My Project

Создано системой Doxygen 1.9.8

Глава 1

lb3

Файлы к лабораторной работе 3

Глава 2

Список тестов

Член SUITE (EncryptTest)
  TextWithNumbers
Член SUITE (KeyTest)
  ValidKey
  Suite KeyTest
Член SUITE (DecryptTest)
  MaxShiftDecrypt
  EmptyDecrypt
  PunctDecrypt
  DigitsDecrypt
  WhitespaceDecrypt
  LowCaseDecrypt
  BasicDecrypt
  Suite DecryptTest
Член SUITE (EncryptTest)
  MaxShiftKey
  NoAlphaString
  EmptyString
Член SUITE (KeyTest)
  Suite KeyTest
Член SUITE (EncryptTest)
  TextWithSpaces
  LowCaseEncrypt
  BasicEncrypt
  Suite EncryptTest
Член SUITE (KeyTest)
  WeakKey
  EmptyKey
  WhitespaceInKey
  PunctuationInKey
  DigitsInKey
  LowCaseKey
  LongKey
  ValidKey

Глава 3

Иерархический список классов

3.1 Иерархия классов

Иерархия классов.

Глава 4

Алфавитный указатель классов

4.1 Классы

Классы с их кратким описанием.

Глава 5

Список файлов

5.1 Файлы

Полный список файлов.

Глава 6

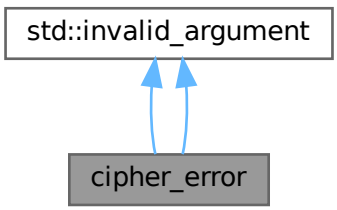
Классы

6.1 Класс cipher_error

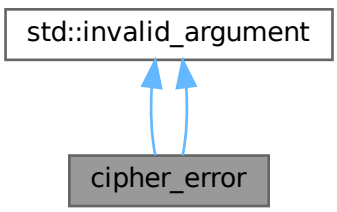
Исключение для ошибок шифрования

#include <modAlphaCipher.h>

Граф наследования:cipher_error:



Граф связей класса cipher_error:



Открытые члены

- cipher_error (const std::string &what_arg)
  Конструктор с строкой
- cipher_error (const char *what_arg)
  Конструктор с си-строкой
- cipher_error (const string &what_arg)
  Конструктор с строкой
- cipher_error (const char *what_arg)
  Конструктор с си-строкой

6.1.1 Подробное описание

Исключение для ошибок шифрования

Исключение для ошибок шифрования маршрутной перестановкой

Наследуется от std::invalid_argument, используется для обработки ошибок при работе с шифром

6.1.2 Конструктор(ы)

6.1.2.1 cipher_error() [1/4]

cipher_error::cipher_error (
const std::string & what_arg) [inline], [explicit]

Конструктор с строкой

Аргументы

in	what_arg	Сообщение об ошибке

6.1.2.2 cipher_error() [2/4]

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

Конструктор с си-строкой

Аргументы

in	what_arg	Сообщение об ошибке

6.1.2.3 cipher_error() [3/4]

cipher_error::cipher_error (
const string & what_arg) [inline], [explicit]

Конструктор с строкой

Аргументы

in	what_arg	Сообщение об ошибке

6.1.2.4 cipher_error() [4/4]

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

Конструктор с си-строкой

Аргументы

in	what_arg	Сообщение об ошибке

Объявления и описания членов классов находятся в файлах:

- 1/modAlphaCipher.h
- 2/route.h

6.2 Класс code

Класс для шифрования методом маршрутной перестановки

#include <route.h>

Открытые члены

- code ()=delete
  Удаленный конструктор по умолчанию
- code (int skey, string text)
  Конструктор с ключом и текстом
- string encryption (const string &text)
  Шифрование текста
- string transcript (const string &text, const string &open_text)
  Дешифрование текста

Закрытые члены

- int getValidKey (int key, const string &Text)
  Проверка валидности ключа
- string getValidOpenText (const string &s)
  Проверка валидности открытого текста
- string getValidCipherText (const string &s, const string &open_text)
  Проверка валидности зашифрованного текста

Закрытые данные

- int key

  Ключ шифрования (количество столбцов)

6.2.1 Подробное описание

Класс для шифрования методом маршрутной перестановки

Шифрование происходит путем записи текста в таблицу по строкам и чтения по столбцам в обратном порядке

6.2.2 Конструктор(ы)

6.2.2.1 code() [1/2]

code::code ( ) [delete]

