Package 'Necklaces'

January 20, 2025

Type Package

Title Necklaces and Bracelets

Version 1.0

Date 2022-08-07

Author Elvira Di Nardo [aut, cph],
Giuseppe Guarino [aut, cre, cph]

Maintainer Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

Description Tools to generate Necklaces, Bracelets, Lyndon words and de Bruijn sequences. The generation relies on integer partitions and uses the 'KStatistics' package. Methods used in the package refers to E. Di Nardo and G. Guarino (2022) <arXiv:2208.06855>.

License GPL

Imports kStatistics

NeedsCompilation no

Repository CRAN

Date/Publication 2022-08-19 12:30:08 UTC

Contents

`Necklaces-package` *Necklaces and Bracelets*

Description

Tools to generate Necklaces, Bracelets, Lyndon words and de Bruijn sequences. The generation relies on integer partitions and uses the 'KStatistics' package. Methods used in the package refers to E. Di Nardo and G. Guarino (2022) <arXiv:2208.06855>.

Details

Using multi-index compositions, necklaces and bracelets are generated as well as Lyndon words and de Bruijn sequences. For multi-index compositions, this package refers to the `kStatistics` package.

Author(s)

Elvira Di Nardo <`elvira.dinardo@unito.it`>,
Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

Elvira Di Nardo [aut, cph], Giuseppe Guarino [aut, cre, cph]

Maintainer: Giuseppe Guarino <giuseppe.guarino@rete.basilicata.it>

References

Di Nardo, E. (2014) On a symbolic representation of non-central Wishart random matrices with applications. Jour. Mult. Anal. Vol.125, 121–135. (`https://arxiv.org/abs/1312.4395`)

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (`https://arxiv.org/abs/2206.15348`)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (`https://arxiv.org/abs/2208.06855`)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

Examples

```
# Sort the following list [2,2,3],[3,2,3],[1,2,3]
#
lSort(list(c(2,2,3),c(3,2,3),c(1,2,3)))

# Generate the elements of the necklace in equivalence relation with
# the input vector c(1,0,2,1)
cNecklaces(c(1,0,2,1))

# The previous result in a compact form
cNecklaces(c(1,0,2,1),TRUE)

# Generate the elements of the bracelet in equivalence relation with
# the input vector (1,0,2,1)
cBracelets(c(1,0,2,1))

# The previous result in a compact form
cBracelets(c(1,0,2,1),TRUE)

# Generate all the necklaces of the configuration (2,1,1)
# corresponding to the vector (1,1,2,3)
fNecklaces(c(2,1,1))

# The previous result in a compact form
fNecklaces(c(2,1,1),TRUE)

# The first value of the alphabet is set equal to zero
fNecklaces(c(2,1,1),TRUE,0)

# Generate all the bracelets of the configuration (2,1,1)
# corresponding to the vector (1,1,2,3)
fBracelets(c(2,1,1))

# The previous result in a compact form
fBracelets(c(2,1,1),TRUE)

# The first value of the alphabet is set equal to zero
fBracelets(c(2,1,1),TRUE,0)

# Generate the list of all the representatives of all the necklaces
# of length 4 over the alphabet {1,2}.
Necklaces(4,2)

# Generate the list of  all the representatives of all the necklaces
# of length 5 over the alphabet {1,2,3}.
Necklaces(5,3)

# Generate the list of  all the representatives of all the necklaces
# of length 5 over the alphabet {0,1,2}.
Necklaces(5,3,0)

# Generate the list of all the representatives of all the bracelets
# of length 4 over the alphabet {1,2}.
Bracelets(4,2)

# Generate the list of  all the representatives of all the bracelets
# of length 5 over the alphabet {1,2,3}.
Bracelets(5,3)

# Generate the list of  all the representatives of all the bracelets
# of length 5 over the alphabet {0,1,2}.
Bracelets(5,3,0)

# Generate all the Lyndon words of length 5 over the alphabet
# {1,2}
LyndonW(5)
```

```
# or equivalently
LyndonW(5,2)

# The previous result in a compact form
LyndonW(5,2,TRUE)

# Generate all the Lyndon words of length 5 over the alphabet
# {0,1}
LyndonW(5,2,TRUE,0)

# Generate the de Bruijn sequence of length 4 on the binary alphabet
# {0,1}
sBruijn(4)
# or equivalently
sBruijn(4,2)

# Generate the de Bruijn sequence of length 2 over the alphabet {0,1,2}
sBruijn(2,3)

# Generate the de Bruijn sequence of length 2 over the alphabet {1,2,3}
sBruijn(2,3,1)

# Generate the de Bruijn sequence of length 2 over the alphabet {1,2,3}
# with a block separator.
sBruijn(2,3,1,TRUE)
```

Bracelets *Bracelets*

Description

The function generates all the representatives of all the bracelets of length `n` over an alphabet of `m` consecutive non-negative integers.

