

Labi

Создано системой Doxygen 1.9.1



---

1 Иерархический список классов	1
1.1 Иерархия классов . . . . .	1
2 Алфавитный указатель классов	3
2.1 Классы . . . . .	3
3 Список файлов	5
3.1 Файлы . . . . .	5
4 Классы	7
4.1 Класс cipher_error . . . . .	7
4.2 Класс modAlphaCipher . . . . .	8
4.2.1 Методы . . . . .	8
4.2.1.1 convert() [1/2] . . . . .	8
4.2.1.2 convert() [2/2] . . . . .	9
4.2.1.3 encrypt() . . . . .	9
4.2.1.4 getValidCipherText() . . . . .	10
4.2.1.5 getValidKey() . . . . .	10
4.2.1.6 getValidOpenText() . . . . .	11
4.2.2 Данные класса . . . . .	11
4.2.2.1 numAlpha . . . . .	12
5 Файлы	13
5.1 Файл main.cpp . . . . .	13
5.1.1 Подробное описание . . . . .	14
5.2 Файл modAlphaCipher.cpp . . . . .	14
5.2.1 Подробное описание . . . . .	15
5.3 Файл modAlphaCipher.h . . . . .	15
5.3.1 Подробное описание . . . . .	16
Предметный указатель	17



# Глава 1

## Иерархический список классов

### 1.1 Иерархия классов

Иерархия классов.

std::invalid_argument	
cipher_error . . . . .	7
modAlphaCipher . . . . .	8



## Глава 2

# Алфавитный указатель классов

### 2.1 Классы

Классы с их кратким описанием.

<code>cipher_error</code>	7
<code>modAlphaCipher</code>	8





## Глава 3

# Список файлов

### 3.1 Файлы

Полный список документированных файлов.

<a href="#">main.cpp</a>	Интерфейс модуля main для шифрования методом Гронсвельда . . . . .	13
<a href="#">modAlphaCipher.cpp</a>	Файл шифрования для модуля Gronsfeld . . . . .	14
<a href="#">modAlphaCipher.h</a>	Заголовочный файл для шифрования методом Гронсвельда . . . . .	15

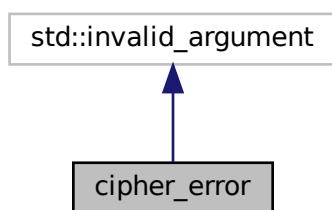


## Глава 4

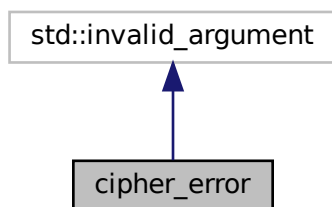
# Классы

### 4.1 Класс cipher\_error

Граф наследования: cipher\_error:



Граф связей класса cipher\_error:



### Открытые члены

- `cipher_error (const char *what_arg)`
- `cipher_error (const std::string &what_arg)`

Объявления и описания членов класса находятся в файле:

- [modAlphaCipher.h](#)

## 4.2 Класс modAlphaCipher

### 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)  
функция зашифрования
- std::wstring [decrypt](#) (const std::wstring &cipher\_text)  
функция расшифрования

### Закрытые члены

- std::vector< int > [convert](#) (const std::wstring &s)  
Преобразование строка-вектор
- std::wstring [convert](#) (const std::vector< int > &v)  
Преобразование вектор-строка
- std::wstring [getValidKey](#) (const std::wstring &s)  
Проверка и преобразование ключа
- std::wstring [getValidOpenText](#) (const std::wstring &s)  
Проверка и преобразование нормального текста
- std::wstring [getValidCipherText](#) (const std::wstring &s)  
Проверка зашифрованного текста

### Закрытые данные

- std::wstring [numAlpha](#)  
Алфавит русского языка, по порядку
- std::map< wchar\_t, int > [alphaNum](#)  
ассоциативный массив
- std::vector< int > [key](#)  
Ключ

### 4.2.1 Методы

#### 4.2.1.1 `convert()` [1/2]

```
std::wstring modAlphaCipher::convert (
    const std::vector< int > & v ) [inline], [private]
```

