUE),
  r_dummy(political)
)

## `peek` to view all columns
## `plot` (`table_heat`) for a graphic representation
library(dplyr)
r_data_frame(n=100,
  id,
  dob,
  animal,
  grade, grade,
  death,
  dummy,
```

```

    grade_letter,
    gender,
    paragraph,
    sentence
  ) %>%
  r_na() %>%
  peek %>%
  plot(palette = "Set1")

```

r_dummy

Title

Description

Description

Usage

```
r_dummy(fun, n, ..., prefix = FALSE, rep.sep = "_")
```

Arguments

fun	A wakefield variable function.
n	The number of rows to produce.
prefix	logical. If TRUE the original factor name (supplied to fun as name argument) will prefix the column names that were generated from the factor's categories.
rep.sep	A separator to use for the variable and category part of names when prefix = TRUE. For example if the <code>age</code> is used (<code>r_dummy(sex)</code>), this results in column names <code>c("Sex_Male", "Sex_Female")</code> .
...	Additional arguments passed to fun.

Value

Returns a `tbl_df`.

See Also

[r_list](#), [r_data_frame](#), [r_series](#)

Examples

```

r_dummy(sex, 10)
r_dummy(race, 1000)
r_dummy(race, 1000, name = "Ethnicity")

```

r_insert	<i>Insert Data Frames Into r_data_frame</i>
----------	---

Description

Safely insert `data.frame` objects into a `r_data_frame` or `r_list`.

Usage

```
r_insert(x, name = "Inserted")
```

Arguments

x	A <code>data.frame</code> to add a <code>seriesname</code> attribute (i.e., <code>attributes(x)[["seriesname"]]</code>)
name	A name to assign to <code>attributes(x)[["seriesname"]]</code> .

Value

Returns a `data.frame` with a `attributes(x)[["seriesname"]]` assigned.

See Also

[seriesname](#)

Examples

```
dat <- dplyr::data_frame(
  Age_1 = age(100), Age_2 = age(100), Age_3 = age(100),
  Smokes = smokes(n=100),
  Sick = ifelse(Smokes, sample(5:10, 100, TRUE), sample(0:4, 100, TRUE)),
  Death = ifelse(Smokes, sample(0:1, 100, TRUE, prob = c(.2, .8)),
    sample(0:1, 100, TRUE, prob = c(.7, .3)))
)

r_data_frame(100,
  id,
  r_insert(dat)
)

r_list(10,
  id,
  r_insert(dat)
)
```

`r_list`*List Production (From Variable Functions)*

Description

Produce a named `list` that allows the user to lazily pass unnamed **wakefield** variable functions (optionally, without call parenthesis).

Usage

```
r_list(n, ..., rep.sep = "_")
```

Arguments

<code>n</code>	The length to pass to the randomly generated vectors.
<code>rep.sep</code>	A separator to use for repeated variable names. For example if the <code>age</code> is used three times (<code>r_list(age, age, age)</code>), the name "Age" will be assigned to all three vectors in the list. The results in column names <code>c("Age_1", "Age_2", "Age_3")</code> . To turn of this behavior use <code>rep.sep = NULL</code> . This results in <code>c("Age", "Age", "Age")</code> for vector names, leading to <code>c("Age", "Age.1", "Age.2")</code> if coerced to a <code>data.frame</code> .
<code>...</code>	A set of optionally named arguments. Using wakefield variable functions require no name or call parenthesis.

Value

Returns a named list of equal length vectors.

Author(s)

Josh O'Brien and Tyler Rinker <tyler.rinker@gmail.com>.

References

<http://stackoverflow.com/a/29617983/1000343>

See Also

[r_data_frame](#), [r_series](#) [r_dummy](#)

Examples

```
r_list(  
  n = 30,  
  id,  
  race,  
  age,  
  sex,
```

```

    hour,
    iq,
    height,
    died,
    Scoring = rnorm
  )

  r_list(
    n = 30,
    id,
    race,
    age(x = 8:14),
    Gender = sex,
    Time = hour,
    iq,
    height(mean=50, sd = 10),
    died,
    Scoring = rnorm
  )

```

r_na

Title

Description

Description

Usage

```
r_na(x, cols = -1, prob = 0.05)
```

Arguments

x	A data.frame or list to randomly replace elements with NAs.
cols	Numeric indices of the columns to include (use - to exclude as well). Default is to assign random NAs to all columns except the first column.
prob	The proportion of each column/vector elements to assign to NA.

Value

Returns a [data.frame](#) or [list](#) with random missing values.

Examples

```

r_na(mtcars)
r_na(mtcars, NULL)

```

```
library(dplyr)
```

```
r_data_frame(  
  n = 30,  
  id,  
  race,  
  age,  
  sex,  
  hour,  
  iq,  
  height,  
  died,  
  Scoring = rnorm,  
  Smoker = valid  
) %>%  
  r_na(prob=.4)
```

r_sample

Generate Random Vector

Description

Generate a random vector.

Usage

```
r_sample(n, x = 1:100, prob = NULL, name = "Sample")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of elements.