Удаленный конструктор по умолчанию

6.2.2.2 code() [2/2]

code::code (
int skey,
string text)

Конструктор с ключом и текстом

Аргументы

in	skey	Ключ шифрования
in	text	Текст для инициализации
in	skey	Ключ шифрования
in	text	Текст для инициализации

Проверяет валидность ключа относительно длины текста

6.2.3 Методы

6.2.3.1 encryption()

string code::encryption (
const string & text)

Шифрование текста

Шифрование текста методом маршрутной перестановки

Аргументы

in	text	Текст для шифрования

Возвращает

Зашифрованный текст

Аргументы

in	text	Текст для шифрования

Возвращает

Зашифрованный текст

Алгоритм:

1. Запись текста в таблицу по строкам
2. Чтение таблицы по столбцам справа налево

6.2.3.2 getValidCipherText()

string code::getValidCipherText (
const string & s,
const string & open_text) [inline], [private]

Проверка валидности зашифрованного текста

Аргументы

in	s	Зашифрованный текст
in	open_text	Исходный открытый текст

Возвращает

Валидированный зашифрованный текст

Исключения

cipher_error	при несоответствии длин

6.2.3.3 getValidKey()

int code::getValidKey (

int key,
const string & Text) [inline], [private]

Проверка валидности ключа

Аргументы

in	key	Ключ для проверки
in	Text	Текст для шифрования

Возвращает

Валидный ключ

Исключения

cipher_error	при невалидном ключе

Аргументы

in	key	Ключ для проверки
in	Text	Текст для шифрования

Возвращает

Валидный ключ

Исключения

cipher_error	если ключ меньше 2 или больше длины текста

6.2.3.4 getValidOpenText()

string code::getValidOpenText (
const string & s) [inline], [private]

Проверка валидности открытого текста

Аргументы

in	s	Текст для проверки

Возвращает

Валидированный текст

Исключения

cipher_error	при невалидном тексте

Аргументы

in	s	Текст для проверки

Возвращает

Валидированный текст (без пробелов, только буквы)

Исключения

cipher_error	при пустом тексте или недопустимых символах

6.2.3.5 transcript()

```
string code::transcript (
            const string & text,
            const string & open_text )
```

Дешифрование текста

Аргументы

in	text	Зашифрованный текст
in	open_text	Исходный открытый текст (для проверки длины)

Возвращает

Расшифрованный текст

Аргументы

in	text	Зашифрованный текст
in	open_text	Исходный открытый текст (для проверки длины)

Возвращает

Расшифрованный текст

Исключения

cipher_error	при несоответствии длин или невалидных символах

6.2.4 Данные класса

6.2.4.1 key

int code::key [private]

Ключ шифрования (количество столбцов)

Объявления и описания членов классов находятся в файлах:

- 2/route.h
- 2/route.cpp

6.3 Класс modAlphaCipher

Класс для шифрования методом модифицированного алфавитного шифра

#include <modAlphaCipher.h>

Открытые члены

- modAlphaCipher ()=delete
  Удаленный конструктор по умолчанию
- modAlphaCipher (const std::string &skey)
  Конструктор с ключом
- std::string encrypt (const std::string &open_text)
  Шифрование текста
- std::string decrypt (const std::string &cipher_text)
  Дешифрование текста

Закрытые члены

- std::vector< int > convert (const std::string &s)
  Преобразование строки в вектор числовых индексов
- std::string convert (const std::vector< int > &v)
  Преобразование вектора индексов в строку
- std::string getValidKey (const std::string &s)
  Проверка и нормализация ключа
- std::string getValidOpenText (const std::string &s)
  Проверка и нормализация открытого текста
- std::string getValidCipherText (const std::string &s)
  Проверка зашифрованного текста

Закрытые данные

- std::wstring numAlpha = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"
  Русский алфавит
- std::map< wchar_t, int > alphaNum
  Отображение символа в его индекс
- std::vector< int > key
  Ключ шифрования в числовом формате

6.3.1 Подробное описание

Класс для шифрования методом модифицированного алфавитного шифра

Реализует шифрование с использованием ключа на основе русского алфавита. Поддерживает только русские буквы, автоматически преобразует регистр.

