Ciphers

Generated by Doxygen 1.8.13

Contents

Chapter 1

README

#Ciphers A program to help with some simple ciphers This program is designed to allow you to play with some simple ciphers, like the Caesar Cipher. Because of how simple these ciphers are they will likely not be of any use if you actually want to keep your data safe

Caesar Cipher This cipher is very easy to learn and simple to encode or decode. It just shifts the letters a certain number of spaces, i.e. for a shift of 3 a=d & x=a

Vigenere Cipher This cipher is more complicated than a simple Caesar cipher. It uses an array of Caesar ciphers and cycles between them based on a key. This makes it more secure, but if you are not carful in your message or have a short key it will still output recognizable paterns that can be easily broken.

Autokey Cipher The Autokey cipher is very similar to the Vigenere cipher, but with a more secure key. Instead of having a repeated key, the key is used one time and the message itself is used as the rest of the key, out to the length of the message. This makes it more secure, but also slower to decrypt.

Atbash Cipher The Atbash cipher is a simple cipher that basically reverses the alphabet. A=Z, B=Y, etc.

Playfair Cipher This cipher is a little more complex. It uses a key to create a grid of letters that is then used to encode 2 letters at a time. It is a simple form of 16-bit encryption. It is still fairly simple to learn to do by hand, just time consuming if it is a long message or if you are trying to crack the key.

Morse Code This is technically not encryption, but it is not exactly English either. It was used originally on the telegraph and was considered an efficient way to send messages at that time. The dots and dashes are representative of long and short pulses from a speaker.

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Class Documentation

4.1 Atbash Class Reference

Public Member Functions

- Atbash ()
  Construct a new Atbash object.
- ~Atbash ()
  Destroy the Atbash object.
- std::string getInputString () const
  Returns the stripped inputString.
- std::string getOutputString () const
  Returns the current outputString.
- std::string encode (std::string input)
  A simple interface that makes sure inputString is set and encodes it using the Atbash cipher.
- std::string decode (std::string input)
  A simple interface that makes sure inputString is set and decodes it using the Atbash cipher.
- void reset ()
  Resets all variables.

4.1.1 Member Function Documentation

4.1.1.1 decode()

```
std::string Atbash::decode (
            std::string input )
```

A simple interface that makes sure inputString is set and decodes it using the Atbash cipher.

Parameters

input	The string that you wish to encode

Returns

The decoded string

4.1.1.2 encode()

```
std::string Atbash::encode (
            std::string input )
```

A simple interface that makes sure inputString is set and encodes it using the Atbash cipher.

Parameters

input	The string that you wish to encode

Returns

The encoded string

4.1.1.3 getInputString()

```
std::string Atbash::getInputString ( ) const
```

Returns the stripped inputString.

Returns

The current inputString

4.1.1.4 getOutputString()

```
std::string Atbash::getOutputString ( ) const
```

Returns the current outputString.

Returns

The current outputString

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

- Headers/Atbash.hpp
- SourceFiles/Atbash.cpp

4.2 Autokey Class Reference

Inheritance diagram for Autokey:

4.3 Caesar Class Reference

Public Member Functions

- Caesar ()

  Construct a new Caesar:: Caesar object.
- ~Caesar ()

  Destroy the Caesar:: Caesar object.
- std::string getInputString () const

  Returns the current inputString.
- int getShift () const

  Returns the current number of letters the cipher is set to shift.
- std::string getOutputString () const

  Returns the last message encoded/decoded.
- std::string encode (int shiftAmount, std::string input)

  Encodes a message using the Caesar cipher.
- std::string decode (int shiftAmount, std::string input)

  Decodes a message using the Caesar cipher.
- void reset ()

  Makes sure all of the variables are blank.

4.3.1 Member Function Documentation

4.3.1.1 decode()

```
std::string Caesar::decode (
            int shiftAmount,
            std::string input )
```

Decodes a message using the Caesar cipher.

Parameters

shiftAmount	The number of letters you need to shift for the cipher
input	The message that needs decoded

Returns

The decoded message

4.3.1.2 encode()

```
std::string Caesar::encode (
            int shiftAmount,
            std::string input )
```

Encodes a message using the Caesar cipher.

Parameters

shiftAmount	The number of letters you need to shift for the cipher
input	The message that needs encoded

Returns

The encoded message

4.3.1.3 getInputString()

```
std::string Caesar::getInputString ( ) const
```

Returns the current inputString.

Returns

The message that was just encoded/decoded

4.3.1.4 getOutputString()

```
std::string Caesar::getOutputString ( ) const
```

Returns the last message encoded/decoded.