See Also

[sample](#)

Examples

```
r_sample(100, name = "Var")
table(r_sample(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))
r_sample(x = c("B", "W"), prob = c(.7, .3), n = 25, name = "Race")
r_sample(25, x = c(TRUE, FALSE))
```

r_sample_binary	<i>Generate Random Binary Vector</i>
-----------------	--------------------------------------

Description

r_sample_binary - Generate a random binary vector.

r_sample_binary_factor - Generate a random binary vector and coerces to a factor.

Usage

```
r_sample_binary(n, x = 1:2, prob = NULL, name = "Binary")
```

```
r_sample_binary_factor(n, x = 1:2, prob = NULL, name = "Binary")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.
x	A vector of length 2 to sample from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of r_data_frame or r_list.

Value

Returns a random binary vector of elements.

See Also

[sample.int](#)

Examples

```
r_sample_binary(100, name = "Var")
table(r_sample_binary(1000))
c("B", "W")[r_sample_binary(10)]
```

r_sample_factor	<i>Generate Random Factor Vector</i>
-----------------	--------------------------------------

Description

Generate a random vector and coerces to a factor.

Usage

```
r_sample_factor(n, x = LETTERS, prob = NULL, name = "Factor")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random actor vector of elements.

See Also

[sample](#)

Examples

```
r_sample_factor(100, name = "Var")
table(r_sample_factor(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))
r_sample_factor(x = c("B", "W"), prob = c(.7, .3), n = 25)
```

r_sample_integer	<i>Generate Random Integer Vector</i>
------------------	---------------------------------------

Description

Generate a random integer vector.

Usage

```
r_sample_integer(n, x = 1:100, prob = NULL, name = "Integer")
```


Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random integer vector of elements.

See Also

[sample](#)

Examples

```
r_sample_integer(100, name = "Var")
table(r_sample_integer(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))
r_sample_integer(x = c("B", "W"), prob = c(.7, .3), n = 25, name = "Race")
r_sample_integer(25, x = c(TRUE, FALSE))
```

r_sample_logical	<i>Generate Random Logical Vector</i>
------------------	---------------------------------------

Description

Generate a random logical (TRUE/FALSE) vector.

Usage

```
r_sample_logical(n, prob = NULL, name = "Logical")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random logical (TRUE/FALSE) vector of elements.

See Also[sample](#)**Examples**

```
r_sample_logical(100, name = "Var")
table(r_sample_logical(1000))
c("B", "W")[r_sample_logical(10)]
```

r_sample_ordered	<i>Generate Random Ordered Factor Vector</i>
------------------	--

Description

Generate a random vector and coerces to an ordered factor.

Usage

```
r_sample_ordered(n, x = LETTERS[1:5], prob = NULL, name = "Ordered")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random factor vector of elements.

See Also[sample](#), [ordered](#)**Examples**

```
r_sample_ordered(100, name = "Var")

lvls <- c("Strongly Agree", "Agree", "Neutral", "Disagree", "Strongly Disagree")
table(r_sample_ordered(x = lvls, n=1000))

(out <- r_sample_ordered(x = c("Black", "Grey", "White"),
  prob = c(.5, .2, .3), n = 100))
slices <- c(table(out))
pie(slices, main="Pie Chart of Colors", col = tolower(names(slices)))
```

r_sample_replace	<i>Generate Random Vector (Without Replacement)</i>
------------------	---

Description

Generate a random vector without replacement.

Usage

```
r_sample_replace(n, x = 1:100, prob = NULL, replace = FALSE,  
  name = "Sample")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
replace	logical. If TRUE sampling is done with replacement. Default is without replacement.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of elements.

See Also

[sample](#)

Examples

```
r_sample(100, name = "Var")  
table(r_sample(x = c("Dog", "Cat", "Fish", "Bird"), n=1000))  
r_sample(x = c("B", "W"), prob = c(.7, .3), n = 25, name = "Race")  
r_sample(25, x = c(TRUE, FALSE))
```

r_series

*Data Frame Series (Repeated Measures)***Description**

Produce a `tbl_df` data frame of repeated measures from a wakefield variable function.

Usage

```
r_series(fun, j, n, ..., integer = FALSE, relate = NULL, rep.sep = "_")
```

Arguments

fun	A wakefield variable function.
j	The number of columns to produce.
n	The number of rows to produce.
integer	logical. If TRUE factor columns will be coerced to integer.
relate	Allows the user to specify the relationship between columns. May be a named list of <code>c("operation", "mean", "sd")</code> or a string of the form of "fM_sd" where 'f' is one of (+, -, *, /), 'M' is a mean value, and 'sd' is a standard deviation of the mean value (e.g., "*4_1"). See relate for details.
rep.sep	A separator to use for repeated variable names. For example if the age is used three times (<code>r_data_frame(age, age, age)</code>), the name "Age" will be assigned to all three columns. The results in column names <code>c("Age_1", "Age_2", "Age_3")</code> .
...	Additional arguments passed to fun.

Value

Returns a `tbl_df`.

References

<https://github.com/trinker/wakefield/issues/1/#issuecomment-96166910>

See Also

[r_list](#), [r_data_frame](#) [r_dummy](#)

Examples