6.3.2 Конструктор(ы)

6.3.2.1 modAlphaCipher() [1/2]

modAlphaCipher::modAlphaCipher ( ) [delete]

Удаленный конструктор по умолчанию

6.3.2.2 modAlphaCipher() [2/2]

modAlphaCipher::modAlphaCipher (
const std::string & skey)

Конструктор с ключом

Аргументы

in	skey	Ключ шифрования

Исключения

cipher_error	при слабом или невалидном ключе

Аргументы

in	skey	Ключ шифрования в виде строки

Исключения

cipher_error	при слабом или невалидном ключе

Инициализирует таблицу преобразований и проверяет ключ на слабость

6.3.3 Методы

6.3.3.1 convert() [1/2]

std::vector< int > modAlphaCipher::convert (
const std::string & s) [private]

Преобразование строки в вектор числовых индексов

Аргументы

in	s	Входная строка

Возвращает

Вектор индексов символов

6.3.3.2 convert() [2/2]

std::string modAlphaCipher::convert (
const std::vector< int > & v) [private]

Преобразование вектора индексов в строку

Аргументы

in	v	Вектор индексов

Возвращает

Результирующая строка

6.3.3.3 decrypt()

std::string modAlphaCipher::decrypt (
const std::string & cipher_text)

Дешифрование текста

Аргументы

in	cipher_text	Зашифрованный текст

Возвращает

Расшифрованный текст

Исключения

cipher_error	при ошибках валидации

Аргументы

in	cipher_text	Зашифрованный текст

Возвращает

Расшифрованный текст

Исключения

cipher_error	при ошибках валидации

Алгоритм: P_i = (C_i - K_{i mod len(K)} + N) mod N

6.3.3.4 encrypt()

```
std::string modAlphaCipher::encrypt (
        const std::string & open_text )
```

Шифрование текста

Аргументы

in	open_text	Открытый текст для шифрования

Возвращает

Зашифрованный текст

Исключения

cipher_error	при ошибках валидации

Аргументы

in	open_text	Открытый текст для шифрования

Возвращает

Зашифрованный текст

Исключения

cipher_error	при ошибках валидации

Алгоритм: C_i = (P_i + K_{i mod len(K)}) mod N

6.3.3.5 getValidCipherText()

```
std::string modAlphaCipher::getValidCipherText (
        const std::string & s )  [private]
```

Проверка зашифрованного текста

Аргументы

in	s	Зашифрованный текст

Возвращает

Валидированный текст

Исключения

cipher_error	при невалидном тексте

Аргументы

in	s	Зашифрованный текст

Возвращает

Валидированный текст

Исключения

cipher_error	при пустом тексте или недопустимых символах

6.3.3.6 getValidKey()

std::string modAlphaCipher::getValidKey (
const std::string & s) [private]

Проверка и нормализация ключа

Аргументы

in	s	Ключ в виде строки

Возвращает

Валидированный ключ

Исключения

cipher_error	при невалидном ключе

Аргументы

in	s	Ключ в виде строки

Возвращает

Валидированный ключ (все символы заглавные)

Исключения

cipher_error	при пустом ключе или не-буквенных символах

6.3.3.7 getValidOpenText()

```
std::string modAlphaCipher::getValidOpenText (
            const std::string & s )  [private]
```

Проверка и нормализация открытого текста

Аргументы

in	s	Открытый текст

Возвращает

Валидированный текст

Исключения

cipher_error	при невалидном тексте

Аргументы

in	s	Открытый текст

Возвращает

Валидированный текст (только заглавные русские буквы)

Исключения

cipher_error	при пустом тексте

6.3.4 Данные класса

6.3.4.1 alphaNum

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

Отображение символа в его индекс

6.3.4.2 key

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

Ключ шифрования в числовом формате

6.3.4.3 numAlpha

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

Русский алфавит

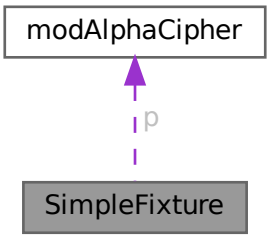
Объявления и описания членов классов находятся в файлах:

- 1/modAlphaCipher.h
- 1/modAlphaCipher.cpp

6.4 Структура SimpleFixture

Фикстура для тестов с предустановленным шифром

Граф связей класса SimpleFixture:



Открытые члены

- SimpleFixture ()
  Конструктор фикстуры
- ~SimpleFixture ()
  Деструктор фикстуры

Открытые атрибуты

- modAlphaCipher * p
  Указатель на шифр

6.4.1 Подробное описание

Фикстура для тестов с предустановленным шифром

6.4.2 Конструктор(ы)

6.4.2.1 SimpleFixture()

SimpleFixture::SimpleFixture ( ) [inline]

