Лабораторная работа №4 - Документирование

1.0

Создано системой Doxygen 1.9.1

1 Иерархический список классов 1
1.1 Иерархия классов 1

2 Алфавитный указатель классов 2
2.1 Классы 2

3 Список файлов 2
3.1 Файлы 2

4 Классы 2
4.1 Класс cipher_error 2
4.1.1 Подробное описание 3
4.1.2 Конструктор(ы) 4
4.2 Класс modAlphaCipher 5
4.2.1 Подробное описание 6
4.2.2 Конструктор(ы) 6
4.2.3 Методы 7
4.2.4 Данные класса 11
4.3 Класс RouteCipher 12
4.3.1 Подробное описание 13
4.3.2 Конструктор(ы) 13
4.3.3 Методы 14
4.3.4 Данные класса 15

5 Файлы 16
5.1 Файл gronsfeld/modAlphaCipher.cpp 16
5.1.1 Подробное описание 16
5.1.2 Функции 17
5.2 Файл gronsfeld/modAlphaCipher.h 17
5.3 Файл route_cipher/routeCipher.cpp 18
5.3.1 Подробное описание 19
5.4 Файл route_cipher/routeCipher.h 19

Предметный указатель 21

1 Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

std::invalid_argument

cipher_error 2

cipher_error 2

modAlphaCipher 5

RouteCipher 12

2 Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

cipher_error
Класс исключения для ошибок шифрования 2

modAlphaCipher
Шифрование методом Гронсфельда 5

RouteCipher
Шифрование методом маршрутной перестановки 12

3 Список файлов

3.1 Файлы

Полный список файлов.

gronsfeld/modAlphaCipher.cpp
Реализация шифра Гронсфельда для русского алфавита 16

gronsfeld/modAlphaCipher.h 17

route_cipher/routeCipher.cpp
Реализация шифра маршрутной перестановки 18

route_cipher/routeCipher.h 19

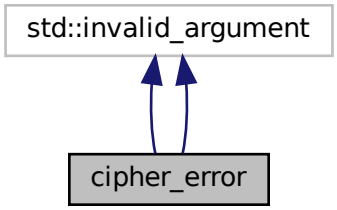
4 Классы

4.1 Класс cipher_error

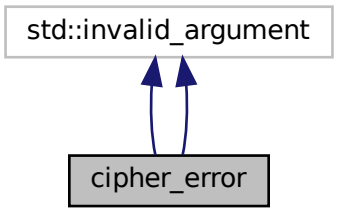
Класс исключения для ошибок шифрования

#include <modAlphaCipher.h>

Граф наследования:cipher_error:



Граф связей класса cipher_error:



Открытые члены

- cipher_error (const std::string &what_arg)
  Конструктор с строкой в качестве параметра
- cipher_error (const char *what_arg)
  Конструктор с C-строкой в качестве параметра
- cipher_error (const std::string &what_arg)
  Конструктор с строкой в качестве параметра
- cipher_error (const char *what_arg)
  Конструктор с C-строкой в качестве параметра

4.1.1 Подробное описание

Класс исключения для ошибок шифрования

Наследуется от std::invalid_argument, используется для всех ошибок, связанных с шифрованием и дешифрованием.

Используется для обработки ошибок в шифре маршрутной перестановки

См. определение в файле modAlphaCipher.h строка 14

4.1.2 Конструктор(ы)

4.1.2.1 cipher_error() [1/4] cipher_error::cipher_error (const std::string & what_arg) [inline], [explicit]

Конструктор с строкой в качестве параметра

Аргументы

what_arg	Сообщение об ошибке

См. определение в файле modAlphaCipher.h строка 20
20 : std::invalid_argument(what_arg) {}

4.1.2.2 cipher_error() [2/4] cipher_error::cipher_error (const char * what_arg) [inline], [explicit]

Конструктор с С-строкой в качестве параметра

Аргументы

what_arg	Сообщение об ошибке

См. определение в файле modAlphaCipher.h строка 26
26 : std::invalid_argument(what_arg) {}

4.1.2.3 cipher_error() [3/4] cipher_error::cipher_error (const std::string & what_arg) [inline], [explicit]

Конструктор с строкой в качестве параметра

Аргументы

what_arg	Сообщение об ошибке

См. определение в файле routeCipher.h строка 16
16 : std::invalid_argument(what_arg) {}