```
r_series(grade, 5, 10)

## Custom name prefix
r_series(likert, 5, 10, name = "Question")

## Convert factors to integers
```

```

r_series(likert_7, 5, 10, integer = TRUE)

## Related variables
r_series(likert, 10, 200, relate = list(operation = "*", mean = 2, sd = 1))
r_series(likert, 10, 200, relate = "--3_1")
r_series(age, 10, 200, relate = "+5_0")

## Change sd to reduce/increase correlation
round(cor(r_series(grade, 10, 10, relate = "+1_2")), 2)
round(cor(r_series(grade, 10, 10, relate = "+1_0")), 2)
round(cor(r_series(grade, 10, 10, relate = "+1_.5")), 2)
round(cor(r_series(grade, 10, 10, relate = "+1_20")), 2)

## Plot Example 1
library(dplyr); library(ggplot2)

dat <- r_data_frame(12,
  name,
  r_series(likert, 10, relate = "+1_.5")
)

# Suggested use of tidyr or reshape2 package here instead
dat <- data.frame(
  ID = rep(dat[[1]], ncol(dat[-1])),
  stack(dat[-1])
)

dat[["Time"]] <- factor(sub("Variable_", "", dat[["ind"]]), levels = 1:10)
ggplot(dat, aes(x = Time, y = values, color = ID, group = ID)) +
  geom_line(size=.8)

## Plot Example 2
dat <- r_data_frame(12,
  name,
  r_series(grade, 100, relate = "+1_2")
)

# Suggested use of tidyr or reshape2 package here instead
dat <- data.frame(
  ID = rep(dat[[1]], ncol(dat[-1])),
  ind = rep(colnames(dat[-1]), each = nrow(dat)),
  values = unlist(dat[-1])
)

dat[["Time"]] <- as.numeric(sub("Grade_", "", dat[["ind"]]))
ggplot(dat, aes(x = Time, y = values, color = ID, group = ID)) +
  geom_line(size=.8) + theme_bw()

```

Description

grade - Generate a random normal vector of scholastic aptitude test (SATs).

Usage

```
sat(n, mean = 1500, sd = 100, min = 0, max = 2400, digits = 0,  
    name = "SAT")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
mean	The mean value for the normal distribution to be drawn from.
sd	The standard deviation of the normal distribution to draw from.
min	A numeric lower boundary cutoff. Results less than this value will be replaced with min.
max	A numeric upper boundary cutoff. Results greater than this value will be replaced with max.
digits	Integer indicating the number of decimal places to be used. Negative values are allowed (see round).
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random normal vector of SAT elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
sat(10)  
hist(sat(10000))  
interval(sat, 5, n = 1000)
```

second	<i>Generate a Random Sequence of Seconds in H:M:S Format</i>
--------	--

Description

Generate a random vector of seconds in H:M:S format.

Usage

```
second(n, x = seq(0, 59, by = 1)/3600, prob = NULL, random = FALSE,  
       name = "Second")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
random	logical. If TRUE the times are randomized, otherwise the times are sequential.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of second time elements in H:M:S format.

See Also

[times](#)

Examples

```
second(20)  
second(20, random=TRUE)  
pie(table(second(2000, x = seq(0, 59, by = 10)/3600, prob = probs(6))))
```

sentence	<i>Generate Random Vector of Sentences</i>
----------	--

Description

Generate a random vector of sentences from the [presidential_debates_2012](#).

Usage

```
sentence(n, x = wakefield::presidential_debates_2012, prob = NULL,  
        name = "Sentence")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random character vector of sentence elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
sentence(10)
```

seriesname	<i>Add Internal Name to Data Frame</i>
------------	--

Description

Adds `attributes(x)[["seriesname"]]` attribute to a [data.frame](#).

Usage

```
seriesname(x, name)
```

Arguments

x	A data.frame to add a seriesname attribute (i.e., <code>attributes(x)[["seriesname"]]</code>)
name	A name to assign to <code>attributes(x)[["seriesname"]]</code> .

Value

Returns a [data.frame](#) with a `attributes(x)[["seriesname"]]` assigned.

Examples

```
seriesname(mtcars, "Cars")
attributes(seriesname(mtcars, "Cars"))
```

sex	<i>Generate Random Vector of Genders</i>
-----	--

Description

Generate a random vector of genders.