Конструктор фикстуры

Создает шифр с ключом "БОРЩ"

6.4.2.2 ~SimpleFixture()

SimpleFixture::~SimpleFixture ( ) [inline]

Деструктор фикстуры

6.4.3 Данные класса

6.4.3.1 p

modAlphaCipher* SimpleFixture::p

Указатель на шифр

Объявления и описания членов структуры находятся в файле:

- 1/main.cpp

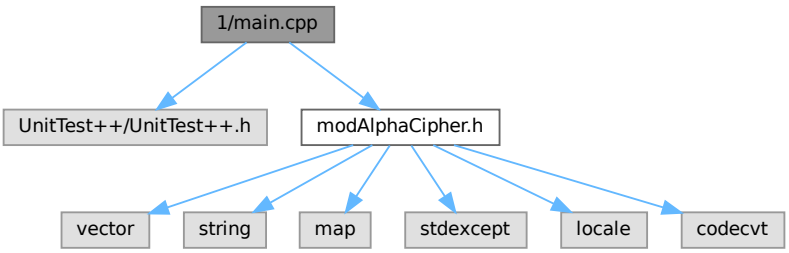
Глава 7

Файлы

7.1 Файл 1/main.cpp

Тесты для класса modAlphaCipher.

#include <UnitTest++/UnitTest++.h>
#include "modAlphaCipher.h"
Граф включаемых заголовочных файлов для main.cpp:



Классы

- struct SimpleFixture

  Фикстура для тестов с предустановленным шифром

Функции

- SUITE (KeyTest)

  Тесты для проверки ключа шифрования
- SUITE (EncryptTest)

  Тесты для шифрования
- SUITE (DecryptTest)

  Тесты для дешифрования
- int main (int argc, char **argv)

  Главная функция для запуска тестов

7.1.1 Подробное описание

Тесты для класса modAlphaCipher.

Автор

Назарова Софья

Дата

2025

Авторство

WECT ПГУ

7.1.2 Функции

7.1.2.1 main()

```
int main (
            int argc,
            char ** argv )
```

Главная функция для запуска тестов

Аргументы

in	argc	Количество аргументов командной строки
in	argv	Аргументы командной строки

Возвращает

Код завершения (0 - все тесты прошли успешно)

7.1.2.2 SUITE() [1/3]

```
SUITE (
            DecryptTest  )
```

Тесты для дешифрования

Тест Suite DecryptTest

Тест BasicDecrypt

Базовое дешифрование

Тест LowCaseDecrypt

Дешифрование текста в нижнем регистре (ожидается исключение)

Тест WhitespaceDecrypt

Дешифрование текста с пробелами (ожидается исключение)

Тест DigitsDecrypt

Дешифрование текста с цифрами (ожидается исключение)

Тест PunctDecrypt

Дешифрование текста со знаками препинания (ожидается исключение)

Тест EmptyDecrypt

Дешифрование пустой строки (ожидается исключение)

Тест MaxShiftDecrypt

Дешифрование с максимальным сдвигом

7.1.2.3 SUITE() [2/3]

SUITE (
EncryptTest)

Тесты для шифрования

Тест Suite EncryptTest

Тест BasicEncrypt

Базовое шифрование

Тест LowCaseEncrypt

Шифрование текста в нижнем регистре

Тест TextWithSpaces

Шифрование текста с пробелами

Тест TextWithNumbers

Шифрование текста с цифрами

Тест EmptyString

Шифрование пустой строки (ожидается исключение)

Тест NoAlphaString

Шифрование строки без букв (ожидается исключение)

Тест MaxShiftKey

Шифрование с максимальным сдвигом (ключ "Я")

7.1.2.4 SUITE() [3/3]

SUITE (
KeyTest)

Тесты для проверки ключа шифрования

Тест Suite KeyTest

Тест ValidKey

Проверка создания шифра с валидным ключом

Тест LongKey

Проверка длинного ключа

Тест LowCaseKey

Проверка ключа в нижнем регистре

Тест DigitsInKey

Проверка ключа с цифрами (ожидается исключение)

Тест PunctuationInKey

Проверка ключа со знаками препинания (ожидается исключение)

Тест WhitespaceInKey

Проверка ключа с пробелами (ожидается исключение)

Тест EmptyKey

Проверка пустого ключа (ожидается исключение)

