Package ‘wakefield’

September 13, 2020

Title Generate Random Data Sets
Version 0.3.6
Maintainer Tyler Rinker <tyler.rinker@gmail.com>
Description Generates random data sets including: data.frames, lists, and vectors.
Depends R (>= 3.2.0)
Imports chron, ggplot2, dplyr, stringi
Suggests testthat
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 7.1.1
NeedsCompilation no
Author  Tyler Rinker [aut, cre],
        Josh O'Brien [ctb],
        Ananda Mahto [ctb],
        Matthew Sigal [ctb],
        Jonathan Carroll [ctb],
        Scott Westenberger [ctb]

Repository  CRAN

Date/Publication  2020-09-13 17:30:02 UTC

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 ................................................................. 22
grade ............................................................... 23
grade_level ......................................................... 24
grady_augmented .................................................. 25
group ............................................................... 26
hair ................................................................. 27
height .............................................................. 28
hour ................................................................. 30
id ................................................................. 31
income ............................................................. 32
internet_browser .................................................... 33
interval ........................................................... 34
iq ................................................................. 35
language ........................................................... 36
languages ......................................................... 37
level .............................................................. 38
likert ............................................................. 39
lorem_ipsum ....................................................... 40
R topics documented:

marital .................................................. 41
military .................................................. 42
minute ................................................... 43
month ..................................................... 44
name ...................................................... 45
name_neutral .......................................... 46
normal ................................................... 47
peek ...................................................... 48
plot.tbl_df ............................................ 49
political ............................................... 50
presidential_debates_2012 ......................... 51
print.available ........................................ 51
print.variable ........................................ 52
probs .................................................... 52
race ...................................................... 53
relate ................................................... 54
religion .................................................. 55
r_data .................................................... 57
r_data_frame ......................................... 58
r_dummy ................................................ 60
r_insert ............................................... 61
r_list .................................................... 62
r_na ...................................................... 63
r_sample .............................................. 64
r_sample_binary ..................................... 65
r_sample_factor ..................................... 66
r_sample_integer .................................... 67
r_sample_logical .................................... 68
r_sample_ordered ................................... 68
r_sample_replace ................................... 69
r_series ............................................... 70
sat ....................................................... 72
second .................................................. 73
sentence ............................................... 74
seriesname ............................................ 75
sex ....................................................... 75
sex_inclusive ......................................... 76
smokes .................................................. 78
speed .................................................... 79
state ..................................................... 80
state_populations .................................... 83
string ................................................... 83
table_heat ............................................ 84
time_stamp ............................................ 85
upper .................................................... 86
valid ..................................................... 88
variables ............................................... 89
varname ............................................... 90
Generate Random Vector of Ages

Description

Generate a random vector of ages within the provided range. The default age range is set between 18 and 89, to match the age ranges which appear (see e.g., https://gssdataexplorer.norc.org/variables/53/vshow).

Usage

```r
age(n, x = 18:89, prob = NULL, name = "Age")
```

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 choose from.
- `prob`: A vector of probabilities to choose 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 integer vector of ages within the provided range (defaults to 18:89).

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

```r
age(10) # draw 10 ages with default values
hist(age(n=10000))
interval(age, 3, n = 1000)
```
animal  Generate Random Vector of animals

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 r_data_frame or r_list.
k The number of the elements of x to sample from (uses sample(x,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 r_data_frame or r_list.

Details

The household pets and probabilities:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td>36.5 %</td>
</tr>
<tr>
<td>Cat</td>
<td>30.4 %</td>
</tr>
<tr>
<td>None</td>
<td>25.8 %</td>
</tr>
<tr>
<td>Bird</td>
<td>3.1 %</td>
</tr>
<tr>
<td>Horse</td>
<td>1.5 %</td>
</tr>
</tbody>
</table>

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()`, `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  Animal List

Description

A dataset containing a character vector animals

Usage

data(animal_list)

Format

A character vector with 591 elements

References

https://a-z-animals.com/animals

---

answer  Generate Random Vector of Answers (Yes/No)

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 `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 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

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

---

**area**  
*Generate Random Vector of Areas*

Description

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

Usage

```r
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 `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`. 
as_integer

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)))
Examples

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

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

---

**car**

*Generate Random Vector of Cars*

**Description**

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

**Usage**

```r
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 `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 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()`
### children

**Generate Random Vector of Number of Children**

**Examples**

```r
children(10)
table(car(10000))
```

**Description**

Generate a random vector of number of children.

**Usage**

```r
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 `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 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**

```r
children(10)
table(table(children(100)))
```
coin

*Generate Random Vector of Coin Flips*

**Description**

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

**Usage**

```r
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 `r_data_frame` or `r_list`.
- `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 `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 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**

```r
coin(10)
100*table(coin(n <- 10000))/n
```
Generate Random Vector of Colors

Description

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

Usage

```r
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 `r_data_frame` or `r_list`.
- `k` The number of the elements of `x` to sample from (uses `sample(x,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 `r_data_frame` or `r_list`.

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()`
date_stamp

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

Generate Random Vector of Dates

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 r_data_frame or r_list.
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 start, k, and by. The x argument takes precedence over the other three if !is.null. Note that start, k, and by 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 start.
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 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 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(), 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

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>The number elements to generate. This can be globally set within the environment of r_data_frame or r_list.</td>
</tr>
<tr>
<td>prob</td>
<td>A vector of probabilities to choose from.</td>
</tr>
<tr>
<td>name</td>
<td>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.</td>
</tr>
</tbody>
</table>

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

dead(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

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 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()`
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

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 automatically assign names to the column/vector name when used inside of r_data_frame or r_list.

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)))


**Description**

Generate a random vector of birth dates.

**Usage**

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

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

**Arguments**

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `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 `start`, `k`, and `by`. The `x` argument takes precedence over the other three if !is.null. Note that `start`, `k`, and `by` 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 `start`.
- `by` The interval to use in building the sequence.
- `prob` A vector of probabilities to chose from.
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 r_data_frame or r_list.

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

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

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()
employment

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 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 following arbitrary probabilities are used:

<table>
<thead>
<tr>
<th>Employment Status</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Time</td>
<td>60%</td>
</tr>
<tr>
<td>Part Time</td>
<td>10%</td>
</tr>
<tr>
<td>Unemployed</td>
<td>10%</td>
</tr>
<tr>
<td>Retired</td>
<td>10%</td>
</tr>
<tr>
<td>Student</td>
<td>10%</td>
</tr>
</tbody>
</table>

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)))
Generate Random Vector of Eye Colors

Description
Generate a random vector of eye colors.

Usage
```r
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 automatically assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Details
The eye colors and probabilities:

<table>
<thead>
<tr>
<th>Color</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brown</td>
<td>44 %</td>
</tr>
<tr>
<td>Blue</td>
<td>30 %</td>
</tr>
<tr>
<td>Green</td>
<td>13 %</td>
</tr>
<tr>
<td>Hazel</td>
<td>9 %</td>
</tr>
<tr>
<td>Gray</td>
<td>4 %</td>
</tr>
</tbody>
</table>

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()`, `education()`.
grade

Examples

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

grade

Generate Random Vector of Grades

Description

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

Usage

```r
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 `r_data_frame` or `r_list`.
- **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 `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.
- **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:

<table>
<thead>
<tr>
<th>Percent</th>
<th>Letter</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>97-100</td>
<td>A+</td>
<td>4.00</td>
</tr>
<tr>
<td>93-96</td>
<td>A</td>
<td>4.00</td>
</tr>
</tbody>
</table>
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

```r
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**

Generate Random Vector of Grade Levels

Description

Generate a random vector of grade levels.
Usage

```r
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 `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 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

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

---

**grady_augmented**

*Augmented List of Grady Ward’s English Words and Mark Kantrowitz’s Names List*

**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 https://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 http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/nlp/corpora/names/readme.txt per the author’s request.

group

Generate Random Vector of Control/Treatment Groups

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 r_data_frame or r_list.

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 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 factor vector of group (control/treatment) elements.
**hair**

**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**

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

---

**Generate Random Vector of Hair Colors**

**Description**

Generate a random vector of hair colors.

**Usage**

```r
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 `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 hair colors and probabilities:
### 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

```r
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)
```

### 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

```r
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 `r_data_frame` or `r_list`.
- **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 `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 normal vector of height elements.

**Note**

`height` rounds to nearest whole number. `height_in` & `height_in` 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()`. 
Examples

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

---

### hour

Generate a Random Sequence of H:M:S Times

#### Description

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

#### Usage

```r
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 `r_data_frame` or `r_list`.
- `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 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 H:M:S time elements.

#### See Also

- `times`

#### Examples

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

```r
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 `r_data_frame` or `r_list`.
- **random**: logical. If TRUE the IDs are randomized, otherwise the IDs are sequential.
- **name**: The name to assign to the output vector’s `varname` attribute. This is used to automatically assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**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**

```r
id(1000)
r_data_frame(n=21, id)
```
## income

### Generate Random Gamma Vector of Incomes

**Description**

Generate a random gamma vector of incomes.

**Usage**

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

**Arguments**

- **n**: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- **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 `varname` attribute. This is used to automatically assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

Incomes are generated using: `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**

```r
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 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 browser use and probabilities (from https://gs.statcounter.com/):

<table>
<thead>
<tr>
<th>Browser</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chrome</td>
<td>50.27 %</td>
</tr>
<tr>
<td>IE</td>
<td>17.50 %</td>
</tr>
<tr>
<td>Firefox</td>
<td>16.89 %</td>
</tr>
<tr>
<td>Safari</td>
<td>9.94 %</td>
</tr>
<tr>
<td>Opera</td>
<td>1.70 %</td>
</tr>
<tr>
<td>Android</td>
<td>1.32 %</td>
</tr>
</tbody>
</table>

Value

Returns a random factor vector of Internet browser elements.

References

https://gs.statcounter.com/
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

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

---

**interval**  
**Cut Numeric Into Factor**

Description

A wrapper for `cut` that cuts the vector and then adds the `varname` produced by the original function.

Usage

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

Arguments

- `fun`  
  A vector producing function.

- `breaks`  
  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 `fun` is to be cut.

- `labels`  
  Labels for the levels of the resulting category. By default, labels are constructed using "(a,b]" interval notation. If `labels = FALSE`, simple integer codes are returned instead of a factor.

- `include.lowest`  
  Logical. If TRUE an 'x[i]' equal to the lowest (or highest, for `right = FALSE`) 'breaks' value should be included.
Generate Random Vector of Intelligence Quotients (IQs)

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

Usage

```r
data iq
```

Arguments

- `n` The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `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 `varname` attribute. This is used to auto assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

Examples

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

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 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 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

```r
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, "%"))
```

Description


Usage

```r
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

level

Generate Random Vector of Levels

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 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

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

<table>
<thead>
<tr>
<th>Level</th>
<th>ELA</th>
<th>Math</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>31.6%</td>
<td>29.8%</td>
</tr>
<tr>
<td>2</td>
<td>37.3%</td>
<td>33.3%</td>
</tr>
<tr>
<td>3</td>
<td>22.3%</td>
<td>22.8%</td>
</tr>
<tr>
<td>4</td>
<td>8.8%</td>
<td>14.0%</td>
</tr>
</tbody>
</table>

Value

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

Other variable functions: `age()`, `animal()`, `answer()`, `area()`, `car()`, `children()`, `coin()`, `color()`, `datestamp()`, `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

```r
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**

```r
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 `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 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

```r
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 r_data_frame or r_list.