Usage

```
Bracelets(n=1, m=1, fn=1)
```

Arguments

`n`	positive integer: the length of the representatives
`m`	positive integer: the number of consecutive non-negative integers in the alphabet
`fn`	integer: the first value of the alphabet, the default is 1

Details

The function generates the list of all representatives of all bracelets having a fixed length `n` on the same alphabet, by default `{1,2,...,m}`. The main block function is the `fBracelets` function of the `Necklaces` package, which is called repeatedly. The input parameters of the `fBracelets` function are generated by using the `mKT` function of the `kStatistics` package. Indeed, given a multi-index `v`, that is a vector of non-negative integers, and a positive integer `n`, the `mKT` function returns all the lists `(v1,...,vn)` of non-negative integer vectors, having the same length of the multi-index `v` and such that `v=v1+...+vn`. Here, the `mKT` function is used with the input vector having length `1` as well as the output vectors `v1,...,vn`, corresponding to the partitions of an integer with a fixed number of parts. As example, the `mKT` function with input `(3,3)` generates the following result:

```
[( 1 )( 1 )( 1 )]
[( 0 )( 1 )( 2 )]
[( 1 )( 0 )( 2 )]
[( 1 )( 2 )( 0 )]
[( 0 )( 2 )( 1 )]
[( 2 )( 0 )( 1 )]
[( 2 )( 1 )( 0 )]
[( 0 )( 3 )( 0 )]
[( 3 )( 0 )( 0 )]
[( 0 )( 0 )( 3 )]
```

Each vector is a possible configuration and then passed to the `fBracelets` function to recover the corresponding bracelet. For example

- the configuration `[( 1 )( 1 )( 1 )]` denotes the vector `(1,2,3)`; calling `fBracelets(c(1,1,1))`, the representative `[1 2 3]` is generated;
- the configuration `[( 0 )( 1 )( 2 )]` denotes the vector `(2,3,3)`; calling `fBracelets(c(0,1,2))`, the representative `[2 3 3]` is generated;
- the configuration `[( 1 )( 0 )( 2 )]` denotes the vector `(1,3,3)`; calling `fBracelets(c(1,0,2))`, the representative `[1 3 3]` is generated;

and so on. As last step, the union of all the outputs gives the expected result:

```
[ 1 1 1 ], [ 1 1 2 ], [ 1 1 3 ], [ 1 2 2 ], [ 1 2 3 ], [ 1 3 3 ], [ 2 2 2 ], [ 2 2 3 ], [ 2 3 3 ], [ 3 3 3 ]
```

that are all the representatives of bracelets of length `3` on the alphabet `{1,2,3}`.

Note: Comparing this example with the one given in the description of the `fNecklaces` function, `[1 3 2]` is missed since it is in the class of the bracelet `[1 2 3] = {(1 2 3),(1 3 2),(2 1 3),(2 3 1),(3 1 2),(3 2 1)}` obtained running `cBracelets(c(1,2,3))`.

Value

list	the list containing all the representatives of all the bracelets of length n over an alphabet of m consecutive non-negative integers.

Note

The function calls the fBracelets function in the Necklaces package and the mKT function in the kStatistics package.

Author(s)

Elvira Di Nardo <elvira.dinardo@unito.it>,
Giuseppe Guarino <giuseppe.guarino@rete.basilicata.it>

References

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (https://arxiv.org/abs/2206.15348)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (https://arxiv.org/abs/2208.06855)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

fBracelets, LyndonW, sBruijn

Examples

```
# Generate the list of all the representatives of all the bracelets
# of length 4 over the alphabet {1,2}.
Bracelets(4,2)

# Generate the list of  all the representatives of all the bracelets
# of length 5 over the alphabet {1,2,3}.
Bracelets(5,3)

# Generate the list of  all the representatives of all the bracelets
# of length 5 over the alphabet {0,1,2}.
Bracelets(5,3,0)
```

`cBracelets` *Elements in a bracelet*

Description

The function generates the elements of a bracelet in equivalence relation with the vector given in input.