4.1.2.4 cipher_error() [4/4] cipher_error::cipher_error (const char * what_arg) [inline], [explicit]

Конструктор с С-строкой в качестве параметра

Аргументы

what_arg	Сообщение об ошибке

См. определение в файле routeCipher.h строка 22
22 : std::invalid_argument(what_arg) {}

Объявления и описания членов классов находятся в файлах:

- gronsfeld/modAlphaCipher.h
- route_cipher/routeCipher.h

4.2 Класс modAlphaCipher

Шифрование методом Гронсфельда

#include <modAlphaCipher.h>

Открытые члены

- modAlphaCipher ()=delete
  Запрет конструктора без параметров
- modAlphaCipher (const std::wstring &skey)
  Конструктор для установки ключа
- std::wstring encrypt (const std::wstring &open_text)
  Зашифровывание текста
- std::wstring decrypt (const std::wstring &cipher_text)
  Расшифровывание текста

Закрытые члены

- std::wstring getValidKey (const std::wstring &skey)
  Валидация и преобразование ключа
- std::wstring getValidOpenText (const std::wstring &text)
  Валидация открытого текста
- std::wstring getValidCipherText (const std::wstring &text)
  Валидация зашифрованного текста
- std::vector< int > convert (const std::wstring &s)
  Преобразование строки в числовой вектор
- std::wstring convert (const std::vector< int > &v)
  Преобразование числового вектора в строку
- void find_space_positions (const std::wstring &text)
  Поиск позиций пробелов в тексте

Закрытые данные

- std::wstring numAlpha = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"
  Алфавит по порядку
- std::map< wchar_t, int > alphaNum
  Ассоциативный массив "символ-номер".
- std::vector< int > key
  Ключ в числовом виде
- std::vector< size_t > space_positions
  Позиции пробелов в исходном тексте

4.2.1 Подробное описание

Шифрование методом Гронсфельда

Класс реализует шифр Гронсфельда для русского алфавита. Ключ устанавливается в конструкторе. Для зашифровывания и расшифровывания предназначены методы encrypt и decrypt.

Предупреждения

Реализация только для русского языка

См. определение в файле modAlphaCipher.h строка 36

4.2.2 Конструктор(ы)

4.2.2.1 modAlphaCipher() [1/2] modAlphaCipher::modAlphaCipher ( ) [delete]

Запрет конструктора без параметров

4.2.2.2 modAlphaCipher() [2/2] modAlphaCipher::modAlphaCipher (const std::wstring & skey)

Конструктор для установки ключа

Аргументы

skey	Ключ шифрования в виде строки

Исключения

cipher_error	если ключ невалиден

См. определение в файле modAlphaCipher.cpp строка 32

```
33 {
34     std::wstring valid_key = getValidKey(skey);
35     for (size_t i = 0; i < numAlpha.size(); i++) {
36         alphaNum[numAlpha[i]] = i;
37     }
38     key = convert(valid_key);
39 }
```

4.2.3 Методы

4.2.3.1 convert() [1/2] std::wstring modAlphaCipher::convert (const std::vector< int > & v) [private]

Преобразование числового вектора в строку

Аргументы

v	Вектор номеров символов

Возвращает

Строка из символов алфавита

Исключения

cipher_error	если индекс вне диапазона алфавита

См. определение в файле modAlphaCipher.cpp строка 148

```
149 {
150     std::wstring result;
151     for (int i : v) {
152         if (i >= 0 && i < (int)numAlpha.size()) {
153             result += numAlpha[i];
154         } else {
155             throw cipher_error("Invalid index in vector during conversion");
156         }
157     }
158     return result;
159 }
```

4.2.3.2 convert() [2/2] std::vector< int > modAlphaCipher::convert (const std::wstring & s) [private]

Преобразование строки в числовой вектор

Аргументы

s	Строка для преобразования

Возвращает

Вектор номеров символов в алфавите

Исключения

cipher_error	если символ не найден в алфавите

См. определение в файле modAlphaCipher.cpp строка 135

```
136 {
137     std::vector<int> result;
138     for (wchar_t c : s) {
139         if (alphaNum.find(c) != alphaNum.end()) {
140             result.push_back(alphaNum[c]);
141         } else {
142             throw cipher_error("Invalid character in text during conversion");
143         }
144     }
145     return result;
146 }
```

4.2.3.3 decrypt() std::wstring modAlphaCipher::decrypt (const std::wstring & cipher_text)

Расшифровывание текста

Аргументы

cipher_text	Зашифрованный текст. Должен содержать только прописные буквы русского алфавита.