... Other arguments passed to stri_rand_lipsum.

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 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

```r
marital(10)
b AR GR
barplot(table(marital(10000)))
```

---

### military  
Generate Random Vector of Military Branches

Description

Generate a random vector of military branches.

Usage

```r
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 `r_data_frame` or `r_list`.
- **x**: A vector of elements to choose from.
- **prob**: A vector of probabilities to choose 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 military branches and probabilities used match approximate U.S. military make-up:

<table>
<thead>
<tr>
<th>Branch</th>
<th>N</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Army</td>
<td>541,291</td>
<td>37.9%</td>
</tr>
<tr>
<td>Air Force</td>
<td>333,772</td>
<td>23.3%</td>
</tr>
<tr>
<td>Navy</td>
<td>317,237</td>
<td>22.2%</td>
</tr>
<tr>
<td>Marine Corps</td>
<td>195,338</td>
<td>13.7%</td>
</tr>
<tr>
<td>Coast Guard</td>
<td>42,357</td>
<td>3.0%</td>
</tr>
</tbody>
</table>

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

```r
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.
month

Generate Random Vector of Months

Usage

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

Description

Generate a random factor vector of months.

Usage

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

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 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  Generate Random Vector of Names

Description

Generate a random vector of first names. This dataset includes all unique entries from the babynames package.

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 \texttt{r\_data\_frame} or \texttt{r\_list}.

x  A vector of elements to chose from.

prob  A vector of probabilities to chose from.

replace  logical. If \texttt{TRUE} sampling is done with replacement. Default is without replacement.

name  The name to assign to the output vector’s \texttt{varname} attribute. This is used to auto assign names to the column/vector name when used inside of \texttt{r\_data\_frame} or \texttt{r\_list}.

Value

Returns a random vector of name elements.

See Also

Other variable functions: \texttt{age()}, \texttt{animal()}, \texttt{answer()}, \texttt{area()}, \texttt{car()}, \texttt{children()}, \texttt{coin()}, \texttt{color}, \texttt{date\_stamp()}, \texttt{death()}, \texttt{dice()}, \texttt{dna()}, \texttt{dob()}, \texttt{dummy()}, \texttt{education()}, \texttt{employment()}, \texttt{eye()}, \texttt{grade\_level()}, \texttt{grade()}, \texttt{group()}, \texttt{hair()}, \texttt{height()}, \texttt{income()}, \texttt{internet\_browser()}, \texttt{iq()}, \texttt{language\_level()}, \texttt{likert()}, \texttt{lorem\_ipsum()}, \texttt{marital()}, \texttt{military()}, \texttt{month()}, \texttt{normal()}, \texttt{political()}, \texttt{race()}, \texttt{religion()}, \texttt{sat()}, \texttt{sentence()}, \texttt{sex\_inclusive()}, \texttt{sex()}, \texttt{smokes()}, \texttt{speed()}, \texttt{state()}, \texttt{string()}, \texttt{upper()}, \texttt{valid()}, \texttt{year()}, \texttt{zip\_code()}

Examples

\begin{verbatim}
name(10)
name(100)
name(1000, replace = TRUE)
\end{verbatim}

\begin{longtable}{ll}
\texttt{name\_neutral} & \textit{Gender Neutral Names} \\
\hline
\end{longtable}

Description

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

Usage

\begin{verbatim}
data(name\_neutral)
\end{verbatim}

Format

A character vector with 662 elements

References

http://www.census.gov
**normal**  
*Generate Random Normal Vector*

**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 `r_data_frame` or `r_list`.  
- `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 `r_data_frame` or `r_list`.  
- `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)

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 data.frame 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.
plot.tbl_df

Plots a tbl_df Object

Description

Plots a tbl_df object.

Usage

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

Arguments

- `x` The tbl_df object.
- `...` Arguments passed to `table_heat`.
Generate Random Vector of Political Parties

Description

Generate a random vector of political parties.

Usage

```r
political(
  n,
  x = c("Democrat", "Republican", "Constitution", "Libertarian", "Green"),
  prob = c(0.577269133302094, 0.410800432748879, 0.00491084954793489,
           0.0037259030330866, 0.0032936813677832),
  name = "Political"
)
```

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

<table>
<thead>
<tr>
<th>Party</th>
<th>N</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Democrat</td>
<td>43,140,758</td>
<td>57.73%</td>
</tr>
<tr>
<td>Republican</td>
<td>30,700,138</td>
<td>41.08%</td>
</tr>
<tr>
<td>Constitution</td>
<td>367,000</td>
<td>.49%</td>
</tr>
<tr>
<td>Libertarian</td>
<td>278,446</td>
<td>.37%</td>
</tr>
<tr>
<td>Green</td>
<td>246,145</td>
<td>.33%</td>
</tr>
</tbody>
</table>

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

```r
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**

```r
data(presidential_debates_2012)
```

**Format**

A character vector with 2911 elements

---

**Print an available Object.**

**Description**

Prints an available object.

**Usage**

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

**Arguments**

- `x` The available object
- `...` ignored
**print.variable**  
*Prints a variable Object*

**Description**
Prints a variable object

**Usage**

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

**Arguments**

- `x`  
The variable object.
- `...`  
Ignored.

**probs**  
*Generate a Random Vector of Probabilities.*

**Description**
Generate a random vector of probabilities that sum to 1.

**Usage**

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

**Arguments**

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

**Value**
Returns a vector of probabilities summing to 1.

**Examples**