Usage

```
cBracelets(v=c(), bOut=FALSE)
```

Arguments

`v`	vector: input vector
`bOut`	boolean: if `TRUE`, the function produces a compact result

Details

The function generates the elements of a bracelet which are in equivalence relation with the vector given in input. The first parameter is the input vector. If the second parameter (`bOut`) is set equal to `TRUE`, the function produces a compact result.

Example: `cBracelets(c(1,0,2,1))` produces the following result:

```
[1] 0 1 1 2
[1] 0 2 1 1
[1] 1 0 2 1
[1] 1 1 0 2
[1] 1 1 2 0
[1] 1 2 0 1
[1] 2 0 1 1
[1] 2 1 1 0
```

`cBracelets(c(1,0,2,1),TRUE)` produces the following result:

```
[ 0 1 1 2 ] ( 1 )
[ 0 2 1 1 ] ( 2 )
[ 1 0 2 1 ] ( 3 )
[ 1 1 0 2 ] ( 4 )
[ 1 1 2 0 ] ( 5 )
[ 1 2 0 1 ] ( 6 )
[ 2 0 1 1 ] ( 7 )
[ 2 1 1 0 ] ( 8 )
```

Note that `0 1 1 2` is the representative of the class, that is the minimum in lexicographical order.

Value

`list`	the list containing all the elements of the bracelet in equivalence relation with the vector given in input

Note

The function is called from the `fBracelets` function in the `Necklaces` package.

Author(s)

Elvira Di Nardo `<elvira.dinardo@unito.it>`,
Giuseppe Guarino `<giuseppe.guarino@rete.basilicata.it>`

References

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (`https://arxiv.org/abs/2206.15348`)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (`https://arxiv.org/abs/2208.06855`)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`fBracelets`

Examples

```
# Generate the elements of the bracelet in equivalence relation with
# the input vector (1,0,2,1)
cBracelets(c(1,0,2,1))

# The previous result in a compact form
cBracelets(c(1,0,2,1),TRUE)
```

cNecklaces — *Elements in a necklace*

Description

The function generates the elements of a necklace in equivalence relation with the vector given in input.

Usage

```
cNecklaces(v=c(), bOut=FALSE)
```

Arguments

`v`	vector: input vector
`bOut`	boolean: if `TRUE`, the function produces a compact result

Details

The function generates the elements of a necklace which are in equivalence relation with the vector given in input. The first parameter is the input vector. If the second parameter (`bOut`) is set equal to `TRUE`, the function produces a compact result.

Example: `cNecklaces(c(1,0,2,1))` produces the following result:

```
[1] 0 2 1 1
[1] 1 0 2 1
[1] 1 1 0 2
[1] 2 1 1 0
```

`cNecklaces(c(1,0,2,1),TRUE)` produces the following result:

```
[ 0 2 1 1 ] ( 1 )
[ 1 0 2 1 ] ( 2 )
[ 1 1 0 2 ] ( 3 )
[ 2 1 1 0 ] ( 4 )
```

Note that `0 2 1 1` is the representative of the class, that is the minimum in lexicographical order.

Value

`list`	the list containing all the elements of the necklace in equivalence relation with the vector given in input

Note

The function is called from the `fNecklaces`, `sBruijn`, `cBracelets` functions in the `Necklaces` package.

Author(s)

Elvira Di Nardo <`elvira.dinardo@unito.it`>,
Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

References

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (`https://arxiv.org/abs/2206.15348`)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (`https://arxiv.org/abs/2208.06855`)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`fNecklaces`, `sBruijn`, `cBracelets`

Examples

```
# Generate the elements of the necklace in equivalence relation with
# the input vector c(1,0,2,1)
cNecklaces(c(1,0,2,1))

# The previous result in a compact form
cNecklaces(c(1,0,2,1),TRUE)
```

`fBracelets` *Bracelets of a fixed configuration*

Description

The function generates all the representatives of the bracelets corresponding to a fixed configuration.

Usage

```
fBracelets(pv=c(), bOut=FALSE, fn=1)
```

Arguments

`pv`	vector: the fixed configuration
`bOut`	boolean: if `TRUE`, the function produces a compact result
`fn`	integer: the first value of the alphabet, the default is 1

Details

The function generates all the representatives of the bracelets corresponding to a fixed configuration. If the second parameter (`bOut`) is set equal to `TRUE`, the function produces a compact result. The third parameter (`fn`) initializes the first value of the alphabet, which by default is equal to 1. For example, to generate all the representatives of the bracelets corresponding to the fixed configuration `(2,1,1)`, run `fBracelets(c(2,1,1))`. In such a case the alphabet is `{1,2,3}`. Using the `nPerm` function of the `kStatistics` package, the function first generates all the permutations of the vector `(1,1,2,3)` corresponding to the configuration `(2,1,1)`, that is

`(I)   (3,2,1,1), (2,3,1,1), (3,1,1,2), ..., (1,1,2,3)` (12 in total)

Then the `cBracelets` function of the `Necklaces` package is called with input equal to each vector in `(I)`. For each obtained list, only the representative survives. At the end all the representatives of the bracelets are printed, that are `[1 1 2 3]`, `[1 2 1 3]`.