Usage

```
sex(n, x = c("Male", "Female"), prob = c(0.51219512195122,
0.48780487804878), name = "Sex")

gender(n, x = c("Male", "Female"), prob = c(0.51219512195122,
0.48780487804878), name = "Gender")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of length 2 to sample from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The genders and probabilities used match approximate gender make-up:

Gender	Percent
Male	51.22 %
Female	48.78 %

Value

Returns a random factor vector of gender elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
sex(10)
100*table(sex(n <- 10000))/n
```

sex_inclusive

Generate Random Vector of Non-Binary Genders

Description

Generate a random vector of non-binary genders. Proportion of trans* category was taken from the [Williams Institute Report](#) (2011), and subtracted equally from the male and female categories.

Usage

```
sex_inclusive(n, x = c("Male", "Female", "Intersex"), prob = NULL,
  name = "Sex")
```

```
gender_inclusive(n, x = c("Male", "Female", "Trans*"), prob = NULL,
  name = "Gender")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The genders and probabilities used match approximate gender make-up:

Gender	Percent
Male	51.07 %
Female	48.63 %
Trans*	0.30 %

Value

Returns a random factor vector of sex or gender elements.

Author(s)

Matthew Sigal <msigal@yorku.ca>

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
sex_inclusive(10)
barplot(table(sex_inclusive(10000)))

gender_inclusive(10)
```

```
barplot(table(gender_inclusive(10000)))
```

smokes	<i>Generate Random Logical Smokes Vector</i>
--------	--

Description

Generate a random logical (TRUE/FALSE) smokes vector.

Usage

```
smokes(n, prob = c(0.822, 0.178), name = "Smokes")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The probabilities are non-smoker: 82.2% vs. smoker: 17.8%.

Value

Returns a random logical vector of smokes elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
smokes(10)
100*table(smokes(n <- 1000))/n
```

speed

Generate Random Vector of Speeds

Description

speed and speed_in - Generate a random normal vector of speeds in inches.

speed_cm - Generate a random normal vector of speeds in centimeters.

Usage

```
speed(n, mean = 55, sd = 10, min = 0, max = NULL, digits = 0,  
      name = "Speed")
```

```
speed_mph(n, mean = 55, sd = 10, min = 0, max = NULL, digits = 1,  
          name = "Speed(mph)")
```

```
speed_kph(n, mean = 88.5, sd = 16, min = 0, max = NULL, digits = 1,  
          name = "Speed(kph)")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
mean	The mean value for the normal distribution to be drawn from.
sd	The standard deviation of the normal distribution to draw from.
min	A numeric lower boundary cutoff. Results less than this value will be replaced with min.
max	A numeric upper boundary cutoff. Results greater than this value will be replaced with max.
digits	Integer indicating the number of decimal places to be used. Negative values are allowed (see round).
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random normal vector of speed elements.

Note

speed rounds to nearest whole number. speed_in & speed_cm round to the nearest tenths.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [state](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
speed(10)
hist(speed(10000))
interval(speed, 5, n = 1000)
```

<code>state</code>	<i>Generate Random Vector of states</i>
--------------------	---

Description

Generate a random factor vector of states.

Usage

```
state(n, x = datasets::state.name,
      prob = wakefield::state_populations[["Proportion"]], name = "State")
```

Arguments

<code>n</code>	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
<code>x</code>	A vector of elements to chose from.
<code>prob</code>	A vector of probabilities to chose from.
<code>name</code>	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Details

The state populations and probabilities:

State	Population	Percent
California	37,253,956	12.09 %
Texas	25,145,561	8.16 %
New York	19,378,102	6.29 %
Florida	18,801,310	6.10 %
Illinois	12,830,632	4.16 %
Pennsylvania	12,702,379	4.12 %

Ohio	11,536,504	3.74 %
Michigan	9,883,640	3.21 %
Georgia	9,687,653	3.14 %
North Carolina	9,535,483	3.09 %
New Jersey	8,791,894	2.85 %
Virginia	8,001,024	2.60 %
Washington	6,724,540	2.18 %
Massachusetts	6,547,629	2.12 %
Indiana	6,483,802	2.10 %
Arizona	6,392,017	2.07 %
Tennessee	6,346,105	2.06 %
Missouri	5,988,927	1.94 %
Maryland	5,773,552	1.87 %
Wisconsin	5,686,986	1.85 %
Minnesota	5,303,925	1.72 %
Colorado	5,029,196	1.63 %
Alabama	4,779,736	1.55 %
South Carolina	4,625,364	1.50 %
Louisiana	4,533,372	1.47 %
Kentucky	4,339,367	1.41 %
Oregon	3,831,074	1.24 %
Oklahoma	3,751,351	1.22 %
Connecticut	3,574,097	1.16 %
Iowa	3,046,355	.99 %
Mississippi	2,967,297	.96 %
Arkansas	2,915,918	.95 %
Kansas	2,853,118	.93 %
Utah	2,763,885	.90 %
Nevada	2,700,551	.88 %
New Mexico	2,059,179	.67 %
West Virginia	1,852,994	.60 %
Nebraska	1,826,341	.59 %
Idaho	1,567,582	.51 %
Hawaii	1,360,301	.44 %
Maine	1,328,361	.43 %
New Hampshire	1,316,470	.43 %
Rhode Island	1,052,567	.34 %
Montana	989,415	.32 %
Delaware	897,934	.29 %
South Dakota	814,180	.26 %
Alaska	710,231	.23 %
North Dakota	672,591	.22 %
Vermont	625,741	.20 %
Wyoming	563,626	.18 %