```r
probs(10)
sum(probs(100))
pie(table(month(10000, prob = probs(12))))
```
### Generate Random Vector of Races

**Description**

Generate a random vector of races.

**Usage**

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

<table>
<thead>
<tr>
<th>Race</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>White</td>
<td>63.70 %</td>
</tr>
<tr>
<td>Hispanic</td>
<td>16.30 %</td>
</tr>
<tr>
<td>Black</td>
<td>12.20 %</td>
</tr>
<tr>
<td>Asian</td>
<td>4.70 %</td>
</tr>
<tr>
<td>Bi-Racial</td>
<td>1.90 %</td>
</tr>
<tr>
<td>Native</td>
<td>.70 %</td>
</tr>
<tr>
<td>Other</td>
<td>.20 %</td>
</tr>
<tr>
<td>Hawaiian</td>
<td>.15 %</td>
</tr>
</tbody>
</table>

**Value**

Returns a random factor vector of elements.
See Also

Other variable functions: \texttt{age()}, \texttt{animal()}, \texttt{answer()}, \texttt{area()}, \texttt{car()}, \texttt{children()}, \texttt{coin()}, \texttt{color()}, \texttt{date\_stamp()}, \texttt{death()}, \texttt{dice()}, \texttt{dna()}, \texttt{dob()}, \texttt{dummy()}, \texttt{education()}, \texttt{employment()}, \texttt{eye()}, \texttt{grade\_level()}, \texttt{grade()}, \texttt{group()}, \texttt{hair()}, \texttt{height()}, \texttt{income()}, \texttt{internet\_browser()}, \texttt{iq()}, \texttt{language()}, \texttt{level()}, \texttt{likert()}, \texttt{lorem\_ipsun()}, \texttt{marital()}, \texttt{military()}, \texttt{month()}, \texttt{name()}, \texttt{normal()}, \texttt{political()}, \texttt{religion()}, \texttt{sat()}, \texttt{sentence()}, \texttt{sex\_inclusive()}, \texttt{sex()}, \texttt{smokes()}, \texttt{speed()}, \texttt{state()}, \texttt{string()}, \texttt{upper()}, \texttt{valid()}, \texttt{year()}, \texttt{zip\_code()}

Examples

\begin{verbatim}
race(10)
100*table(race(n <- 10000))/n
\end{verbatim}

\section*{Description}

Generate columns that are related.

\section*{Usage}

\begin{verbatim}
relate(
x, j,
name = NULL,
operation = "+",
mean = 5,
sd = 1,
rep.sep = "",
digits = max(nchar(sub("[^.]*.", "", x)))
)
\end{verbatim}

\section*{Arguments}

\begin{verbatim}
x \hspace{1cm} A starting column.
j \hspace{1cm} The number of columns to produce.
name \hspace{1cm} 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 \hspace{1cm} A operation character vector of length 1; either c("+", "-", "+", "/"). This is the relationship between columns.
mean \hspace{1cm} Mean is the average value to add, subtract, multiply, or divide by.
sd \hspace{1cm} 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).
\end{verbatim}
A separator to use for repeated variable names. For example if the age is used three times (r_data_frame(age, age, age)), 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

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

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

Value

Returns a random factor vector of religion elements.

References

https://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,
\texttt{r_data}

\begin{verbatim}
normal(), political(), race(), sat(), sentence(), sex_inclusive(), sex(), smokes(), speed(), state(), string(), upper(), valid(), year(), zip_code()
\end{verbatim}

Examples

\begin{verbatim}
religion(10)
barplot(table(religion(10000)))
pie(table(religion(10000)))
\end{verbatim}

\textbf{r_data \hspace{1cm} Pre-Selected Column Data Set}

Description

\texttt{r_data} - Generate a data set with pre-set columns selected.
\texttt{r_data\_theme} - Generate a themed data set with pre-set columns.

Usage

\begin{verbatim}
r_data(n = 500, ...)
r_data_theme(n = 100, data_theme = "the\_works")
\end{verbatim}

Arguments

\begin{verbatim}
n \hspace{1cm} The length to pass to the randomly generated vectors (number of rows).
data\_theme \hspace{1cm} A data theme. Currently selections include:
\texttt{the\_works} all available variable functions
\texttt{survey} ID column plus 10 numeric 5-point Likert type response columns
\texttt{survey2} ID column plus 10 5-point Likert type response columns
\texttt{...} A set of optionally named arguments. Using \texttt{wakefield} variable functions require no name or call parenthesis.
\end{verbatim}

Details

The pre-selected columns include:

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

The user may use \ldots to add additional columns. \texttt{r_data} is a convenience function to quickly produce a data set. For more specific usage use the more flexible \texttt{r_data\_frame} function.
**r_data_frame**

**Description**

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

**Usage**

```r
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 `age` is used three times (`r_data_frame(age, age, age)`), the name "Age" will be assigned to all three columns. The results in column names `c("Age_1", "Age_2", "Age_3")`. To turn of this behavior use `rep.sep = NULL`. This results in `c("Age", "Age.1", "Age.2")` column names in the `data.frame`.

- `...`  
  A set of optionally named arguments. Using `wakefield` variable functions require no name or call parenthesis.

**Value**

Returns a `tbl_df`. 

---

**Examples**

```r
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")
```
Author(s)

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

References

https://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_dummy
)
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)
library(wakefield)
library(dplyr)
library(ggplot2)
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

**Generate Random Dummy Values**

**Description**

Generate random values from a `wakefield` variable function.

**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 age is used (r_dummy(sex)), this results in column names c("Sex_Male","Sex_Female").

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")
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

- `n` The length to pass to the randomly generated vectors.
- `rep.sep` A separator to use for repeated variable names. For example if the `age` is used three times (`r_list(age, age, age)`), the name "Age" will be assigned to all three vectors in the list. The results in column names `c("Age_1", "Age_2", "Age_3")`. To turn of this behavior use `rep.sep = NULL`. This results in `c("Age", "Age", "Age")` for vector names, leading to `c("Age", "Age.1", "Age.2")` if coerced to a `data.frame`.
- `...` 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>.
r_na

References
https://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

Replace a Proportion of Values With NA

Description
Replaces a proportion of values with NA. Useful for simulating missing data.

