Package 'schoolmath'

February 20, 2015

Type Package

Title Functions and datasets for math used in school

Version 0.4

Date 2009-10-26

Author Joerg Schlarmann, Josef Wienand

Maintainer Joerg Schlarmann <schlarmann@produnis.de>

Depends R (>= 2.10)

Description This package contains functions and datasets for math taught in school. A main focus is set to prime-calculation

License GPL (>= 2)

Repository CRAN

Date/Publication 2012-07-23 10:35:33

NeedsCompilation no

R topics documented:

`schoolmath-package` *Functions and datasets for math used in school*

Description

This package contains functions and datasets for math taught in school. A main focus is set to prime-calculation

Details

Package:	schoolmath
Type:	Package
Version:	0.4
Date:	2009-10-26
License:	GPL version 2 or newer

Author(s)

Joerg Schlarmann, Josef Wienand

Maintainer: Joerg Schlarmann <schlarmann@produnis.de>

`cancel.fraction` *cancel fractions to their simplest form*

Description

This function cancels a fraction to its simplest form, using greatest common divisor

Usage

```
cancel.fraction(numerator, denominator)
```

Arguments

`numerator`	fraction's numerator
`denominator`	fraction's denominator

Author(s)

Joerg Schlarmann

See Also

gcd

Examples

```
## to cancel fraction 42/56 type:
cancel.fraction(42, 56)
```

`decimal2fraction` *convert a decimal-number into fraction*

Description

This function converts a decimal number into a fraction

Usage

```
decimal2fraction(decimal, period = 0)
```

Arguments

`decimal`	the decimal number to be converted, given without an repeating ending
`period`	if the decimal places have an repeating ending (period), set the period here. See examples.

Author(s)

Joerg Schlarmann

Examples

```
## converting 23.4323
decimal2fraction(23.4323)

## converting a number with decimal period, e.g. 12.12344444444444444444
decimal2fraction(12.123, 4)
```

gcd *Greatest common divisor of two numbers*

Description

This function gives the greatest common divisor of two numbers

Usage

```
gcd(x, y)
```

Arguments

x	first number
y	second number

Author(s)

Joerg Schlarmann

See Also

scm

Examples

```
gcd(42, 56)
```

is.decimal *check wether a vector contains numbers with decimal places*

Description

This function checks, wether a vector contains numbers with decimal places. It returns TRUE or FALSE

Usage

```
is.decimal(x)
```

Arguments

x	a number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

`is.whole`

Examples

```
is.decimal(3)    # this will return FALSE
is.decimal(2.01) # this will return TRUE

x <- c(1,2,3,4,5.5, 6.03, 23.07)
is.decimal(x)
```

`is.even` *check wether numbers of a vector are even*

Description

This function checks wether the numbers of a vector are even. It returns TRUE/FALSE

Usage

```
is.even(x)
```

Arguments

`x`	A number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

`is.odd`

Examples

```
is.even(3)  # this will return FALSE
is.even(2)  # this will return TRUE

x <- c(1,2,3,4,5, 6, 7)
is.even(x)
```

is.negative *check wether numbers of a vector are negative*

Description

This function checks wether the numbers of a vector are negative. It returns TRUE/FALSE

Usage

```
is.negative(x)
```

Arguments

x	A number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

is.positive is.real.positive

Examples

```
is.negative(3)  # this will return FALSE
is.negative(-2)  # this will return TRUE

x <- c(-1, -2, 3.02, 4, -5.2, 6, -7)
is.negative(x)
```

is.odd *check wether numbers of a vector are odd*

Description

This function checks wether the numbers of a vector are odd. It returns TRUE/FALSE

Usage

```
is.odd(x)
```

Arguments

x	A number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

is.even

Examples

```
is.odd(2)  # this will return FALSE
is.odd(3)  # this will return TRUE

x <- c(1,2,3,4,5, 6, 7)
is.odd(x)
```

---

is.positive *check wether numbers of a vector are positive*

Description

This function checks wether the numbers of a vector are positive. It returns TRUE/FALSE.

Usage

```
is.positive(x)
```

Arguments

x	A number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

is.negative is.real.positive

Examples

```
is.positive(-3)   # this will return FALSE
is.positive(2)  # this will return TRUE

x <- c(-1, -2, 3.02, 4, -5.2, 6, -7)
is.positive(x)
```

is.prim *check wether a vector contains prime-numbers*

Description

This function checks, wether a vector contains prime-numbers. It returns TRUE or FALSE

Usage

```
is.prim(y)
```

Arguments

y	a number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

primes

Examples

```
is.prim(8)  # this will return FALSE
is.prim(11) # this will return TRUE

x <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11)
is.prim(x)
```

is.real.positive *check wether numbers of a vector are real positive*

Description

This function checks wether the numbers of a vector are positive. It returns TRUE/FALSE. Real positive means, that zero is included as a positive number.

Usage

```
is.real.positive(x)
```

Arguments

x	A number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

is.negative is.positive

Examples

```
is.real.positive(-3)   # this will return FALSE
is.real.positive(0)    # this will return TRUE

x <- c(0, -1, -2, 3.02, 4, -5.2, 6, -7)
is.real.positive(x)
```

---

is.whole *check wether a vector contains numbers with decimal places*

Description

This function checks, wether a vector contains whole numbers without decimal places. It returns TRUE or FALSE

Usage

```
is.whole(x)
```

Arguments

x	a number or vector to be checked

Author(s)

Joerg Schlarmann

See Also

is.decimal

Examples

```
is.whole(3.12)  # this will return FALSE
is.whole(2)     # this will return TRUE

x <- c(1, 2, 3, 4, 5.5, 6.03, 23.07)
is.whole(x)
```

`prime.factor` *giving prime-factors of a number*

Description

This function calculates the prime-factors of a number

Usage

```
prime.factor(n)
```

Arguments

`n`	the number to be checked

Author(s)

Joerg Schlarmann

Examples

```
prime.factor(21)
prime.factor(100)
```

`primes` *generate prime-numbers*

Description

This function generates prime-numbers, which can be found between a start- and an end-number.

Usage

```
primes(start = 12, end = 9999)
```

Arguments

start	start-number
end	end-number

Author(s)

Joerg Schlarmann

See Also

is.prim

Examples

```
primes(12,150) # list prime-numbers between 12 and 150
```

---

primlist *prime-numbers between 1 and 9999999*

Description

This is a list of prime-numbers between 1 and 9999999

Usage

```
data(primlist)
```

Format

a vector containing a list of prime-numbers between 1 and 9999999

---

scm *calculating the smallest common multiple of two numbers*

Description

This function calculates the smallest common multiple (least common multiple) of two numbers

Usage

```
scm(x, y)
```

Arguments

x	first number
y	second number

Author(s)

Joerg Schlarmann

See Also

gcd

Examples

```
scm(3528, 3780)
```

Index