Value

Returns a random character vector of state elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [string](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
state(10)
pie(table(state(10000)))
sort(100*table(state(n <- 10000))/n)
```

state_populations	<i>State Populations (2010)</i>
-------------------	---------------------------------

Description

A dataset containing U.S. state populations.

Usage

```
data(state_populations)
```

Format

A data frame with 50 rows and 3 variables

Details

- State. The 50 U.S. states.
- Population. Population of state.
- Proportion. Proportion of total U.S. population.

References

http://en.wikipedia.org/wiki/List_of_U.S._states_and_territories_by_population

string	<i>Generate Random Vector of Strings</i>
--------	--

Description

Generate a random vector of strings.

Usage

```
string(n, x = "[A-Za-z0-9]", length = 10, name = "String")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A character vector specifying character classes to draw elements from.
length	Integer vector, desired string lengths.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random character vector of string elements.

See Also

[stri_rand_strings](#)

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [upper](#), [valid](#), [year](#), [zip_code](#)

Examples

```
string(10)
```

table_heat

View Data Table Column Types as Heat Map

Description

Generate a heat map of column types from a [data.frame](#).

Usage

```
table_heat(x, flip = FALSE, palette = "Set3", print = interactive(),
  sep = "\n")
```

Arguments

x	A data.frame .
flip	logical. If TRUE the data.frame is flipped so that the columns are on the y axis and observations on the x axis. This is useful when there are many columns or the column names are longer.
palette	A palette to chose from. See scale_fill_brewer for more. These choices should exceed the number of unique column types. Use NULL to use ggplot2 's default color scheme.
print	logical. If TRUE the pot is printed. Option for use in document construction such as knitr or rmarkdown .
sep	A separator to use between column types. Column types are determined via <code>sapply(x, class)</code> . When multiple types are present these are collapsed. By default the <code>\n</code> is used.

Details

By default coumn names retain their order. Column types are ordered alphabetically in the legend, with NA appearing last.

Value

Returns a **ggplot2** object.

Examples

```
table_heat(mtcars) #boring
table_heat(CO2)
table_heat(iris)
table_heat(state_populations)

dat <- r_data_frame(100,
  lorem_ipsum,
  birth,
  animal,
```

```

    age,
    grade, grade,
    death,
    dummy,
    grade_letter
  )

table_heat(dat)
table_heat(dat, flip=TRUE)

table_heat(r_data_theme(), flip=TRUE)

## NA values
table_heat(r_na(dat, NULL))

## Colors
table_heat(r_na(dat, NULL), palette = NULL)
table_heat(r_na(dat, NULL), palette = "Set1")
table_heat(r_na(dat, NULL), palette = "Set2")
table_heat(r_na(dat, NULL), palette = "Set1")
table_heat(r_na(dat, NULL), palette = "Dark2")
table_heat(r_na(dat, NULL), palette = "Spectral")
table_heat(r_na(dat, NULL), palette = "Reds")

```

time_stamp

Generate a Random Sequence of Times in H:M:S Format

Description

Generate a random vector of times in H:M:S format.

Usage

```
time_stamp(n, x = seq(0, 23, by = 1), prob = NULL, random = FALSE,
           name = "Time")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
random	logical. If TRUE the times are randomized, otherwise the times are sequential.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of time elements in H:M:S format.

See Also

[times](#)

Examples

```
time_stamp(20)
time_stamp(20, random=TRUE)
pie(table(time_stamp(2000, x = seq(0, 23, by = 2), prob = probs(12))))
```

upper

Generate Random Letter Vector

Description

upper - Generates a random character vector of upper case letters.

lower - Generates a random character vector of lower case letters.

upper_factor - Generates a random factor vector of upper case letters.

lower_factor - Generates a random factor vector of lower case letters.

Usage

```
upper(n, k = 5, x = LETTERS, prob = NULL, name = "Upper")
```

```
lower(n, k = 5, x = c("a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k",
  "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z"),
  prob = NULL, name = "Lower")
```

```
upper_factor(n, k = 5, x = LETTERS, prob = NULL, name = "Upper")
```

```
lower_factor(n, k = 5, x = c("a", "b", "c", "d", "e", "f", "g", "h", "i",
  "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x",
  "y", "z"), prob = NULL, name = "Lower")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
k	The number of the elements of x to sample from (uses 1:k).
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's varname attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random character/factor vector of letter elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [valid](#), [year](#), [zip_code](#)

Examples

```
upper(10)
lower(10)
upper_factor(10)
lower_factor(10)
barplot(table(upper(10000)))
barplot(table(upper(10000, prob = probs(5))))
```

valid

Generate Random Logical Vector

Description

Generate a random logical (TRUE/FALSE) vector.