Возвращает

Расшифрованная строка с восстановленными пробелами

Исключения

cipher_error	если текст пустой или содержит недопустимые символы

См. определение в файле modAlphaCipher.cpp строка 106

```
107 {
108     std::wstring valid_cipher_text = getValidCipherText(cipher_text);
109     std::vector<int> work = convert(valid_cipher_text);
110 
111     for (size_t i = 0; i < work.size(); i++) {
112         work[i] = (work[i] + numAlpha.size() - key[i % key.size()]) % numAlpha.size();
113     }
114 
115     std::wstring decrypted_without_spaces = convert(work);
116     std::wstring result;
117     size_t text_index = 0;
118     size_t space_index = 0;
119 
120     for (size_t i = 0; i < decrypted_without_spaces.length() + space_positions.size(); i++) {
121         if (space_index < space_positions.size() && i == space_positions[space_index]) {
122             result += L' ';
123             space_index++;
124         } else {
125             if (text_index < decrypted_without_spaces.length()) {
126                 result += decrypted_without_spaces[text_index];
```

```
127                 text_index++;
128             }
129         }
130     }
131
132     return result;
133 }
```

4.2.3.4 encrypt() std::wstring modAlphaCipher::encrypt (const std::wstring & open_text)

Зашифровывание текста

Аргументы

open_text	Открытый текст. Не должен быть пустой строкой. Строчные символы автоматически преобразуются к прописным. Все не-буквы удаляются.

Возвращает

Зашифрованная строка

Исключения

cipher_error	если текст пустой

См. определение в файле modAlphaCipher.cpp строка 93

```
94 {
95     std::wstring valid_text = getValidOpenText(open_text);
96     find_space_positions(open_text);
97
98     std::vector<int> work = convert(valid_text);
99     for (size_t i = 0; i < work.size(); i++) {
100         work[i] = (work[i] + key[i % key.size()]) % numAlpha.size();
101     }
102
103     return convert(work);
104 }
```

4.2.3.5 find_space_positions() void modAlphaCipher::find_space_positions (const std::wstring & text) [private]

Поиск позиций пробелов в тексте

Аргументы

text	Исходный текст

См. определение в файле modAlphaCipher.cpp строка 83

```
84 {
85     space_positions.clear();
86     for (size_t i = 0; i < text.length(); i++) {
87         if (text[i] == L' ') {
```

```
88             space_positions.push_back(i);
89         }
90     }
91 }
```

4.2.3.6 getValidCipherText() std::wstring modAlphaCipher::getValidCipherText (const std::wstring & text) [private]

Валидация зашифрованного текста

Аргументы

text	Зашифрованный текст

Возвращает

Исходный текст

Исключения

cipher_error	если текст пустой или содержит недопустимые символы

См. определение в файле modAlphaCipher.cpp строка 71

```
71                                                  {
72     if (text.empty()) {
73         throw cipher_error("Empty cipher text");
74     }
75     for (wchar_t c : text) {
76         if (alphaNum.find(c) == alphaNum.end()) {
77             throw cipher_error("Invalid cipher text: character not in alphabet found");
78         }
79     }
80     return text;
81 }
```

4.2.3.7 getValidKey() std::wstring modAlphaCipher::getValidKey (const std::wstring & skey) [private]

Валидация и преобразование ключа

Аргументы

skey	Ключ в виде строки

Возвращает

Валидный ключ в верхнем регистре

Исключения

cipher_error	если ключ пустой или содержит некириллические символы

См. определение в файле modAlphaCipher.cpp строка 41

```
41                                                  {
42     if (skey.empty()) {
43         throw cipher_error("Empty key");
44     }
45     std::wstring tmp;
46     for (wchar_t c : skey) {
47         if (!isCyrillicLetter(c)) {
48             throw cipher_error("Invalid key character: non-cyrillic character found");
49         }
50         tmp += toUpperCyrillic(c);
51     }
52     if (tmp.empty()) {
53         throw cipher_error("Empty key after processing");
54     }
55     return tmp;
56 }
```

