Project 1.2

Revision 0

Generated by Doxygen 1.9.1

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

Class Documentation

2.1 Hill Class Reference

```
#include <Hill.hpp>
```

Public Member Functions

- Hill ()
- Hill (const Matrix &K, bool encryption)
- Hill (const Matrix &E, const Matrix &D)
- Matrix getE () const
- Matrix getD () const
- bool setE (const Matrix &E)
- bool setD (const Matrix &D)
- std::string encrypt (const std::string &P) const
- std::string encrypt (const std::string &P, const Matrix &E)
- std::string decrypt (const std::string &C) const
- std::string decrypt (const std::string &C, const Matrix &D)
- bool kpa (const std::vector< std::string > &P, const std::vector< std::string > &C, unsigned int n)

2.1.1 Detailed Description

A C++ class to perform encryption/decryption and cryptanalysis using/of the Hill cipher with a 29 character alphabet.

2.1.2 Constructor & Destructor Documentation

2.1.2.1 Hill() [1/3]

```
Hill::Hill ( )
```

Default constructor. It should set the encryption key to {2,4,3,5} (2-by-2) and the decryption key to its inverse.

2.1.2.2 Hill() [2/3]

```
Hill::Hill (
            const Matrix & K,
            bool encryption )
```

Parameterized constructor. Use the parameter to set the encryption (E) and decryption (D) keys; if parameter is invalid then set E/D to a 0-by-0 matrix.

Parameters

K	- a matrix representing the encryption or decryption key.
encryption	- true if the key is the encryption key, false if the key is the decryption key

2.1.2.3 Hill() [3/3]

```
Hill::Hill (
            const Matrix & E,
            const Matrix & D )
```

Parameterized constructor. Use the parameters to set the encryption (E) and decryption (D) keys; if a parameter is invalid or inconsistent then set E/D to a 0-by-0 matrix.

Parameters

E	- encryption key.
D	- decryption key.

2.1.3 Member Function Documentation

2.1.3.1 decrypt() [1/2]

```
std::string Hill::decrypt (
            const std::string & C ) const
```

Decrypt the given ciphertext using the previous set decryption key, an empty string if the decryption key is invalid.

Parameters

C	- the ciphertext to decrypt

Returns

the plaintext resulting from decrypting the ciphertext using the stored decryption matrix.

2.1.3.2 decrypt() [2/2]

```
std::string Hill::decrypt (
            const std::string & C,
            const Matrix & D )
```

Decrypt the given ciphertext using the given decryption key, an empty string if the decryption key is invalid.

Parameters

C	- the plaintext to encrypt
D	- the key to use to decrypt the ciphertext

Returns

the plaintext resulting from decrypting the ciphertext using the given decryption matrix.

2.1.3.3 encrypt() [1/2]

```
std::string Hill::encrypt (
            const std::string & P ) const
```

Encrypt the given plaintext using the previous set encryption key, an empty string if the encryption key is invalid.

Parameters

P	- the plaintext to encrypt

Returns

the ciphertext resulting from encrypting the plaintext using the stored encryption matrix.

2.1.3.4 encrypt() [2/2]

```
std::string Hill::encrypt (
            const std::string & P,
            const Matrix & E )
```

Encrypt the given plaintext using the given encryption key, an empty string if the encryption key is invalid.

Parameters

P	- the plaintext to encrypt
E	- the key to use to encrypt the plaintext

Returns

the ciphertext resulting from encrypting the plaintext using the given encryption matrix.

2.1.3.5 getD()

```
Matrix Hill::getD ( ) const
```

Returns the current decryption key.

Returns

the decryption key (Matrix D), if no decryption key is set a 0-by-0 matrix.

2.1.3.6 getE()

```
Matrix Hill::getE ( ) const
```

Returns the current encryption key.

Returns

the encryption key (Matrix E), if no encryption key is set a 0-by-0 matrix.

2.1.3.7 kpa()

```
bool Hill::kpa (
             const std::vector< std::string > & P,
             const std::vector< std::string > & C,
             unsigned int n )
```

Mount a known-plaintext attack against the Hill cipher assuming an n-by-n encryption matrix. Set E/D to the encryption/decryption key if they can be recovered.

Parameters

P	- the plaintexts that correspond to C
C	- the ciphertexts that correspond to P

Returns

true if the encryption and decryption keys have been recovered.

2.1.3.8 setD()

```
bool Hill::setD (
            const Matrix & D )
```

Sets the decryption key (Matrix D) and encryption key (Matrix E); if the parameter is invalid then set E/D to a 0-by-0 matrix.

Parameters

D	- decryption key.

Returns

true if set is successful, false otherwise.

2.1.3.9 setE()

```
bool Hill::setE (
            const Matrix & E )
```

Sets the encryption key (Matrix E) and decryption key (Matrix D); if the parameter is invalid then set E/D to a 0-by-0 matrix.

Parameters

E	- encryption key.

Returns

true if set is successful, false otherwise.

The documentation for this class was generated from the following file:

- Hill.hpp

Index