Value

`list`	the list containing all the representatives of the bracelets corresponding to a fixed configuration.

Note

The function calls the `cBracelets` function in the `Necklaces` package and the `nPerm` function in the `kStatistics` package.

Author(s)

Elvira Di Nardo <`elvira.dinardo@unito.it`>,
Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

References

Di Nardo, E. (2014) On a symbolic representation of non-central Wishart random matrices with applications. Jour. Mult. Anal. Vol.125, 121–135. (`https://arxiv.org/abs/1312.4395`)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (`https://arxiv.org/abs/2208.06855`)

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (`https://arxiv.org/abs/2206.15348`)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`cBracelets`, `LyndonW`, `sBruijn`

Examples

```
# Generate all the bracelets of the configuration (2,1,1)
# corresponding to the vector (1,1,2,3)
fBracelets(c(2,1,1))

# The previous result in a compact form
fBracelets(c(2,1,1),TRUE)

# The first value of the alphabet is set equal to zero
fBracelets(c(2,1,1),TRUE,0)
```

---

fNecklaces *Necklaces of a fixed configuration*

Description

The function generates all the representatives of the necklaces corresponding to a fixed configuration.

Usage

```
fNecklaces(pv=c(), bOut=FALSE, fn=1)
```

Arguments

`pv`	vector: the fixed configuration
`bOut`	boolean: if `TRUE`, the function produces a compact result
`fn`	integer: the first value of the alphabet, the default is 1

Details

The function generates all the representatives of the necklaces corresponding to a fixed configuration. If the second parameter (`bOut`) is set equal to `TRUE`, the function produces a compact result. The third parameter (`fn`) initializes the first value of the alphabet, which by default is equal to 1. For example, to generate all the representatives of necklaces corresponding to the fixed configuration `(2,1,1)`, run `fNecklaces(c(2,1,1))`. In such a case the alphabet is `{1,2,3}`. Using the `nPerm` function of the `kStatistics` package, the function first generates all the permutations of the vector `(1,1,2,3)` corresponding to the configuration `(2,1,1)`:

`(I)` `(3,2,1,1)`, `(2,3,1,1)`, `(3,1,1,2)`, `...`, `(1,1,2,3)` (12 in total)

Then the `cNecklaces` function of the `Necklaces` package is called with input equal to each vector in `(I)`. For each obtained list, only the representative survives. At the end all the representatives of the necklaces are printed, that are `[1 1 2 3]`, `[1 1 3 2]`, `[1 2 1 3]`.

Value

list	the list containing all the representatives of the necklaces corresponding to a fixed configuration.

Note

The function calls the `cNecklaces` function in the `Necklaces` package and the `nPerm` function in the `kStatistics` package.

Author(s)

Elvira Di Nardo <elvira.dinardo@unito.it>,
Giuseppe Guarino <giuseppe.guarino@rete.basilicata.it>

References

Di Nardo, E. (2014) On a symbolic representation of non-central Wishart random matrices with applications. Jour. Mult. Anal. Vol.125, 121–135. (https://arxiv.org/abs/1312.4395)

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (https://arxiv.org/abs/2206.15348)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (https://arxiv.org/abs/2208.06855)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`cBracelets`, `LyndonW`, `sBruijn`

Examples

```
# Generate all the necklaces of the configuration (2,1,1)
# corresponding to the vector (1,1,2,3)
fNecklaces(c(2,1,1))

# The previous result in a compact form
fNecklaces(c(2,1,1),TRUE)

# The first value of the alphabet is set equal to zero
fNecklaces(c(2,1,1),TRUE,0)
```

`lSort` *Sort a list of vectors*

Description

The function takes in input a list of vectors and returns the same list ordered in a lexicographical way.

Usage

```
lSort(pL = list())
```

Arguments

`pL`	list of vectors to be ordered

Details

The function takes as input a list of vectors and returns the same list ordered in a lexicographical way. For example if the input list is `(2,2,3),(3,2,3),(1,2,3)`, then the output of the function `lSort` produces the following result: `(1,2,3),(2,2,3),(3,2,3)`.