4.2.3.8 getValidOpenText() std::wstring modAlphaCipher::getValidOpenText (const std::wstring & text) [private]

Валидация открытого текста

Аргументы

text	Открытый текст

Возвращает

Текст без не-букв, в верхнем регистре

Исключения

cipher_error	если после обработки текст пуст

См. определение в файле modAlphaCipher.cpp строка 58

```
58                                                  {
59     std::wstring tmp;
60     for (wchar_t c : text) {
61         if (isCyrillicLetter(c)) {
62             tmp += toUpperCyrillic(c);
63         }
64     }
65     if (tmp.empty()) {
66         throw cipher_error("Empty open text after processing");
67     }
68     return tmp;
69 }
```

4.2.4 Данные класса

4.2.4.1 alphaNum std::map<wchar_t, int> modAlphaCipher::alphaNum [private]

Ассоциативный массив "символ-номер".

См. определение в файле modAlphaCipher.h строка 40

4.2.4.2 key std::vector<int> modAlphaCipher::key [private]

Ключ в числовом виде

См. определение в файле modAlphaCipher.h строка 41

4.2.4.3 numAlpha std::wstring modAlphaCipher::numAlpha = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ" [private]

Алфавит по порядку

См. определение в файле modAlphaCipher.h строка 39

4.2.4.4 space_positions std::vector<size_t> modAlphaCipher::space_positions [private]

Позиции пробелов в исходном тексте

См. определение в файле modAlphaCipher.h строка 42

Объявления и описания членов классов находятся в файлах:

- gronsfeld/modAlphaCipher.h
- gronsfeld/modAlphaCipher.cpp

4.3 Класс RouteCipher

Шифрование методом маршрутной перестановки

#include <routeCipher.h>

Открытые члены

- RouteCipher ()=delete
  Запрет конструктора без параметров
- RouteCipher (int k)
  Конструктор для установки ключа
- std::string encrypt (const std::string &text)
  Шифрование текста методом маршрутной перестановки
- std::string decrypt (const std::string &text)
  Дешифрование текста методом маршрутной перестановки

Закрытые данные

- int key

  Количество столбцов таблицы

4.3.1 Подробное описание

Шифрование методом маршрутной перестановки

Класс реализует шифр табличной маршрутной перестановки. Маршрут записи: по горизонтали слева направо, сверху вниз. Маршрут считывания: сверху вниз, справа налево.

Предупреждения

Ключ (количество столбцов) должен быть положительным числом

См. определение в файле routeCipher.h строка 32

4.3.2 Конструктор(ы)

4.3.2.1 RouteCipher() [1/2] RouteCipher::RouteCipher ( ) [delete]

Запрет конструктора без параметров

4.3.2.2 RouteCipher() [2/2] RouteCipher::RouteCipher (int k)

Конструктор для установки ключа

Аргументы

k	Количество столбцов таблицы

Исключения

cipher_error	если k <= 0

См. определение в файле routeCipher.cpp строка 18

```
18                          {
19     if (k <= 0) {
20         throw cipher_error("Ключ (количество столбцов) должен быть положительным числом");
21     }
22     key = k;
23 }
```

4.3.3 Методы

4.3.3.1 decrypt() std::string RouteCipher::decrypt (const std::string & text)

Дешифрование текста методом маршрутной перестановки

Аргументы

text	Зашифрованный текст. Записывается в таблицу по столбцам справа налево, считывается построчно.

Возвращает

Расшифрованный текст

Исключения

cipher_error	если текст пустой

См. определение в файле routeCipher.cpp строка 50

```
50                                          {
51     if (text.empty()) {
52         throw cipher_error("Текст не может быть пустым");
53     }
54 
55     int rows = (text.length() + key - 1) / key;
56     vector<vector<char» table(rows, vector<char>(key, ' '));
57 
58     size_t index = 0;
59     for (int col = key - 1; col >= 0; col--) {
60         for (int row = 0; row < rows; row++) {
61             if (row * key + col < static_cast<int>(text.length())) {
62                 if (index < text.length()) {
63                     table[row][col] = text[index++];
64                 }
65             }
66         }
67     }
68 
69     string result;
70     for (int row = 0; row < rows; row++) {
71         for (int col = 0; col < key; col++) {
72             if (row * key + col < static_cast<int>(text.length())) {
73                 result += table[row][col];
74             }
75         }
76     }
77 
78     return result;
79 }
```

4.3.3.2 encrypt() std::string RouteCipher::encrypt (const std::string & text)

Шифрование текста методом маршрутной перестановки

Аргументы

text	Текст для шифрования. Записывается в таблицу построчно, считывается по столбцам справа налево.