Преобразование вектор-строка

## Аргументы

in	v	Вектор числовых значений
----	---	--------------------------

## Возвращает

Строка из символов преобразование вектор-строка

## 4.2.1.2 convert() [2/2]

```
std::vector< int > modAlphaCipher::convert (
    const std::wstring & s ) [inline], [private]
```

## Преобразование строка-вектор

## Аргументы

in	s	Строка из символов
----	---	--------------------

## Возвращает

Вектор числовых значений преобразование строка-вектор

## 4.2.1.3 encrypt()

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

## функция зашифрования

## Зашифровывание

## Аргументы

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

## Возвращает

Зашифрованная строка

## Исключения

<code>cipher_error</code> , если	текст пустой
----------------------------------	--------------

4.2.1.4 `getValidCipherText()`

```
std::wstring modAlphaCipher::getValidCipherText (
    const std::wstring & s ) [inline], [private]
```

## Проверка зашифрованного текста

Текст проверяется на пустоту и наличие запрещённых символов.

## Предупреждения

Запрещёнными символами считаются все символы кроме букв русского языка

## Аргументы

in	s	Строка с введённым текстом
----	---	----------------------------

## Возвращает

Строка с проверенным текстом

## Исключения

<code>cipher_error</code> , если	текст пустой или содержит запрещённые символы проверка зашифрованного текста
----------------------------------	--

4.2.1.5 `getValidKey()`

```
std::wstring modAlphaCipher::getValidKey (
    const std::wstring & s ) [inline], [private]
```

## Проверка и преобразование ключа

Ключ проверяется на наличие запрещённых символов и пустоту и преобразуется. Строчные буквы преобразуются в заглавные

## Предупреждения

Запрещёнными символами считаются все символы кроме букв русского языка

## Аргументы

in	s	Строка с введённым ключом
----	---	---------------------------

## Возвращает

Строка с преобразованным ключом

## Исключения

<code>cipher_error</code> , если	ключ пустой, слабый или имеет недопустимые символы проверка и преобразование ключа
----------------------------------	--

## 4.2.1.6 getValidOpenText()

```
std::wstring modAlphaCipher::getValidOpenText (
    const std::wstring & s ) [inline], [private]
```

## Проверка и преобразование нормального текста

Текст проверяется на пустоту и преобразуется. Строчные буквы преобразуются в заглавные, запрещённые символы удаляются из текста

## Предупреждения

Запрещёнными символами считаются все символы кроме букв русского языка

## Аргументы

in	s	Строка с введённым текстом
----	---	----------------------------

## Возвращает

Строка с преобразованным текстом

## Исключения

<code>cipher_error</code> , если	текст пустой проверка и преобразование нормального текста
----------------------------------	---

## 4.2.2 Данные класса

#### 4.2.2.1 numAlpha

`std::wstring modAlphaCipher::numAlpha [private]`

Инициализатор

```
=  
L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"
```

Алфавит русского языка, по порядку

Объявления и описания членов классов находятся в файлах:

- [modAlphaCipher.h](#)
- [modAlphaCipher.cpp](#)



## Глава 5

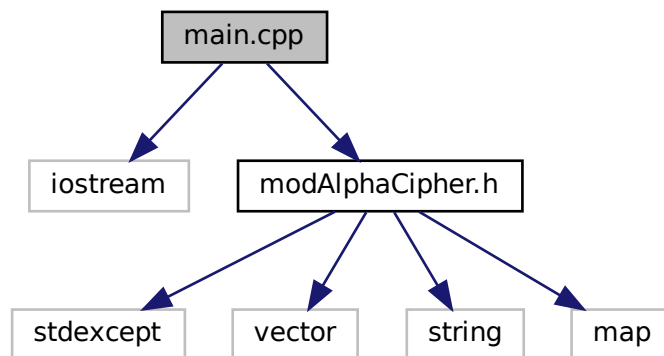
# Файлы

### 5.1 Файл main.cpp

Интерфейс модуля main для шифрования методом Гронсвельда

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

Граф включаемых заголовочных файлов для main.cpp:



### Функции

- void check (const wstring &Text, const wstring &key)
- int main (int argc, char \*\*argv)

### 5.1.1 Подробное описание

Интерфейс модуля main для шифрования методом Гронсвельда

Автор

Нефедов А.А.

