Package 'ciphertext'

December 20, 2024

Type Package

Title Classical Cryptography Methods for Words and Phrases

Version 0.1.0

Description Classical cryptography methods for words and brief phrases.
Substitution, transposition and concealment (null) ciphers are available, like
Caesar, Vigenère, Atbash, affine, simple substitution, Playfair,
rail fence, Scytale, single column, and Polybius ciphers.

License GPL-3

URL https://github.com/Luigi-Annic/ciphertext

BugReports https://github.com/Luigi-Annic/ciphertext/issues

Encoding UTF-8

Depends R (>= 4.3.0)

RoxygenNote 7.2.3

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Luigi Annicchiarico [cre, aut]

Maintainer Luigi Annicchiarico <luigi.annic@gmail.com>

Repository CRAN

Date/Publication 2024-12-20 10:40:05 UTC

Contents

`affine` *affine*

Description

The affine cipher is a monoalphabetic substitutoin cipher, where each letter is enciphered with the function (ax+b) mod 26 (26 is the number of letters in the alphabet)

Usage

```
affine(word, a, b, decrypt = FALSE)
```

Arguments

`word`	Word or phrase to be encrypted
`a`	First parameter. This value and 26 must be coprime
`b`	Second parameter. Magnitude of the shift
`decrypt`	If 'FALSE' (default), the program ciphers the input word, If 'TRUE', the program decrypts it.

Value

a string

References

https://en.wikipedia.org/wiki/Affine_cipher

Examples

```
affine("Hello", 1, -1)
```

atbash *atbash*

Description

The Atbash cipher is a type of monoalphabetic cipher which takes the alphabet and maps it to its reverse. It is a particular case of the affine cipher, with ‘a‘=‘b‘= (‘m‘-1). As ‘m‘ is the number of letters and is equal to 26, it means that ‘a‘ = ‘b‘ = 25. Encrypting and decrypting are not separate for this cipher.

Usage

```
atbash(word)
```

Arguments

word	Word or phrase to be encrypted

Value

a string

References

https://en.wikipedia.org/wiki/Atbash

Examples

```
atbash("abcxyz")
```

caesar *caesar*

Description

caesar encryption

Usage

```
caesar(word, key, decrypt = FALSE)
```

Arguments

word	Word or phrase to be encrypted
key	numeric key
decrypt	If ‘FALSE‘ (default), the program ciphers the input word, If ‘TRUE‘, the program decrypts it.

Value

a string

Examples

```
caesar("Hello", 1)
```

`nullcipher` *nullcipher*

Description

A null cipher is an encryption method where the plaintext is mixed with a large amount of non-cipher material (decoy).

Usage

```
nullcipher(phrase, index, decrypt = TRUE)
```

Arguments

`phrase`	Word or phrase to be decrypted
`index`	letter of interest for each word in the phrase. Also a pattern vector can be entered.
`decrypt`	Only Decryption is possible for now, but will be updated in the future

Value

a string

References

https://en.wikipedia.org/wiki/Null_cipher

Examples

```
nullcipher("handy set false posts", c(1,2,3))
```

playfair *playfair*

Description

The Playfair cipher is a symmetric method which encrypts pairs of letters using a modified Polybius square

Usage

```
playfair(word, key, added_letter = "x", decrypt = FALSE)
```

Arguments

word	Word or phrase to be encrypted or decrypted
key	Word for creating the modified Polybius square
added_letter	Letter to be added in case two letters of a pair are identical; usually "x" is used
decrypt	If 'FALSE' (default), the program ciphers the input word, If 'TRUE', the program decrypts it.

Value

a string

References

https://en.wikipedia.org/wiki/Playfair_cipher

Examples

```
playfair( "instruments", "monarchy", added_letter = "z")
playfair("gatlmzclrqtx", "monarchy", added_letter = "z", decrypt = TRUE)
```

polybius *polybius*

Description

The polybius square is a device which associates each letter to a pair of coordinates. The letter J is excluded and replaced with I in order to get 25 letters and create a 5x5 matrix.

Usage

```
polybius(input, decrypt = FALSE)
```

Arguments

input	Word or phrase to be encrypted, or character vector with the sequence of coordinate numbers if we need to decrypt
decrypt	If 'FALSE' (default), the program ciphers the input word, If 'TRUE', the program decrypts it.

Value

a string

References

https://en.wikipedia.org/wiki/Polybius_square

Examples

```
polybius("hello world")
polybius("23 15 31 31 34 52 34 42 31 14", decrypt = TRUE)
```

---

railfence *railfence*

Description

The rail fence is a transposition cipher where the text is written upwards and downwards diagonally (zigzag) on the rails of the fence

Usage

```
railfence(word, key)
```

Arguments

word	Word or phrase to be encrypted
key	numeric key (number of rails)

Value

a string

References

https://en.wikipedia.org/wiki/Rail_fence_cipher

Examples

```
railfence('we are discovered flee at once',3)
```

scytale *scytale*

Description

The Scytale is a transposition cipher The diameter of the Scytale (the number of turns) can be regarded as the key of the cipher.

Usage

```
scytale(word, key, decrypt = FALSE)
```

Arguments

word	Word or phrase to be encrypted or decrypted
key	Number of turns of the band
decrypt	If 'FALSE' (default), the program ciphers the input word, If 'TRUE', the program decrypts it.

Value

a string

References

https://en.wikipedia.org/wiki/Scytale

Examples

```
scytale('we are discovered flee at once',3)
```

simple_substitution *simple_substitution*

Description

simple substitution cipher. Each letter is monoalphabetically associated with a different one used for the encryption.

Usage

```
simple_substitution(word, key = "", seed = sample(1:1000, 1))
```

Arguments

word	Word or phrase to be encrypted
key	Word to be used as key for the encryption. If not provided, a random shuffle is performed
seed	Seed for reproducibility of the encryption if key is not provided

Value

a list with custom class "cipher", which modifies the printing defaults. The list contains the initial phrase (initial), the ciphered output (encrypted), and the alphabet order (keyalphabet)

Examples

```
simple_substitution("hello world", seed = 1234)
simple_substitution("hello world", key = "zebras")
```

---

singlecolumn *singlecolumn*

Description

In a columnar transposition cipher, the message is written out in rows of a fixed length, and then read out again column by column. The order of the column follows the alphabetcial order of the letters present in the key

Usage

```
singlecolumn(word, key, rm.blanks = TRUE)
```

Arguments

word	Word or phrase to be encrypted
key	word key: for example, the key "bcea" suggests that the column order is "2-3-4-1"
rm.blanks	Should spaces between words be removed? By default set to 'TRUE'

Value

a string

References

https://www.geeksforgeeks.org/columnar-transposition-cipher/

Examples

```
singlecolumn("This is wikipedia", "cipher")
```

vigenere *vigenere*

Description

Vigenère cipher is a method of encrypting alphabetic text where each letter of the plaintext is encoded with a different Caesar cipher, whose increment is determined by the corresponding letter the key

Usage

```
vigenere(word, key, decrypt = FALSE)
```

Arguments

word	Word or phrase to be encrypted
key	character key
decrypt	If 'FALSE' (default), the program ciphers the input word, If 'TRUE', the program decrypts it.

Value

a string

References

https://en.wikipedia.org/wiki/Vigen

Examples

```
vigenere("hello world", "opla")
```

Index