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
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
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 `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 elements.

See Also

- `sample`

Examples

```r
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

**Generate Random Binary Vector**

**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
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`. 
Generate Random Factor Vector

Description

Generate a random vector and coerces to a factor.

Usage

```r
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 `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 actor vector of elements.

See Also

`sample`
Generate Random Integer Vector

Description

Generate a random integer vector.

Usage

```r
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 `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 integer vector of elements.

See Also

- `sample`

Examples

```r
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

**Generate Random Logical Vector**

**Description**

Generate a random logical (TRUE/FALSE) vector.

**Usage**

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

**Arguments**

- `n`: The number elements to generate. This can be globally set within the environment of `r_data_frame` or `r_list`.
- `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 logical (TRUE/FALSE) vector of elements.

**See Also**

`sample`

**Examples**

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

---

### r_sample_ordered

**Generate Random Ordered Factor Vector**

**Description**

Generate a random vector and coerces to an ordered factor.

**Usage**

```r
r_sample_ordered(n, x = LETTERS[1:5], prob = NULL, name = "Ordered")
```
Generate Random Vector (Without Replacement)

**Description**

Generate a random vector without replacement.

**Usage**

```r
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 `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 factor vector of elements.

**Examples**

```r
r_sample_replace(100, name = "Var")

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

(out <- r_sample_replace(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)))
```
replace logical. If TRUE sampling is done with replacement. Default is without replacement.

ame 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 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 c("operation","mean","sd") 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 (r_data_frame(age, age, age)), the name "Age" will be assigned to all three columns. The results in column names c("Age_1", "Age_2", "Age_3").

... 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 = "\times", 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()

---

sat Generate Random Vector of Scholastic Aptitude Test (SATs)

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 r_data_frame or r_list.
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 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 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

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

**second**

*Generate a Random Sequence of Seconds in H:M:S Format*

Description

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

Usage

```r
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 `r_data_frame` or `r_list`.
- `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 `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 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  Generate Random Vector of Sentences

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 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 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**

*Add Internal Name to Data Frame*

**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., attributes(x)["seriesname"]).
- `name` A name to assign to attributes(x)["seriesname"].

**Value**

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

**Examples**

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

---

**sex**

*Generate Random Vector of Genders*

**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 = "Sex"
)
```
sex_inclusive

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 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.

Details

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

<table>
<thead>
<tr>
<th>Gender</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>51.22%</td>
</tr>
<tr>
<td>Female</td>
<td>48.78%</td>
</tr>
</tbody>
</table>

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
**sex_inclusive**

**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 `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 automatically assign names to the column/vector name when used inside of `r_data_frame` or `r_list`.

**Details**

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

<table>
<thead>
<tr>
<th>Gender</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Male</td>
<td>51.07 %</td>
</tr>
<tr>
<td>Female</td>
<td>48.63 %</td>
</tr>
<tr>
<td>Trans*</td>
<td>0.30 %</td>
</tr>
</tbody>
</table>

**Value**
Returns a random factor vector of sex or gender elements.

**Author(s)**
Matthew Sigal <msigel@yorku.ca>
See Also

Other variable functions: age(), animal(), answer(), area(), car(), children(), coin(),
color, date_stmp(), 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 Generate Random Logical Smokes Vector

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 r_data_frame or r_list.
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 probabilities are non-smoker: 82.2% vs. smoker: 17.8%.

Value

Returns a random logical vector of smokes elements.
speed

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