Версия

1.0

Дата

17.06.2023

Авторство

ИБСТ ПГУ

Предупреждения

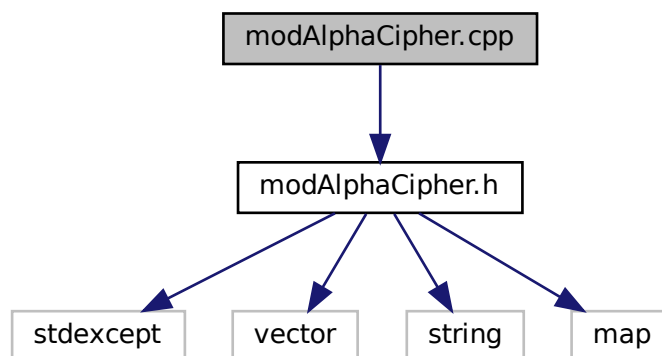
Это учебный пример

## 5.2 Файл modAlphaCipher.cpp

Файл шифрования для модуля Gronsfeld.

```
#include "modAlphaCipher.h"
```

Граф включаемых заголовочных файлов для modAlphaCipher.cpp:



### 5.2.1 Подробное описание

Файл шифрования для модуля Gronsfield.

Автор

Нефедов А.А.

Версия

1.0

Дата

17.06.2023

Авторство

ИБСТ ПГУ

Предупреждения

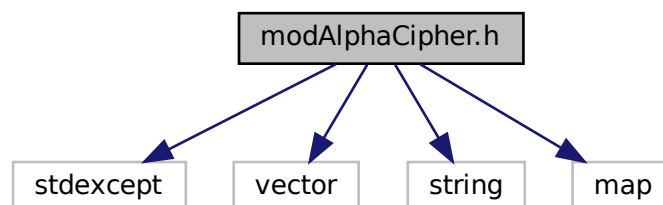
Это учебный пример

## 5.3 Файл modAlphaCipher.h

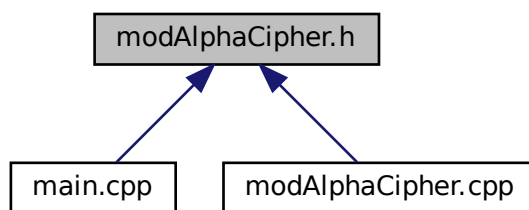
Заголовоачный файл для шифрования методом Гронсвельда

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

Граф включаемых заголовочных файлов для modAlphaCipher.h:



Граф файлов, в которые включается этот файл:



## Классы

- class `modAlphaCipher`
- class `cipher_error`

### 5.3.1 Подробное описание

Заголовоачный файл для шифрования методом Гронсвельда

Автор

Нефедов А.А.

Версия

1.0

Дата

17.06.2023

Авторство

ИБСТ ПГУ

Предупреждения

Это учебный пример

# Предметный указатель

- cipher\_error, [7](#)
- convert
  - modAlphaCipher, [8](#), [9](#)
- encrypt
  - modAlphaCipher, [9](#)
- getValidCipherText
  - modAlphaCipher, [10](#)
- getValidKey
  - modAlphaCipher, [10](#)
- getValidOpenText
  - modAlphaCipher, [11](#)
- main.cpp, [13](#)
- modAlphaCipher, [8](#)
  - convert, [8](#), [9](#)
  - encrypt, [9](#)
  - getValidCipherText, [10](#)
  - getValidKey, [10](#)
  - getValidOpenText, [11](#)
  - numAlpha, [11](#)
- modAlphaCipher.cpp, [14](#)
- modAlphaCipher.h, [15](#)
- numAlpha
  - modAlphaCipher, [11](#)