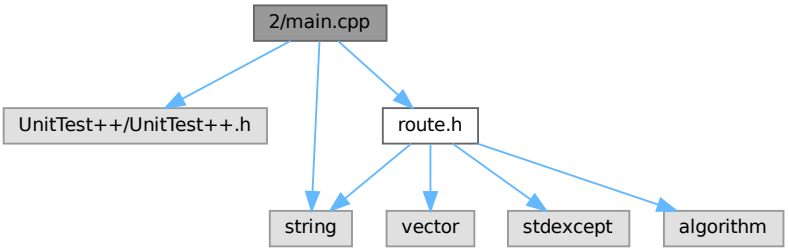
Тест WeakKey

Проверка слабого ключа (все символы одинаковые, ожидается исключение)

7.2 Файл 2/main.cpp

Тесты для класса code (шифр маршрутной перестановки)

#include <UnitTest++/UnitTest++.h>
#include "route.h"
#include <string>
Граф включаемых заголовочных файлов для main.cpp:



Функции

- SUITE (KeyTest)
  Тесты для проверки ключа шифрования

7.2.1 Подробное описание

Тесты для класса code (шифр маршрутной перестановки)

Автор

Назарова Софья

Дата

2025

7.2.2 Функции

7.2.2.1 SUITE()

SUITE (
KeyTest)

Тесты для проверки ключа шифрования

Тест Suite KeyTest

Тест ValidKey

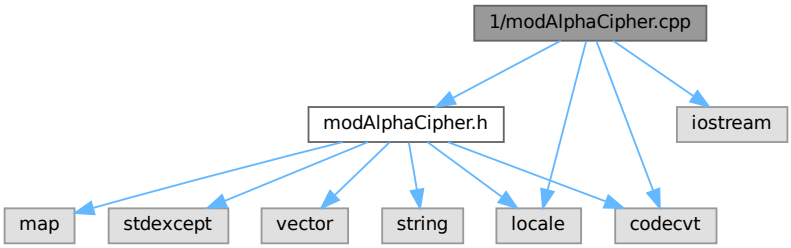
Проверка валидного ключа

7.3 Файл 1/modAlphaCipher.cpp

Реализация класса modAlphaCipher.

```
#include "modAlphaCipher.h"
#include <locale>
#include <codecvt>
#include <iostream>
```

Граф включаемых заголовочных файлов для modAlphaCipher.cpp:



Переменные

- std::wstring_convert< std::codecvt_utf8< wchar_t >, wchar_t > codec
  Конвертер UTF-8.

7.3.1 Подробное описание

Реализация класса modAlphaCipher.

Автор

Назарова Софья

Дата

2025

Авторство

WECT ПГУ

7.3.2 Переменные

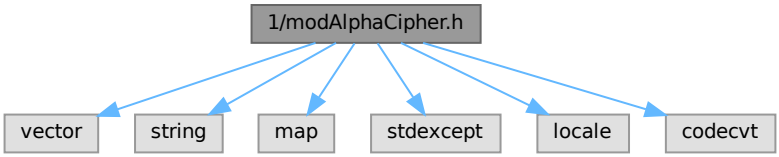
7.3.2.1 codec

std::wstring_convert<std::codecvt_utf8<wchar_t>, wchar_t> codec

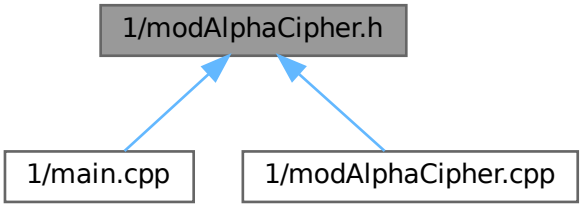
Конвертер UTF-8.

7.4 Файл 1/modAlphaCipher.h

#include <vector>
#include <string>
#include <map>
#include <stdexcept>
#include <locale>
#include <codecvt>
Граф включаемых заголовочных файлов для modAlphaCipher.h:



Граф файлов, в которые включается этот файл:



Классы

- class cipher_error

  Исключение для ошибок шифрования
- class modAlphaCipher

  Класс для шифрования методом модифицированного алфавитного шифра

7.5 modAlphaCipher.h

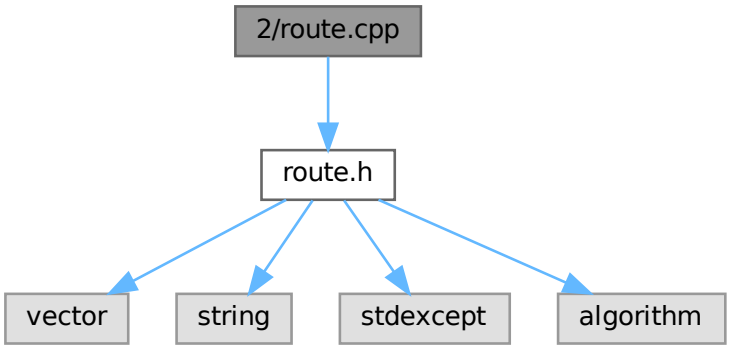
См. документацию.
```
00001 #pragma once
00002 #include <vector>
00003 #include <string>
00004 #include <map>
00005 #include <stdexcept>
00006 #include <locale>
```

```
00007 #include <codecvt>
00008
00015 class cipher_error: public std::invalid_argument {
00016     public:
00021         explicit cipher_error (const std::string& what_arg):
00022         std::invalid_argument(what_arg) {}
00023
00028         explicit cipher_error (const char* what_arg):
00029         std::invalid_argument(what_arg) {}
00030 };
00031
00038 class modAlphaCipher {
00039     private:
00040         std::wstring numAlpha = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";
00041         std::map<wchar_t,int> alphaNum;
00042         std::vector<int> key;
00043
00049         std::vector<int> convert(const std::string& s);
00050
00056         std::string convert(const std::vector<int>& v);
00057
00064         std::string getValidKey(const std::string & s);
00065
00072         std::string getValidOpenText(const std::string & s);
00073
00080         std::string getValidCipherText(const std::string & s);
00081
00082     public:
00086         modAlphaCipher()=delete;
00087
00093         modAlphaCipher(const std::string& skey);
00094
00101         std::string encrypt(const std::string& open_text);
00102
00109         std::string decrypt(const std::string& cipher_text);
00110 };
```

7.6 Файл 2/route.cpp

Реализация класса code (шифр маршрутной перестановки)

#include "route.h"
Граф включаемых заголовочных файлов для route.cpp:



7.6.1 Подробное описание

Реализация класса code (шифр маршрутной перестановки)

Автор

Назарова Софья

Дата

2025

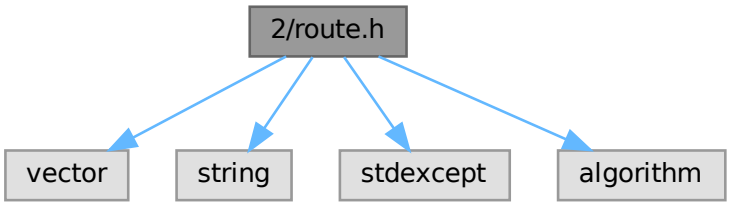
Авторство

WECT ПГУ

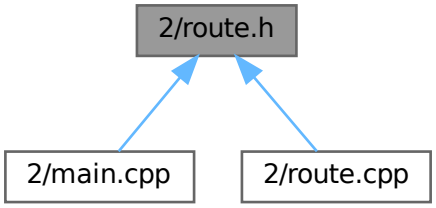
7.7 Файл 2/route.h

Заголовочный файл для класса code (шифр маршрутной перестановки)

#include <vector>
#include <string>
#include <stdexcept>
#include <algorithm>
Граф включаемых заголовочных файлов для route.h:



Граф файлов, в которые включается этот файл:



Классы

- class cipher_error

  Исключение для ошибок шифрования
- class code

  Класс для шифрования методом маршрутной перестановки

7.7.1 Подробное описание

Заголовочный файл для класса code (шифр маршрутной перестановки)

Автор

Назарова Софья

Дата

2025

7.8 route.h

См. документацию.

```
00001
00008 #pragma once
00009 #include <vector>
00010 #include <string>
00011 #include <stdexcept>
00012 #include <algorithm>
00013 using namespace std;
00014
00019 class cipher_error: public invalid_argument {
00020     public:
00025         explicit cipher_error (const string& what_arg):
00026             invalid_argument(what_arg) {}
00027
00032         explicit cipher_error (const char* what_arg):
00033             invalid_argument(what_arg) {}
00034 };
00035
00042 class code {
00043     private:
00044         int key;
00045
00053         inline int getValidKey(int key, const string& Text);
00054
00061         inline string getValidOpenText(const string& s);
00062
00070         inline string getValidCipherText(const string& s, const string& open_text);
00071
00072     public:
00076         code() = delete;
00077
00083         code(int skey, string text);
00084
00090         string encryption(const string& text);
00091
00098         string transcript(const string& text, const string& open_text);
00099 };
```

7.9 Файл README.md

Предметный указатель