Возвращает

Зашифрованный текст

Исключения

cipher_error	если текст пустой

См. определение в файле routeCipher.cpp строка 25

```
25                                          {
26     if (text.empty()) {
27         throw cipher_error("Текст не может быть пустым");
28     }
29
30     int rows = (text.length() + key - 1) / key;
31     vector<vector<char» table(rows, vector<char>(key, ' '));
32
33     for (size_t i = 0; i < text.length(); i++) {
34         int row = i / key;
35         int col = i % key;
36         table[row][col] = text[i];
37     }
38
39     string result;
40     for (int col = key - 1; col >= 0; col--) {
41         for (int row = 0; row < rows; row++) {
42             if (row * key + col < static_cast<int>(text.length())) {
43                 result += table[row][col];
44             }
45         }
46     }
47     return result;
48 }
```

4.3.4 Данные класса

4.3.4.1 key int RouteCipher::key [private]

Количество столбцов таблицы

См. определение в файле routeCipher.h строка 34

Объявления и описания членов классов находятся в файлах:

- route_cipher/routeCipher.h
- route_cipher/routeCipher.cpp

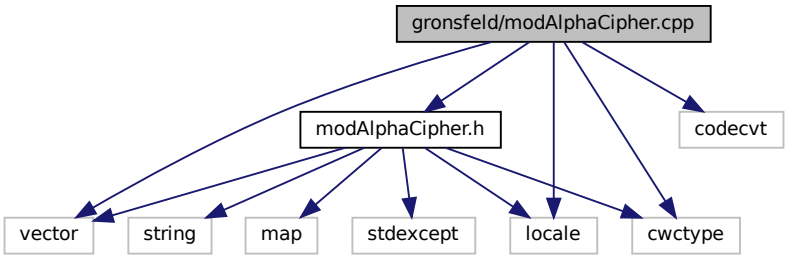
5 Файлы

5.1 Файл gronsfeld/modAlphaCipher.cpp

Реализация шифра Гронсфельда для русского алфавита

```
#include "modAlphaCipher.h"
#include <locale>
#include <codecvt>
#include <vector>
#include <cwctype>
```

Граф включаемых заголовочных файлов для modAlphaCipher.cpp:



Функции

- bool isCyrillicLetter (wchar_t c)
- wchar_t toUpperCyrillic (wchar_t c)

5.1.1 Подробное описание

Реализация шифра Гронсфельда для русского алфавита

Автор

Тенишев Рамиль

Версия

1.0

Дата

05.12.2025

Авторство

Учебный проект

Предупреждения

Это учебный пример

5.1.2 Функции

5.1.2.1 isCyrillicLetter() bool isCyrillicLetter (wchar_t c)

См. определение в файле modAlphaCipher.cpp строка 18

```
18                                  {
19     return (c >= L'А' && c <= L'Я') || (c >= L'а' && c <= L'я') || c == L'Ё' || c == L'ё';
20 }
```

5.1.2.2 toUpperCyrillic() wchar_t toUpperCyrillic (wchar_t c)

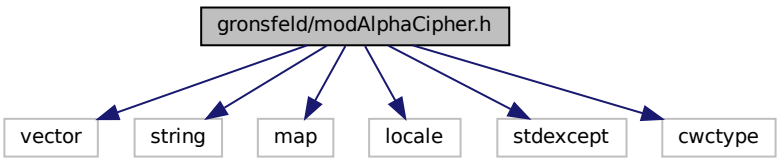
См. определение в файле modAlphaCipher.cpp строка 23

```
23                                  {
24     if (c >= L'а' && c <= L'я') {
25         return c - (L'а' - L'А');
26     } else if (c == L'ё') {
27         return L'Ё';
28     }
29     return c;
30 }
```

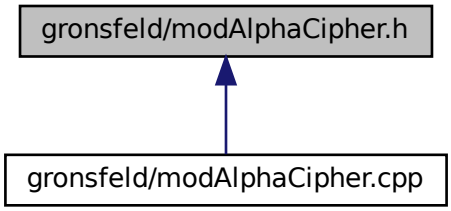
5.2 Файл gronsfeld/modAlphaCipher.h

```
#include <vector>
#include <string>
#include <map>
#include <locale>
#include <stdexcept>
#include <cwctype>
```