Usage

```
valid(n, prob = NULL, name = "Valid")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random logical vector of elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [year](#), [zip_code](#)

Examples

```
valid(10)
100*table(valid(n <- 1000))/n
```

variables

Available Variable Functions

Description

See a listing of all available variable functions for use in [r_data_frame](#) or [r_list](#).

Usage

```
variables(type = NULL, ncols = 5, ...)
```

Arguments

type	The output type. Must be either NULL (returns a character vector), "matrix", or "list"; or the user may extract a specific type from a list using: "character", "date", "factor", "integer", "logical", "numeric", "ordered factor". Setting type = TRUE will also return a list . The list version breaks the variable functions into classes. Specifying a specific class (e.g., type = "numeric" will list only variable functions that yield a numeric output.
ncols	The number of columns to use if type = "matrix".
...	Other arguments passed to matrix .

Value

Returns a [character](#) vector, [matrix](#) of all variable functions, or a [list](#) of variable functions by type.

Examples

```
variables()

variables("list")
variables(TRUE)
names(variables("list"))
variables("ordered factor")
variables("numeric")
```

```
variables("matrix")
variables("matrix", ncols=3)
variables("matrix", 1)
variables("matrix", byrow = TRUE)
```

varname	<i>Add Internal Name to Vector</i>
---------	------------------------------------

Description

Adds the class variable and an internal `attributes(x)[["varname"]]` attribute to a vector.

Usage

```
varname(x, name)
```

Arguments

x	A vector to add a varname attribute (i.e., <code>attributes(x)[["varname"]]</code>)
name	A name to assign to <code>attributes(x)[["varname"]]</code> .

Value

Returns a vector of the class variable with a `attributes(x)[["varname"]]` assigned.

Examples

```
varname(1:10, "A")
attributes(varname(1:10, "A"))
sum(varname(1:10, "A"))

varname(LETTERS, "Caps")
attributes(varname(LETTERS, "Caps"))
paste(varname(LETTERS, "Caps"), collapse="")
```

wakefield	<i>Generate Random Data Sets</i>
-----------	----------------------------------

Description

Generates random data sets including: data.frames, lists, and vectors.

year *Generate Random Vector of Years*

Description

Generate a random vector of years.

Usage

```
year(n, x = 1996:as.numeric(format(Sys.Date(), "%Y")), prob = NULL,  
     name = "Year")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of year elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [zip_code](#)

Examples

```
year(10)  
pr <- probs(length(1996:2016))  
pie(table(year(10000, x= 1996:2016, prob = pr)))
```

zip_code	<i>Generate Random Vector of Zip Codes</i>
----------	--

Description

Generate a random vector of zip codes.

Usage

```
zip_code(n, k = 10, x = 10000:99999, prob = NULL, name = "Zip")
```

Arguments

n	The number elements to generate. This can be globally set within the environment of <code>r_data_frame</code> or <code>r_list</code> .
k	The number of the elements of <code>x</code> to sample from (uses <code>sample(x, k)</code>).
x	A vector of elements to chose from.
prob	A vector of probabilities to chose from.
name	The name to assign to the output vector's <code>varname</code> attribute. This is used to auto assign names to the column/vector name when used inside of <code>r_data_frame</code> or <code>r_list</code> .

Value

Returns a random vector of zip code elements.

See Also

Other variable.functions: [age](#), [animal](#), [answer](#), [area](#), [car](#), [children](#), [coin](#), [color](#), [date_stamp](#), [death](#), [dice](#), [dna](#), [dob](#), [dummy](#), [education](#), [employment](#), [eye](#), [grade_level](#), [grade](#), [group](#), [hair](#), [height](#), [income](#), [internet_browser](#), [iq](#), [language](#), [level](#), [likert](#), [lorem_ipsum](#), [marital](#), [military](#), [month](#), [name](#), [normal](#), [political](#), [race](#), [religion](#), [sat](#), [sentence](#), [sex_inclusive](#), [sex](#), [smokes](#), [speed](#), [state](#), [string](#), [upper](#), [valid](#), [year](#)