Value

`list`	the input list ordered in lexicographical way

Note

Called by the `cNecklaces`, `cBracelets`, `fNecklaces`, `Necklaces`, `Bracelets`, `LyndonW`, `sBruijn` functions in the `Necklaces` package.

Author(s)

Elvira Di Nardo <`elvira.dinardo@unito.it`>,
Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

References

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`cNecklaces`, `cBracelets`, `fNecklaces`, `Necklaces`, `Bracelets`, `LyndonW`, `sBruijn`

Examples

```
# Sort the following list (2,2,3),(3,2,3),(1,2,3)
#
lSort(list(c(2,2,3),c(3,2,3),c(1,2,3)))
```

`LyndonW` *Lyndon words*

Description

The function generates Lyndon words from necklaces of length `n` over an alphabet of `m` consecutive non-negative integers.

Usage

```
LyndonW(n=1, m=2, bOut=FALSE, fn=1)
```

Arguments

`n`	positive integer: the length of the representatives
`m`	positive integer: the number of consecutive non-negative integers in the alphabet
`bOut`	boolean: if `TRUE`, the function produces a compact result
`fn`	integer: the first value of the alphabet, the default is 1

Details

The function generates Lyndon words from necklaces of length `n` over an alphabet of `m` consecutive non-negative integers. The last parameter (`fn`) initializes the first value of the alphabet, which by default is equal to 1. If the parameter (`bOut`) is set equal to `TRUE`, the function produces a compact result. As example, running `LyndonW(5,2, TRUE,0)`, the function generates Lyndon words in compact form, from the binary necklaces of length 5, that are `[0 0 0 0 1]`, `[0 0 0 1 1]`, `[0 0 1 0 1]`,`[0 0 1 1 1]`, `[0 1 0 1 1]`, `[0 1 1 1 1]`.

Value

`list`	the list containing all the Lyndon words of length `n` over an alphabet of `m` consecutive non-negative integers.

Note

The function calls the `cNecklaces` and `lSort` functions in the `Necklaces` package.

Author(s)

Elvira Di Nardo <`elvira.dinardo@unito.it`>,
Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

References

Di Nardo, E. (2014) On a symbolic representation of non-central Wishart random matrices with applications. Jour. Mult. Anal. Vol.125, 121–135. (https://arxiv.org/abs/1312.4395)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (https://arxiv.org/abs/2208.06855)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`cNecklaces`, `sBruijn`

Examples

```
# Generate all the Lyndon words of length 5 over the alphabet
# {1,2}
LyndonW(5)
# or equivalently
LyndonW(5,2)

# The previous result in a compact form
LyndonW(5,2,TRUE)

# Generate all the Lyndon words of length 5 over the alphabet
# {0,1}
LyndonW(5,2,TRUE,0)
```

`Necklaces` *Necklaces*

Description

The function generates all the representatives of all the necklaces of length `n` over an alphabet of `m` consecutive non-negative integers.

Usage

```
Necklaces(n=1, m=1, fn=1)
```

Arguments

`n`	positive integer: the length of the representatives
`m`	positive integer: the number of consecutive non-negative integers in the alphabet
`fn`	integer: the first value of the alphabet, the default is 1

Details

The function generates the list of all representatives of all necklaces having a fixed length `n` on the same alphabet, by default `{1,2,...,m}`. The main block function is the `fNecklaces` function of the `Necklaces` package, which is called repeatedly. The input parameters of the `fNecklaces` function are generated by using the `mKT` function of the `kStatistics` package. Indeed, given a multi-index `v`, that is a vector of non-negative integers, and a positive integer `n`, the `mKT` function returns all the lists `(v1,...,vn)` of non-negative integer vectors, having the same length of the multi-index `v` and such that `v=v1+...+vn`. Here, the `mKT` function is used with the input vector having length `1` as well as the output vectors `v1,...,vn`, corresponding to the partitions of an integer with a fixed number of parts. As example, the `mKT` function with input `(3,3)` generates the following result:

```
[( 1 )( 1 )( 1 )]
[( 0 )( 1 )( 2 )]
[( 1 )( 0 )( 2 )]
[( 1 )( 2 )( 0 )]
[( 0 )( 2 )( 1 )]
[( 2 )( 0 )( 1 )]
[( 2 )( 1 )( 0 )]
[( 0 )( 3 )( 0 )]
[( 3 )( 0 )( 0 )]
[( 0 )( 0 )( 3 )]
```

