

Проект №1 (файл modAlphaCipher.h)  
1.0

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
4.1.1 Подробное описание . . . . .	8
4.2 Класс modAlphaCipher . . . . .	8
4.2.1 Подробное описание . . . . .	8
4.2.2 Конструктор(ы) . . . . .	8
4.2.2.1 modAlphaCipher() . . . . .	8
4.2.3 Методы . . . . .	9
4.2.3.1 decrypt() . . . . .	9
4.2.3.2 encrypt() . . . . .	9
5 Файлы	11
5.1 Файл modAlphaCipher.h . . . . .	11
5.1.1 Подробное описание . . . . .	12
Предметный указатель	13



# 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.

<a href="#">cipher_error</a>	Класс, предназначенный для обработки исключений . . . . .	7
<a href="#">modAlphaCipher</a>	Класс, который реализует шифрование методом "Гронсвельда" . . . . .	8





## Глава 3

# Список файлов

### 3.1 Файлы

Полный список документированных файлов.

<a href="#">modAlphaCipher.h</a>	
Описание класса <a href="#">modAlphaCipher</a>	11



## Глава 4

# Классы

### 4.1 Класс `cipher_error`

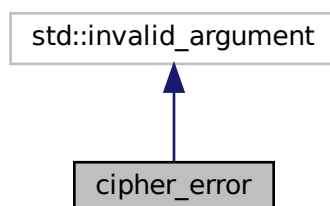
Класс, предназначенный для обработки исключений

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

Граф наследования: `cipher_error`:



Граф связей класса `cipher_error`:



## Открытые члены

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

### 4.1.1 Подробное описание

Класс, предназначенный для обработки исключений

Объявления и описания членов класса находятся в файле:

- [modAlphaCipher.h](#)

## 4.2 Класс modAlphaCipher

Класс, который реализует шифрование методом "Гронсвельда".

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

## Открытые члены

- [modAlphaCipher \(\)=delete](#)  
Запрещённый конструктор без параметров
- [modAlphaCipher \(const std::wstring &skey\)](#)  
Конструктор для ключа
- `std::wstring encrypt (const std::wstring &open_text)`  
Метод для шифрования
- `std::wstring decrypt (const std::wstring &cipher_text)`  
Метод, предназначенный для расшифрования

### 4.2.1 Подробное описание

Класс, который реализует шифрование методом "Гронсвельда".

Предупреждения

Работает только с русскоязычными сообщениями

### 4.2.2 Конструктор(ы)

#### 4.2.2.1 modAlphaCipher()

```
modAlphaCipher::modAlphaCipher (
    const std::wstring & skey )
```

Конструктор для ключа

Цикл `for` построен по строке-алфавиту и на каждом шаге добавляет в ассоциативный массив символ и его номер.

```
for (unsigned i=0; i<numAlpha.size(); i++) {
    alphaNum[numAlpha[i]]=i;
}
```

## Аргументы

<code>std::wstring</code>	- ключ в виде строки
---------------------------	----------------------

## 4.2.3 Методы

## 4.2.3.1 decrypt()

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

Метод, предназначенный для расшифрования

Здесь сначала формируется вектор `work` из строки шифротекста с помощью метода `convert()`. А также происходит проверка шифротекста на наличие ошибки при помощи метода `getValidAlphabetText()`.

```
vector<int> work = convert(getValidAlphabetText(cipher_text));
```

Если при зашифровывании мы прибавляли значение ключа, то при расшифровывании значения ключа надо вычитать. А чтобы не получить отрицательных значений, выполняется еще прибавление значения модуля, так как такое прибавление не влияет на результат модулю.

```
for(unsigned i=0; i < work.size(); i++) {
    work[i] = (work[i] + alphaNum.size() - key[i % key.size()]) % alphaNum.size();
}
```

## Аргументы

<code>std::wstring</code>	<code>cipher_text</code> - сообщение, которое нужно расшифровать
---------------------------	--

## Исключения

<code>cipher_error</code> , если	строка, которая поступила на вход пустая или в ней есть недопустимые символы
----------------------------------	--

## Возвращает

строка расшифрованного текста типа `"wstring"`

## 4.2.3.2 encrypt()

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

Метод для шифрования

Здесь сначала формируется вектор `work` из строки открытого текста с помощью метода `convert()`. А также происходит проверка текста на наличие ошибки при помощи метода `getValidAlphabetText()`.

```
vector<int> work = convert(getValidAlphabetText(open_text));
```

Затем в цикле к каждому элементу вектора прибавляется элемент ключа по модулю размера алфавита. Так как ключ может быть короче текста, то при индексации ключа выполняется операция по модулю размера ключа. Это позволяет использовать ключ циклически на длинных сообщениях.

```
for(unsigned i=0; i < work.size(); i++) {  
    work[i] = (work[i] + key[i % key.size()]) % alphaNum.size();  
}
```

Далее, при возврате значения, вектор `work` опять преобразуется в строку.

Аргументы

<code>std::wstring</code>	<code>open_text</code> - сообщение, которое нужно зашифровать
---------------------------	---

Исключения

<code>cipher_error</code> , если	строка, которая поступила на вход пустая или в ней есть недопустимые символы
----------------------------------	--

Возвращает

строка зашифрованного текста типа `"wstring"`

Объявления и описания членов класса находятся в файле:

- [modAlphaCipher.h](#)

## Глава 5

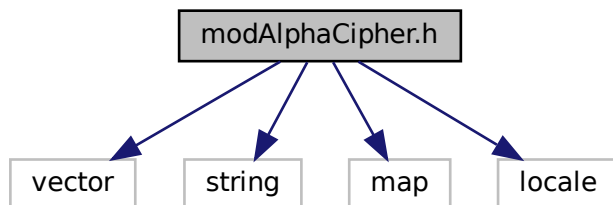
# Файлы

### 5.1 Файл modAlphaCipher.h

Описание класса `modAlphaCipher`.

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

Граф включаемых заголовочных файлов для `modAlphaCipher.h`:



### Классы

- class `modAlphaCipher`  
Класс, который реализует шифрование методом "Гронсвельда".
- class `cipher_error`  
Класс, предназначенный для обработки исключений

### 5.1.1 Подробное описание

Описание класса [modAlphaCipher](#).

Автор

Котунов А.А.

Версия

1.0

Дата

08.11.2022

Авторство

ИБСТ ПГУ



# Предметный указатель

`cipher_error`, [7](#)

`decrypt`

`modAlphaCipher`, [9](#)

`encrypt`

`modAlphaCipher`, [9](#)

`modAlphaCipher`, [8](#)

`decrypt`, [9](#)

`encrypt`, [9](#)

`modAlphaCipher`, [8](#)

`modAlphaCipher.h`, [11](#)