Examples

```
zip_code(10)
pie(table(zip_code(10000, prob = probs(10))))
```

Index

- *Topic **age**
 - age, 4
- *Topic **animal**
 - animal, 5
- *Topic **answer**
 - answer, 6
- *Topic **area**
 - area, 7
- *Topic **army**
 - military, 41
- *Topic **birth**
 - dob, 17
- *Topic **branch**
 - military, 41
- *Topic **browser**
 - internet_browser, 32
- *Topic **capitals**
 - upper, 84
- *Topic **car**
 - car, 9
- *Topic **character**
 - lorem_ipsum, 39
 - string, 81
- *Topic **children**
 - children, 10
- *Topic **class,**
 - table_heat, 82
- *Topic **coin**
 - coin, 11
- *Topic **color**
 - color, 12
- *Topic **correlate**
 - relate, 52
- *Topic **cut**
 - interval, 33
- *Topic **datasets**
 - animal_list, 6
 - grady_augmented, 25
 - languages, 36
 - name_neutral, 45
 - presidential_debates_2012, 49
 - state_populations, 80
- *Topic **date**
 - date_stamp, 13
- *Topic **death**
 - death, 14
- *Topic **democrat**
 - political, 48
- *Topic **dice**
 - dice, 15
- *Topic **died**
 - death, 14
- *Topic **divorce**
 - marital, 40
- *Topic **dna**
 - dna, 16
- *Topic **dob**
 - dob, 17
- *Topic **dummy**
 - r_dummy, 58
- *Topic **education**
 - education, 19
- *Topic **employment**
 - employment, 20
- *Topic **eye**
 - eye, 21
- *Topic **factor**
 - r_sample_ordered, 66
- *Topic **false**
 - valid, 85
- *Topic **gender**
 - sex, 73
 - sex_inclusive, 74
- *Topic **gpa**
 - grade, 22
- *Topic **grade**
 - grade, 22
 - grade_level, 24

- *Topic **group**
 - group, 25
- *Topic **hair**
 - hair, 26
- *Topic **head**
 - coin, 11
- *Topic **height**
 - height, 27
- *Topic **hour**
 - hour, 29
- *Topic **identification**
 - id, 30
- *Topic **id**
 - id, 30
- *Topic **income**
 - income, 31
- *Topic **insert**
 - r_insert, 59
- *Topic **integer**
 - as_integer, 8
- *Topic **intelligence**
 - iq, 34
- *Topic **interval**
 - interval, 33
- *Topic **iq**
 - iq, 34
- *Topic **iris**
 - eye, 21
- *Topic **language**
 - language, 35
- *Topic **letters**
 - upper, 84
- *Topic **level**
 - level, 37
- *Topic **likert**
 - likert, 38
- *Topic **list**
 - r_list, 60
- *Topic **logical**
 - valid, 85
- *Topic **lower**
 - upper, 84
- *Topic **marines**
 - military, 41
- *Topic **marital**
 - marital, 40
- *Topic **married**
 - marital, 40
- *Topic **military**
 - military, 41
- *Topic **minute**
 - minute, 42
- *Topic **missing**
 - r_na, 61
- *Topic **month**
 - month, 43
- *Topic **name**
 - name, 44
- *Topic **navy**
 - military, 41
- *Topic **na**
 - r_na, 61
- *Topic **normal**
 - normal, 45
- *Topic **no**
 - answer, 6
- *Topic **numeric**
 - as_integer, 8
- *Topic **ordered**
 - r_sample_ordered, 66
- *Topic **percent**
 - probs, 50
- *Topic **pet**
 - animal, 5
- *Topic **political**
 - political, 48
- *Topic **politics**
 - political, 48
- *Topic **probability**
 - probs, 50
- *Topic **race**
 - race, 51
- *Topic **related**
 - relate, 52
- *Topic **religion**
 - religion, 53
- *Topic **republican**
 - political, 48
- *Topic **responses**
 - likert, 38
- *Topic **sat**
 - sat, 69
- *Topic **second**
 - second, 71
- *Topic **sentence**
 - sentence, 72

- *Topic **sex**
 - sex, 73
 - sex_inclusive, 74
 - *Topic **smoking**
 - smokes, 76
 - *Topic **speed**
 - speed, 77
 - *Topic **state**
 - state, 78
 - *Topic **string**
 - lorem_ipsum, 39
 - string, 81
 - *Topic **tail**
 - coin, 11
 - *Topic **time**
 - hour, 29
 - minute, 42
 - second, 71
 - time_stamp, 83
 - *Topic **true**
 - valid, 85
 - *Topic **type**
 - table_heat, 82
 - variables, 86
 - *Topic **upper**
 - upper, 84
 - *Topic **valid**
 - valid, 85
 - *Topic **widow**
 - marital, 40
 - *Topic **year**
 - year, 88
 - *Topic **yes**
 - answer, 6
 - *Topic **zip_code**
 - zip_code, 89
-
- age, 4, 5, 7–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 52, 54, 56, 58, 60, 68, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - animal, 4, 5, 7–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - animal_list, 6
 - answer, 4, 5, 6, 8–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - area, 4, 5, 7, 7, 9–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - as.integer, 8
 - as_integer, 8
 - birth (dob), 17
 - car, 4, 5, 7, 8, 9, 10–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - character, 30, 86
 - children, 4, 5, 7–9, 10, 11–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - class, 8
 - coin, 4, 5, 7–10, 11, 12–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - color, 4, 5, 7–11, 12, 13–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - cut, 33, 34
 - data.frame, 8, 46, 56, 59–61, 73, 82
 - date_stamp, 4, 5, 7–12, 13, 14–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - death, 4, 5, 7–13, 14, 15–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - dice, 4, 5, 7–14, 15, 16–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - died (death), 14
 - dna, 4, 5, 7–15, 16, 17, 18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - dob, 4, 5, 7–16, 17, 18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - dummy, 4, 5, 7–17, 18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
 - education, 4, 5, 7–18, 19, 21–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89