Each vector is a possible configuration and then passed to the `fNecklaces` function to recover the corresponding necklace. For example

- the configuration `[( 1 )( 1 )( 1 )]` denotes the vector `[1,2,3]`; calling `fNecklaces(c(1,1,1))`, the representatives `[1,2,3]` and `[1,3,2]` are generated;
- the configuration `[( 0 )( 1 )( 2 )]` denotes the vector `[2,3,3]`; calling `fNecklaces(c(0,1,2))`, the reprsentative `[2,3,3]` is generated
- the configuration `[( 1 )( 0 )( 2 )]` denotes the vector `[1,3,3]`; calling `fNecklaces(c(1,0,2))`, the representative `[1,3,3]` is generated;

and so on. As last step, the union of all the outputs gives the expected result:

```
[ 1 1 1 ], [ 1 1 2 ], [ 1 1 3 ], [ 1 2 2 ], [ 1 2 3 ], [ 1 3 2 ], [ 1 3 3 ], [ 2 2 2 ], [ 2 2 3 ], [ 2 3 3 ], [ 3 3 3 ]
```

that are all the representatives of necklaces of length `3` on the alphabet `{1,2,3}`.

Value

`list`	the list containing all the representatives of all the necklaces of length `n` over an alphabet of `m` consecutive non-negative integers.

Note

The function calls the `fNecklaces` function in the `Necklaces` package and the `mKT` functions in the `kStatistics` package.

Author(s)

Elvira Di Nardo <elvira.dinardo@unito.it>,
Giuseppe Guarino <giuseppe.guarino@rete.basilicata.it>

References

Di Nardo, E., and Guarino., G. (2022) kStatistics: Unbiased Estimates of Joint Cumulant Products from the Multivariate Faa Di Bruno's Formula. The R journal - In press. (https://arxiv.org/abs/2206.15348)

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (https://arxiv.org/abs/2208.06855)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`fNecklaces`, `LyndonW`, `sBruijn`

Examples

```
# Generate the list of all the representatives of all the necklaces
# of length 4 over the alphabet {1,2}.
Necklaces(4,2)


# Generate the list of  all the representatives of all the necklaces
# of length 5 over the alphabet {1,2,3}.
Necklaces(5,3)


# Generate the list of  all the representatives of all the necklaces
# of length 5 over the alphabet {0,1,2}.
Necklaces(5,3,0)
```

`sBruijn` *The de Bruijn sequence*

Description

The function generates the (minimum) de Bruijn sequence of length `n` over an alphabet of `m` consecutive non-negative integers.

Usage

```
sBruijn(n=1,m=2, fn=0, bSep=FALSE)
```

Arguments

`n`	positive integer: the length of the representatives
`m`	positive integer: the number of consecutive non-negative integers in the alphabet
`fn`	integer: the first value of the alphabet, the default is 0
`bSep`	boolean: if `TRUE`, a separator is inserted between the output blocks

Details

The function generates the (minimum) de Bruijn sequence of order `n` over an alphabet of `m` consecutive non-negative integers. The parameter (`fn`) assigns the first value of the alphabet, which by default is equal to 0. If (`bSep`) is set equal to `TRUE`, a separator is inserted between the output blocks.

Value

`string`	the de Bruijn sequence

Note

The function calls the `Necklaces` `function and the` `cNecklaces` function in the `Necklaces` package.

Author(s)

Elvira Di Nardo <`elvira.dinardo@unito.it`>,
Giuseppe Guarino <`giuseppe.guarino@rete.basilicata.it`>

References

Di Nardo, E., and Guarino., G. (2022) Necklaces and bracelets in R - (`https://arxiv.org/abs/2208.06855`)

Flajolet, P., and Sedgewick, R. (2009) Analytic combinatorics. Cambridge University press.

See Also

`Necklaces`, `cNecklaces`, `LyndonW`, `sBruijn`

Examples

```
# Generate the de Bruijn sequence of length 4 on the binary alphabet
# {0,1}
sBruijn(4)
# or equivalently
sBruijn(4,2)

# Generate the de Bruijn sequence of length 2 over the alphabet {0,1,2}
```

```
sBruijn(2,3)

# Generate the de Bruijn sequence of length 2 over the alphabet {1,2,3}
sBruijn(2,3,1)

# Generate the de Bruijn sequence of length 2 over the alphabet {1,2,3}
# with a block separator.
sBruijn(2,3,1,TRUE)
```

Index