```r
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

```r
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 \texttt{r\_data\_frame} or \texttt{r\_list}.
- **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 \texttt{min}.
- **max**: A numeric upper boundary cutoff. Results greater than this value will be replaced with \texttt{max}.
- **digits**: Integer indicating the number of decimal places to be used. Negative values are allowed (see \texttt{round}).
- **name**: The name to assign to the output vector's \texttt{varname} attribute. This is used to auto assign names to the column/vector name when used inside of \texttt{r\_data\_frame} or \texttt{r\_list}.

Value
Returns a random normal vector of speed elements.

Note
speed rounds to nearest whole number. speed\_in & speed\_in round to the nearest tenths.

See Also
Other variable functions: \texttt{age()}, \texttt{animal()}, \texttt{answer()}, \texttt{area()}, \texttt{car()}, \texttt{children()}, \texttt{coin()}, \texttt{color()}, \texttt{date\_stamp()}, \texttt{death()}, \texttt{dice()}, \texttt{dna()}, \texttt{dob()}, \texttt{dummy()}, \texttt{education()}, \texttt{employment()}, \texttt{eye()}, \texttt{grade\_level()}, \texttt{grade()}, \texttt{group()}, \texttt{hair()}, \texttt{height()}, \texttt{income()}, \texttt{internet\_browser()}, \texttt{iq()}, \texttt{language()}, \texttt{level()}, \texttt{likert()}, \texttt{lorem\_ipsum()}, \texttt{marital()}, \texttt{military()}, \texttt{month()}, \texttt{name()}, \texttt{normal()}, \texttt{political()}, \texttt{race()}, \texttt{religion()}, \texttt{sat()}, \texttt{sentence()}, \texttt{sex\_inclusive()}, \texttt{sex()}, \texttt{smokes()}, \texttt{state()}, \texttt{string()}, \texttt{upper()}, \texttt{valid()}, \texttt{year()}, \texttt{zip\_code()}

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

<table>
<thead>
<tr>
<th>state</th>
<th>Generate Random Vector of states</th>
</tr>
</thead>
</table>

Description
Generate a random factor vector of states.
Usage

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

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 state populations and probabilities:

<table>
<thead>
<tr>
<th>State</th>
<th>Population</th>
<th>Percent</th>
</tr>
</thead>
<tbody>
<tr>
<td>California</td>
<td>37,253,956</td>
<td>12.09 %</td>
</tr>
<tr>
<td>Texas</td>
<td>25,145,561</td>
<td>8.16 %</td>
</tr>
<tr>
<td>New York</td>
<td>19,378,102</td>
<td>6.29 %</td>
</tr>
<tr>
<td>Florida</td>
<td>18,801,310</td>
<td>6.10 %</td>
</tr>
<tr>
<td>Illinois</td>
<td>12,830,632</td>
<td>4.16 %</td>
</tr>
<tr>
<td>Pennsylvania</td>
<td>12,702,379</td>
<td>4.12 %</td>
</tr>
<tr>
<td>Ohio</td>
<td>11,536,504</td>
<td>3.74 %</td>
</tr>
<tr>
<td>Michigan</td>
<td>9,883,640</td>
<td>3.21 %</td>
</tr>
<tr>
<td>Georgia</td>
<td>9,687,653</td>
<td>3.14 %</td>
</tr>
<tr>
<td>North Carolina</td>
<td>9,535,483</td>
<td>3.09 %</td>
</tr>
<tr>
<td>New Jersey</td>
<td>8,791,894</td>
<td>2.85 %</td>
</tr>
<tr>
<td>Virginia</td>
<td>8,001,024</td>
<td>2.60 %</td>
</tr>
<tr>
<td>Washington</td>
<td>6,724,540</td>
<td>2.18 %</td>
</tr>
<tr>
<td>Massachusetts</td>
<td>6,547,629</td>
<td>2.12 %</td>
</tr>
<tr>
<td>Indiana</td>
<td>6,483,802</td>
<td>2.10 %</td>
</tr>
<tr>
<td>Arizona</td>
<td>6,392,017</td>
<td>2.07 %</td>
</tr>
<tr>
<td>Tennessee</td>
<td>6,346,105</td>
<td>2.06 %</td>
</tr>
<tr>
<td>Missouri</td>
<td>5,988,927</td>
<td>1.94 %</td>
</tr>
<tr>
<td>Maryland</td>
<td>5,773,552</td>
<td>1.87 %</td>
</tr>
<tr>
<td>Wisconsin</td>
<td>5,686,986</td>
<td>1.85 %</td>
</tr>
<tr>
<td>Minnesota</td>
<td>5,303,925</td>
<td>1.72 %</td>
</tr>
<tr>
<td>Colorado</td>
<td>5,029,196</td>
<td>1.63 %</td>
</tr>
<tr>
<td>Alabama</td>
<td>4,779,736</td>
<td>1.55 %</td>
</tr>
<tr>
<td>South Carolina</td>
<td>4,625,364</td>
<td>1.50 %</td>
</tr>
<tr>
<td>Louisiana</td>
<td>4,533,372</td>
<td>1.47 %</td>
</tr>
<tr>
<td>State</td>
<td>Population</td>
<td>Percentage</td>
</tr>
<tr>
<td>------------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>Kentucky</td>
<td>4,339,367</td>
<td>1.41 %</td>
</tr>
<tr>
<td>Oregon</td>
<td>3,831,074</td>
<td>1.24 %</td>
</tr>
<tr>
<td>Oklahoma</td>
<td>3,751,351</td>
<td>1.22 %</td>
</tr>
<tr>
<td>Connecticut</td>
<td>3,574,097</td>
<td>1.16 %</td>
</tr>
<tr>
<td>Iowa</td>
<td>3,046,355</td>
<td>.99 %</td>
</tr>
<tr>
<td>Mississippi</td>
<td>2,967,297</td>
<td>.96 %</td>
</tr>
<tr>
<td>Arkansas</td>
<td>2,915,918</td>
<td>.95 %</td>
</tr>
<tr>
<td>Kansas</td>
<td>2,853,118</td>
<td>.93 %</td>
</tr>
<tr>
<td>Utah</td>
<td>2,763,885</td>
<td>.90 %</td>
</tr>
<tr>
<td>Nevada</td>
<td>2,700,551</td>
<td>.88 %</td>
</tr>
<tr>
<td>New Mexico</td>
<td>2,059,179</td>
<td>.67 %</td>
</tr>
<tr>
<td>West Virginia</td>
<td>1,852,994</td>
<td>.60 %</td>
</tr>
<tr>
<td>Nebraska</td>
<td>1,826,341</td>
<td>.59 %</td>
</tr>
<tr>
<td>Idaho</td>
<td>1,567,582</td>
<td>.51 %</td>
</tr>
<tr>
<td>Hawaii</td>
<td>1,360,301</td>
<td>.44 %</td>
</tr>
<tr>
<td>Maine</td>
<td>1,328,361</td>
<td>.43 %</td>
</tr>
<tr>
<td>New Hampshire</td>
<td>1,316,470</td>
<td>.43 %</td>
</tr>
<tr>
<td>Rhode Island</td>
<td>1,052,567</td>
<td>.34 %</td>
</tr>
<tr>
<td>Montana</td>
<td>989,415</td>
<td>.32 %</td>
</tr>
<tr>
<td>Delaware</td>
<td>897,934</td>
<td>.29 %</td>
</tr>
<tr>
<td>South Dakota</td>
<td>814,180</td>
<td>.26 %</td>
</tr>
<tr>
<td>Alaska</td>
<td>710,231</td>
<td>.23 %</td>
</tr>
<tr>
<td>North Dakota</td>
<td>672,591</td>
<td>.22 %</td>
</tr>
<tr>
<td>Vermont</td>
<td>625,741</td>
<td>.20 %</td>
</tr>
<tr>
<td>Wyoming</td>
<td>563,626</td>
<td>.18 %</td>
</tr>
</tbody>
</table>

**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**  
*State Populations (2010)*

**Description**
- A dataset containing U.S. state populations.

**Usage**
```r
data(state_populations)
```

**Format**
- A data frame with 50 rows and 3 variables

**Details**
- State. The 50 U.S. states.
- Proportion. Proportion of total U.S. population.

**References**

---

**string**  
*Generate Random Vector of Strings*

**Description**
- Generate a random vector of strings.

**Usage**
```r
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 `r_data_frame` or `r_list`.
- `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 `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 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 sapply(x,class). When multiple types are present these are collapsed. By default the \n is used.
Details