Returns

The messge that was just encoded/decoded

4.3.1.5 getShift()

```
int Caesar::getShift ( ) const
```

Returns the current number of letters the cipher is set to shift.

Returns

The current number of letters the cipher is currently set to shift

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

- Headers/Caesar.hpp
- SourceFiles/Caesar.cpp

4.4 Morse Class Reference

Public Member Functions

- Morse ()
  Construct a new Morse object.
- ~Morse ()
  Destroy the Morse object.
- std::string getInputString () const
  Returns the string that needs encoded/decoded.
- std::string getOutputString () const
  Returns the encoded/decoded message.
- std::string encode (std::string input)
  Encodes the message contained in input.
- std::string decode (std::string input)
  Decodes the message contained in input.
- void reset ()
  Makes sure all variables are blank.

4.4.1 Member Function Documentation

4.4.1.1 decode()

```
std::string Morse::decode (
            std::string input )
```

Decodes the message contained in input.

Parameters

input	The message that needs decoded

Returns

The decoded message

4.4.1.2 encode()

```
std::string Morse::encode (
            std::string input )
```

Encodes the message contained in input.

Parameters

input	The message that needs encoded

Returns

The encoded message

4.4.1.3 getInputString()

```
std::string Morse::getInputString ( ) const
```

Returns the string that needs encoded/decoded.

Returns

The string that needs encoded/decoded

4.4.1.4 getOutputString()

```
std::string Morse::getOutputString ( ) const
```

Returns the encoded/decoded message.

Returns

The encoded/decoded message

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

- Headers/Morse.hpp
- SourceFiles/Morse.cpp

4.5 Playfair Class Reference

Public Member Functions

- Playfair ()

  Construct a new Playfair object.
- ~Playfair ()

  Destroy the Playfair object.
- std::string encode (std::string keyword, std::string input)

  Uses keyword to set the grid, encodes input using the rules of the Playfair cipher, and returns the new message.
- std::string decode (std::string keyword, std::string input)

  Uses keyword to set the grid, decodes input using the rules of the Playfair cipher, and returns the new message.
- std::string getKeyword () const

  Returns the current keyword.
- std::string getInputString () const

  Returns the current inputString.
- std::string getOutputString () const

  Returns the current outputString.
- std::string getGrid () const

  Returns a representation of the current grid.
- void reset ()

  Makes sure all variables are blank.

Static Public Member Functions

- static char getReplaced ()

  Returns REPLACED.
- static char getReplacer ()

  Returned REPLACER.
- static char getDoubled ()

  Return DOUBLED.
- static void setReplaced (const char replaced)

  Makes sure replaced is a valid character and sets REPLACED.
- static void setReplacer (const char replacer)

  Makes sure replacer is a valid character and sets REPLACER.
- static void setDoubled (const char doubled)

  Makes sure doubled is a valid character and sets DOUBLED.

4.5.1 Member Function Documentation

4.5.1.1 decode()

```
std::string Playfair::decode (
            std::string keyword,
            std::string input )
```

Uses keyword to set the grid, decodes input using the rules of the Playfair cipher, and returns the new message.

Parameters

keyword	The keyword used to create the grid
input	The message that needs decoded

Returns

The decoded message

4.5.1.2 encode()

```
std::string Playfair::encode (
            std::string keyword,
            std::string input )
```

Uses keyword to set the grid, encodes input using the rules of the Playfair cipher, and returns the new message.

Parameters

keyword	The keyword used to create the grid
input	The message that needs encoded

Returns

The encoded message

4.5.1.3 getDoubled()

```
char Playfair::getDoubled ( )  [static]
```

Return DOUBLED.

Returns

The letter that is added to the end to make the message of even length and placed between 2 ajoining letters that are the same

4.5.1.4 getGrid()

```
std::string Playfair::getGrid ( ) const
```

Returns a representation of the current grid.

Returns

A string representation of the grid

4.5.1.5 getInputString()

```
std::string Playfair::getInputString ( ) const
```

Returns the current inputString.

Returns

The current message that needs encoded/decoded

4.5.1.6 getKeyword()

```
std::string Playfair::getKeyword ( ) const
```

Returns the current keyword.

Returns

The current keyword

4.5.1.7 getOutputString()

```
std::string Playfair::getOutputString ( ) const
```

Returns the current outputString.

Returns

The encoded/decoded message

4.5.1.8 getReplaced()

```
char Playfair::getReplaced ( )  [static]
```

Returns REPLACED.

Returns

The letter that needs replaced for the cipher to work

4.5.1.9 getReplacer()

```
char Playfair::getReplacer ( )  [static]
```

Returned REPLACER.