Граф включаемых заголовочных файлов для modAlphaCipher.h:



Граф файлов, в которые включается этот файл:



Классы

- class cipher_error

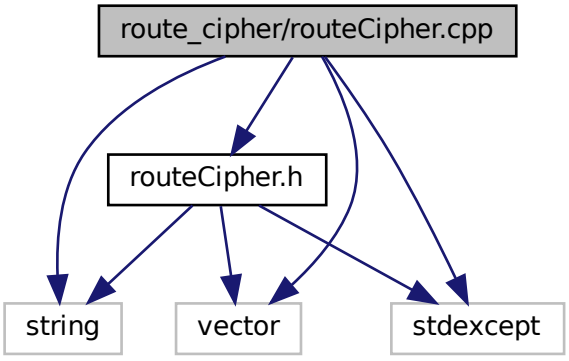
  Класс исключения для ошибок шифрования
- class modAlphaCipher

  Шифрование методом Гронсфельда

5.3 Файл route_cipher/routeCipher.cpp

Реализация шифра маршрутной перестановки

#include "routeCipher.h"
#include <vector>
#include <string>
#include <stdexcept>
Граф включаемых заголовочных файлов для routeCipher.cpp:



5.3.1 Подробное описание

Реализация шифра маршрутной перестановки

Автор

Тенишев Рамиль

Версия

1.0

Дата

05.12.2025

Авторство

Учебный проект

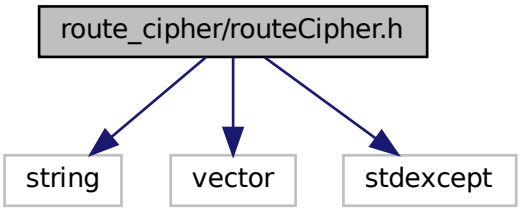
Предупреждения

Это учебный пример

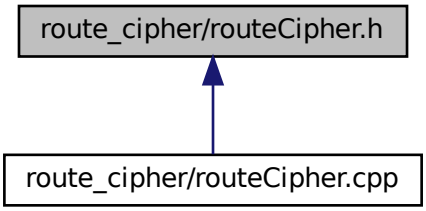
5.4 Файл route_cipher/routeCipher.h

```
#include <string>
#include <vector>
#include <stdexcept>
```

Граф включаемых заголовочных файлов для routeCipher.h:



Граф файлов, в которые включается этот файл:



Классы

- class cipher_error

  Класс исключения для ошибок шифрования
- class RouteCipher

  Шифрование методом маршрутной перестановки

Предметный указатель

alphaNum
    modAlphaCipher, 11

cipher_error, 2
    cipher_error, 4
convert
    modAlphaCipher, 7

decrypt
    modAlphaCipher, 8
    RouteCipher, 14

encrypt
    modAlphaCipher, 9
    RouteCipher, 14

find_space_positions
    modAlphaCipher, 9

getValidCipherText
    modAlphaCipher, 10
getValidKey
    modAlphaCipher, 10
getValidOpenText
    modAlphaCipher, 11
gronsfeld/modAlphaCipher.cpp, 16
gronsfeld/modAlphaCipher.h, 17

isCyrillicLetter
    modAlphaCipher.cpp, 17

key
    modAlphaCipher, 12
    RouteCipher, 15

modAlphaCipher, 5
    alphaNum, 11
    convert, 7
    decrypt, 8
    encrypt, 9
    find_space_positions, 9
    getValidCipherText, 10
    getValidKey, 10
    getValidOpenText, 11
    key, 12
    modAlphaCipher, 6
    numAlpha, 12
    space_positions, 12
modAlphaCipher.cpp
    isCyrillicLetter, 17
    toUpperCyrillic, 17

numAlpha
    modAlphaCipher, 12

route_cipher/routeCipher.cpp, 18
route_cipher/routeCipher.h, 19

RouteCipher, 12
    decrypt, 14
    encrypt, 14
    key, 15
    RouteCipher, 13

space_positions
    modAlphaCipher, 12

toUpperCyrillic
    modAlphaCipher.cpp, 17