By default column 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

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 \texttt{r\_data\_frame} or \texttt{r\_list}.
x A vector of elements to chose from.
prob A vector of probabilities to chose from.
random logical. If \texttt{TRUE} the times are randomized, otherwise the times are sequential.
name The name to assign to the output vector’s \texttt{varname} attribute. This is used to auto assign names to the column/vector name when used inside of \texttt{r\_data\_frame} or \texttt{r\_list}.

Value

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

See Also

times

Examples

\begin{verbatim}
  time_stamp(20)
  time_stamp(20, random=TRUE)
  pie(table(time_stamp(2000, x = seq(0, 23, by = 2), prob = probs(12))))
\end{verbatim}

---

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 r_data_frame or r_list.

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 r_data_frame or r_list.

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 r_data_frame or r_list.
- **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 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**  

*Add Internal Name to Vector*

**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., attributes(x)["varname"]).
- `name`: A name to assign to attributes(x)["varname"].

**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**  

*Generate Random Data Sets*

**Description**

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

Generate Random Vector of Zip Codes

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 r_data_frame or r_list.
k  The number of the elements of x to sample from (uses sample(x,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 r_data_frame or r_list.

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(1000, prob = probs(10))))
Index

* age
  age, 4
* animal
  animal, 5
* answer
  answer, 6
* area
  area, 7
* army
  military, 42
* birth
  dob, 17
* branch
  military, 42
* browser
  internet_browser, 33
* capitals
  upper, 86
* car
  car, 9
* character
  lorem_ipsum, 40
  string, 83
* children
  children, 10
* class
  table_heat, 84
* coin
  coin, 11
* color
  color, 12
* correlate
  relate, 54
* cut
  interval, 34
* datasets
  animal_list, 6
  grady_augmented, 25
  languages, 37
  name_neutral, 46
  presidential.debates_2012, 51
  state_populations, 83
* date
  date_stamp, 13
* death
  death, 14
* democrat
  political, 50
* dice
  dice, 15
* died
  death, 14
* divorce
  marital, 41
* dna
  dna, 16
* dob
  dob, 17
* dummy
  r_dummy, 60
* education
  education, 19
* employment
  employment, 20
* eye
  eye, 22
* factor
  r_sample_ordered, 68
* false
  valid, 88
* gender
  sex, 75
  sex_inclusive, 76
* gpa
  grade, 23
* grade
  grade, 23
  grade_level, 24
* group
  group, 26
* hair
  hair, 27
* head
  coin, 11
* height
  height, 28
* hour
  hour, 30
* identification
  id, 31
* id
  id, 31
* income
  income, 32
* insert
  r_insert, 61
* integer
  as_integer, 8
* intelligence
  iq, 35
* interval
  interval, 34
* iq
  iq, 35
* iris
  eye, 22
* language
  language, 36
* letters
  upper, 86
* level
  level, 38
* likert
  likert, 39
* list
  r_list, 62
* logical
  valid, 88
* lower
  upper, 86
* marines
  military, 42
* marital
  marital, 41
* married
  marital, 41
* military
  military, 42
* minute
  minute, 43
* missing
  r_na, 63
* month
  month, 44
* name
  name, 45
* navy
  military, 42
* na
  r_na, 63
* normal
  normal, 47
* no
  answer, 6
* numeric
  as_integer, 8
* ordered
  r_sample_ordered, 68
* percent
  probs, 52
* pet
  animal, 5
* political
  political, 50
* politics
  political, 50
* probability
  probs, 52
* race
  race, 53
* related
  relate, 54
* religion
  religion, 55
* republican
  political, 50
* responses
  likert, 39
* sat
  sat, 72
* second
  second, 73
* sentence
  sentence, 74
INDEX

* sex
  sex, 75
  sex_inclusive, 76
* smoking
  smokes, 78
* speed
  speed, 79
* state
  state, 80
* string
  lorem_ipsum, 40
  string, 83
* tail
  coin, 11
* time
  hour, 30
  minute, 43
  second, 73
  time_stamp, 85
* true
  valid, 88
* type
  variables, 89
* upper
  upper, 86
* valid
  valid, 88
* variable functions
  age, 4
  animal, 5
  answer, 6
  area, 7
  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, 22
  grade, 23
  grade_level, 24
  group, 26
  hair, 27
  height, 28
  income, 32
  internet_browser, 33
  iq, 35
  language, 36
  level, 38
  likert, 39
  lorem_ipsum, 40
  marital, 41
  military, 42
  month, 44
  name, 45
  normal, 47
  political, 50
  race, 53
  religion, 55
  sat, 72
  sentence, 74
  sex, 75
  sex_inclusive, 76
  smokes, 78
  speed, 79
  state, 80
  string, 83
  upper, 86
  valid, 88
  year, 91
  zip_code, 92
* widow
  marital, 41
* year
  year, 91
* yes
  answer, 6
* zip_code
  zip_code, 92

age, 4, 6–12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54–56, 58, 61, 62, 70, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
animal, 4, 5, 7–12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54–56, 58, 61, 62, 70, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
animal_list, 6
answer, 4, 6, 6, 8–12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46,
96