Returns

The letter that replaces REPLACED

4.5.1.10 setDoubled()

```
void Playfair::setDoubled (
            const char doubled )  [static]
```

Makes sure doubled is a valid character and sets DOUBLED.

Parameters

doubled	The letter to replace DOUBLED

4.5.1.11 setReplaced()

```
void Playfair::setReplaced (
            const char replaced )  [static]
```

Makes sure replaced is a valid character and sets REPLACED.

Parameters

replaced	The letter to replace REPLACED

4.5.1.12 setReplacer()

```
void Playfair::setReplacer (
            const char replacer )  [static]
```

Makes sure replacer is a valid character and sets REPLACER.

Parameters

replacer	The letter to replace REPLACER

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

- Headers/Playfair.hpp
- SourceFiles/Playfair.cpp

4.6 Vigenere Class Reference

Inheritance diagram for Vigenere:

Public Member Functions

- Vigenere ()

  Construct a new Vigenere object.
- ~Vigenere ()

  Destroy the Vigenere object.

- std::string getInputString () const

  Returns the string you wish to encode/decode.
- std::string getOutputString () const

  Returns the encoded/decoded message.
- std::string getKeyword () const

  Returns the keyword you are using for the cipher.
- std::vector< unsigned int > getOffsets () const

  Returns the vector that holds the offsets of the letters used during encoding/decoding.
- std::string encode (std::string key, std::string input)

  Sets all of the variables for the cipher and returns the result.
- std::string decode (std::string key, std::string input)

  Sets all of the variables for the cipher and returns the result.
- void reset ()

  Makes sure all of the variables are empty.

Protected Member Functions

- void setOffset ()

  Uses keyword to set the offsets of the letters used during encoding/decoding.
- void setInputString (std::string input)

  Removes all invalid characters from input and sets it to inputString.
- void setKeyword (std::string key)

  Removes all invalid characters from key and sets it to keyword.
- std::string encode ()

  Encodes the inputString using the Vigenere cipher and saves the result in outputString.
- std::string decode ()

  Decodes the inputString using the Vigenere cipher and saves the result in outputString.

Protected Attributes

- std::string **inputString**
- std::string **outputString**
- std::string **keyword**
- std::vector< unsigned int > **offset**

4.6.1 Member Function Documentation

4.6.1.1 decode() [1/2]

```
std::string Vigenere::decode ( )  [protected]
```

Decodes the inputString using the Vigenere cipher and saves the result in outputString.

Returns

The decoded message

4.6.1.2 decode() [2/2]

```
std::string Vigenere::decode (
            std::string key,
            std::string input )
```

Sets all of the variables for the cipher and returns the result.

Parameters

key	The keyword used in the cipher
input	The message that you wish to decode

Returns

The decoded message

4.6.1.3 encode() [1/2]

```
std::string Vigenere::encode ( )  [protected]
```

Encodes the inputString using the Vigenere cipher and saves the result in outputString.

Returns

The encoded message

4.6.1.4 encode() [2/2]

```
std::string Vigenere::encode (
            std::string key,
            std::string input )
```

Sets all of the variables for the cipher and returns the result.

Parameters

key	The keyword used in the cipher
input	The message that you wish to encode

Returns

The encoded message

4.6.1.5 getInputString()

```
std::string Vigenere::getInputString ( ) const
```

Returns the string you wish to encode/decode.

Returns

The string you wish to encode/decode

4.6.1.6 getKeyword()

```
std::string Vigenere::getKeyword ( ) const
```

Returns the keyword you are using for the cipher.

Returns

The keyword you are using for the cipher

4.6.1.7 getOffsets()

```
std::vector< unsigned int > Vigenere::getOffsets ( ) const
```

Returns the vector that holds the offsets of the letters used during encoding/decoding.

Returns

A the vector holding offsets of the letters used during encoding/decoding

Note

Used mostly for debuging

4.6.1.8 getOutputString()

```
std::string Vigenere::getOutputString ( ) const
```

Returns the encoded/decoded message.

Returns

The encoded/decoded message

4.6.1.9 setInputString()

```
void Vigenere::setInputString (
            std::string input )  [protected]
```

Removes all invalid characters from input and sets it to inputString.

Parameters

input	The string you want to encode/decode

4.6.1.10 setKeyword()

```
void Vigenere::setKeyword (
            std::string key )  [protected]
```

Removes all invalid characters from key and sets it to keyword.

Parameters

key	The keyword used for the cipher

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

- Headers/Vigenere.hpp
- SourceFiles/Vigenere.cpp

Index