- ela (level), 37
- employment, 4, 5, 7–18, 20, 20, 22–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- eye, 4, 5, 7–18, 20, 21, 21, 23, 24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- factor, 8, 30
- gamma, 31
- gender (sex), 73
- gender_inclusive (sex_inclusive), 74
- gpa (grade), 22
- grade, 4, 5, 7–18, 20–22, 22, 24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- grade_letter (grade), 22
- grade_level, 4, 5, 7–18, 20–23, 24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- grady_augmented, 25
- group, 4, 5, 7–18, 20–24, 25, 27, 28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- hair, 4, 5, 7–18, 20–24, 26, 26, 28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- head, 47
- height, 4, 5, 7–18, 20–24, 26, 27, 27, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- height_cm (height), 27
- height_in (height), 27
- hour, 29
- id, 30
- id_factor (id), 30
- income, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- internet_browser, 4, 6–18, 20–24, 26–28, 31, 32, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- interval, 33
- iq, 4, 6–18, 20–24, 26–28, 31, 33, 34, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- language, 4, 6–18, 20–24, 26–28, 31, 33, 35, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- languages, 36
- level, 4, 6–18, 20–24, 26–28, 31, 33, 35, 37, 39–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- likert, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38, 38, 40–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- likert_5 (likert), 38
- likert_7 (likert), 38
- list, 60, 61, 86
- lorem_ipsum, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38, 39, 39, 41–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- lower (upper), 84
- lower_factor (upper), 84
- marital, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–40, 40, 42–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- math (level), 37
- matrix, 86
- military, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–41, 41, 43, 44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- minute, 42
- month, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–42, 43, 44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- mtcars, 9
- name, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–43, 44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- name_neutral, 45
- normal, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 45, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- normal_round (normal), 45
- ordered, 66
- package-wakefield (wakefield), 87
- paragraph (lorem_ipsum), 39
- peek, 46
- pet (animal), 5

- plot.tbl_df, 47
- political, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 48, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- presidential_debates_2012, 35, 49, 72
- primary (color), 12
- print.available, 49
- print.variable, 50
- probs, 50
- r_data, 54
- r_data_frame, 55, 56, 58–60, 68, 86
- r_data_theme (r_data), 54
- r_dummy, 56, 58, 60, 68
- r_insert, 59
- r_list, 56, 58, 59, 60, 68, 86
- r_na, 61
- r_sample, 62
- r_sample_binary, 63
- r_sample_binary_factor (r_sample_binary), 63
- r_sample_factor, 64
- r_sample_integer, 64
- r_sample_logical, 65
- r_sample_ordered, 66
- r_sample_replace, 67
- r_series, 8, 52, 56, 58, 60, 68
- race, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 51, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- relate, 52, 68
- religion, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 53, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- rnorm, 45, 46
- round, 23, 28, 31, 34, 46, 70, 77
- sample, 62, 64–67
- sample.int, 18, 63
- sat, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 69, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- scale_fill_brewer, 82
- second, 71
- sentence, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- seq.Date, 13, 17
- seriesname, 59, 73
- sex, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 73, 75, 76, 78, 80, 81, 85, 86, 88, 89
- sex_inclusive, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74, 74, 76, 78, 80, 81, 85, 86, 88, 89
- smokes, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74, 75, 76, 78, 80, 81, 85, 86, 88, 89
- speed, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 77, 80, 81, 85, 86, 88, 89
- speed_kph (speed), 77
- speed_mph (speed), 77
- sprintf, 30
- state, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 78, 81, 85, 86, 88, 89
- state_populations, 80
- stri_rand_lipsum, 39, 40
- stri_rand_strings, 81
- string, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- table_heat, 48, 82
- tbl_df, 52, 55, 56, 58, 68
- time_stamp, 83
- times, 29, 42, 71, 84
- upper, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 84, 86, 88, 89
- upper_factor (upper), 84
- valid, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 85, 88, 89
- variables, 86
- varname, 87
- wakefield, 87
- wakefield-package (wakefield), 87
- year, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89
- zip_code, 4, 6–18, 20–24, 26–28, 31, 33, 35, 38–44, 46, 49, 52, 54, 70, 72, 74–76, 78, 80, 81, 85, 86, 88, 89