48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
area, 4, 6, 7, 7, 9–12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
as.integer, 8

birth (dob), 17
car, 4, 6–8, 9, 10–12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
color, 4, 6–11, 12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
class, 8
coin, 4, 6–10, 11, 12, 14–16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
cut, 34, 35
data.frame, 8, 48, 58, 61, 62, 64, 75, 84
date_stamp, 4, 6–12, 13, 15, 16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
dead (death), 14
dna, 4, 6–12, 14, 15, 16, 18–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
dob, 4, 6–12, 14–16, 17, 19–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
dummy, 4, 6–12, 14–16, 18, 18, 20–22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
ele(ela), 38
employment, 4, 6–12, 14–16, 18–20, 20, 22, 24, 25, 27–29, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
eye, 4, 6–12, 14–16, 18–21, 22, 24, 25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
factor, 8, 31

gamma, 32
gender (sex), 75
gender_inclusive (sex_inclusive), 76
gpa (grade), 23
grade, 4, 6–12, 14–16, 18–21, 23, 23, 25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
grade_letter (grade), 23
grade_level, 4, 6–12, 14–16, 18–21, 23, 24, 24, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
grade_augmented, 25
group, 4, 6–12, 14–16, 18–21, 23–25, 26, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92

head, 49
height, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 28, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
height_cm(height), 28
height_in(height), 28
hour, 30

id, 31
id_factor(id), 31
income, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
internet_browser, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
interval, 34
iq, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
language, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
languages, 37
level, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 38, 40–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
likert, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39, 39, 41–43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
likert_5(likert), 39
likert_7(likert), 39
list, 62, 64, 89
lorem_ipsum, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39, 40–40, 42, 43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
lower, 86
lower_factor, 86
marital, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–41, 41, 43, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
math(level), 38
matrix, 89
military, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–42, 42, 45, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
minute, 43
month, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 44, 46, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
mtcars, 9
name, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 45, 48, 51, 54, 56, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
name_neutral, 46
normal, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 47, 51, 54, 57, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
normal_round(normal), 47
ordered, 69
package-wakefield (wakefield), 90
paragraph (lorem_ipsum), 40
peek, 48
pet (animal), 5
plot.tbl_df, 49
political, 4, 6–12, 14–16, 18–21, 23–25, 27, 28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48, 50, 54, 57, 73, 74, 76, 78–80, 82, 84, 87, 88, 91, 92
presidential.debates_2012, 36, 51, 74
primary (color), 12
print.available, 51
print.variable, 52
probs, 52
r.data, 57
r.data.frame, 57, 58, 58, 61, 63, 71, 89
r.data.theme (r.data), 57
r.dummy, 59, 60, 63, 71
r.insert, 61
r.list, 59, 61, 62, 71, 89
INDEX

r_na, 63
r_sample, 64
r_sample_binary, 65
r_sample_binary_factor
  (r_sample_binary), 65
r_sample_factor, 66
r_sample_integer, 67
r_sample_logical, 68
r_sample_ordered, 68
r_sample_replace, 69
r_series, 65–70
race, 6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 53, 57, 73, 74, 76, 78–80, 82, 84,
  87, 88, 91, 92
relate, 54, 70
religion, 4–6–12, 14–16, 18–21, 23–25, 27,
  28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  48, 51, 54, 55, 73, 74, 76, 78–80, 82,
  84, 87, 88, 91, 92
rnorm, 47, 48
round, 23, 29, 32, 35, 47, 48, 72, 80
sample, 65–70
sample.int, 19, 66
sat, 6–12, 14–16, 18–21, 23–25, 27, 28, 30,
  32, 34, 36, 37, 39–43, 45, 46, 48, 51,
  54, 57, 72, 74, 76, 78–80, 82, 84, 87,
  88, 91, 92
scale_fill_brewer, 84
second, 73
sentence, 6–12, 14–16, 18–21, 23–25, 27,
  28, 30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  48, 51, 54, 57, 73, 74, 76, 78–80, 82,
  84, 87, 88, 91, 92
seq.Date, 13, 14, 17
seriesname, 61, 75
sex, 4–6–12, 14–16, 18–21, 23–25, 27, 28, 30,
  32, 34, 36, 37, 39–43, 45, 46, 48, 51,
  54, 57, 73, 74, 75, 78–80, 82, 84, 87,
  88, 91, 92
sex_inclusive, 4–6–12, 14–16, 18–21,
  23–25, 27, 28, 30, 32, 34, 36, 37,
  39–43, 45, 46, 48, 51, 54, 57, 73, 74,
  76, 76, 79, 80, 82, 84, 87, 88, 91, 92
smokes, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78, 80, 82,
  84, 87, 88, 91, 92
speed, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78, 79, 79, 82,
  84, 87, 88, 91, 92
speed_kph(speed), 79
speed_mph(speed), 79
sprintf, 31
state, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78–80, 82, 84,
  87, 88, 91, 92
state_populations, 83
stri_rand_lipsum, 41
stri_rand_strings, 84
string, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78–80, 82, 83,
  87, 88, 91, 92
table_heat, 49, 84
tbl_df, 55, 58, 61, 70, 71
time_stamp, 85	imes

times, 30, 44, 74, 86
upper, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78–80, 82, 84,
  86, 88, 91, 92
upper_factor (upper), 86
valid, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78–80, 82, 84,
  87, 88, 91, 92
variables, 89
varname, 90
wakefield, 90
year, 4–6–12, 14–16, 18–21, 23–25, 27, 28,
  30, 32, 34, 36, 37, 39–43, 45, 46, 48,
  51, 54, 57, 73, 74, 76, 78–80, 82, 84,
  87, 88, 91, 92
zip_code, 4–6–12, 14–16, 18–21, 23–25, 27,
  28, 30, 32, 34, 36, 37, 39–43, 45, 46,
  48, 51, 54, 57, 73, 74, 76, 78–80, 82,
  84, 87, 88, 91, 92