Grons

1.0

Создано системой Doxygen 1.9.8

Глава 1

Иерархический список классов

1.1 Иерархия классов

Иерархия классов.

Глава 2

Алфавитный указатель классов

2.1 Классы

Классы с их кратким описанием.

Глава 3

Список файлов

3.1 Файлы

Полный список документированных файлов.

Глава 4

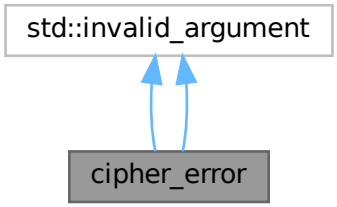
Классы

4.1 Класс cipher_error

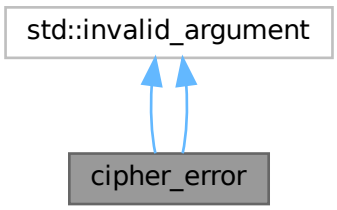
Класс исключений для ошибок шифрования

#include <modAlphaCipher.h>

Граф наследования:cipher_error:



Граф связей класса cipher_error:



Открытые члены

- cipher_error (const string &what_arg)
  Конструктор с строковым параметром
- cipher_error (const char *what_arg)
  Конструктор с C-строкой

4.1.1 Подробное описание

Класс исключений для ошибок шифрования

Используется для обработки ошибок, связанных с невалидными ключами, текстами и другими проблемами при работе с шифром Гронсвельда. Наследует функциональность std::invalid_argument.

4.1.2 Конструктор(ы)

4.1.2.1 cipher_error() [1/2]

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

Конструктор с строковым параметром

Аргументы

what_arg	Сообщение об ошибке в виде std::string

4.1.2.2 cipher_error() [2/2]

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

Конструктор с C-строкой

Аргументы

what_arg	Сообщение об ошибке в виде C-строки

Объявления и описания членов класса находятся в файле:

- modAlphaCipher.h

4.2 Класс modAlphaCipher

Класс для шифрования и расшифрования текста методом Гронсвельда

#include <modAlphaCipher.h>

Открытые члены

- modAlphaCipher ()=delete
  Удаленный конструктор по умолчанию
- modAlphaCipher (const wstring &skey)
  Конструктор с установкой ключа шифрования
- wstring encrypt (const wstring &open_text)
  Шифрует открытый текст
- wstring decrypt (const wstring &cipher_text)
  Расшифровывает зашифрованный текст

Закрытые члены

- vector< int > convert (const wstring &s)
  Преобразует строку в вектор числовых позиций
- wstring convert (const vector< int > &v)
  Преобразует вектор числовых позиций обратно в строку
- wstring getValidKey (const wstring &s)
  Проверяет и нормализует ключ шифрования
- wstring getValidOpenText (const wstring &s)
  Проверяет и нормализует открытый текст
- wstring getValidCipherText (const wstring &s)
  Проверяет валидность зашифрованного текста
- wstring to_upper (const wstring &s)
  Преобразует строку к верхнему регистру

Закрытые данные

- wstring numAlpha = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ"
  Русский алфавит по порядку (33 символа)
- map< wchar_t, int > alphaNum
  Ассоциативный массив "символ -> позиция в алфавите".
- vector< int > key
  Ключ шифрования в числовом представлении

4.2.1 Подробное описание

Класс для шифрования и расшифрования текста методом Гронсвельда

Реализует симметричное шифрование, где каждый символ открытого текста сдвигается на соответствующую позицию в ключе. Поддерживает только русский алфавит в верхнем регистре, включая букву Ё.

Алгоритм работы:

- Текст и ключ преобразуются в числовые представления
- Каждый символ текста сдвигается на позицию соответствующего символа ключа
- Используется модульная арифметика по размеру алфавита (33 символа)

Особенности:

- Автоматическое преобразование строчных букв в прописные
- Игнорирование не-русских символов при шифровании
- Строгая валидация входных данных

4.2.2 Конструктор(ы)

4.2.2.1 modAlphaCipher() [1/2]

modAlphaCipher::modAlphaCipher ( ) [delete]

Удаленный конструктор по умолчанию

Класс требует обязательной установки ключа при создании

4.2.2.2 modAlphaCipher() [2/2]

modAlphaCipher::modAlphaCipher (
const wstring & skey)

Конструктор с установкой ключа шифрования

Аргументы

skey	Ключ шифрования в виде wide string

Исключения

cipher_error	Если ключ невалиден

Пример использования:
```
modAlphaCipher cipher(L"ПАРОЛЬ"); // Создание шифратора с ключом "ПАРОЛЬ"
```

Аргументы

skey	Ключ шифрования в виде wide string

Исключения

cipher_error	Если ключ невалиден

Инициализирует таблицу соответствия символов и выполняет валидацию ключа.

См. также

getValidKey

4.2.3 Методы

4.2.3.1 convert() [1/2]

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

Преобразует вектор числовых позиций обратно в строку

Аргументы

v	Вектор позиций символов в алфавите

Возвращает

Строка, восстановленная из числовых позиций

Обратное преобразование для получения текстового представления после операций шифрования/расшифрования.

Аргументы

v	Вектор позиций символов в алфавите

Возвращает

Строка, восстановленная из числовых позиций

Выполняет обратное преобразование - из числовых позиций в текстовые символы. Использует строку numAlpha для обратного преобразования.

4.2.3.2 convert() [2/2]

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

Преобразует строку в вектор числовых позиций

Аргументы

s	Входная строка для преобразования

Возвращает

Вектор целых чисел, представляющих позиции символов в алфавите

Внутренний метод, используемый для преобразования текста и ключа в числовое представление для выполнения операций шифрования.

Аргументы

s	Входная строка для преобразования

Возвращает

Вектор целых чисел, представляющих позиции символов в алфавите

Каждый символ строки заменяется его позицией в алфавите. Использует предварительно инициализированную таблицу alphaNum.

4.2.3.3 decrypt()

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

Расшифровывает зашифрованный текст

Аргументы

cipher_text	Текст для расшифрования

Возвращает

Исходный открытый текст

Исключения

cipher_error	Если текст пуст или содержит недопустимые символы

Алгоритм расшифрования:

1. Валидация шифртекста
2. Преобразование в числовые позиции
3. Применение обратного ключа: (позиция_шифртекста + 33 - позиция_ключа) mod 33
4. Преобразование обратно в строку

Аргументы

cipher_text	Текст для расшифрования

Возвращает

Исходный открытый текст

Исключения

cipher_error	Если текст пуст или содержит недопустимые символы

Выполняет обратное преобразование для восстановления исходного текста. Использует модульную арифметику для корректного обратного сдвига.

См. также

getValidCipherText

convert

4.2.3.4 encrypt()

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

Шифрует открытый текст

Аргументы

open_text	Текст для шифрования

Возвращает

Зашифрованная строка в верхнем регистре

Исключения

cipher_error	Если текст пуст или не содержит русских букв

Алгоритм шифрования:

1. Валидация и нормализация текста
2. Преобразование в числовые позиции
3. Применение ключа: (позиция_текста + позиция_ключа) mod 33
4. Преобразование обратно в строку

Аргументы

open_text	Текст для шифрования

Возвращает

Зашифрованная строка в верхнем регистре

Исключения

cipher_error	Если текст пуст или не содержит русских букв

Выполняет преобразование текста через алгоритм Гронсвельда. Ключ применяется циклически к каждому символу текста.

См. также

getValidOpenText

convert

4.2.3.5 getValidCipherText()

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

Проверяет валидность зашифрованного текста

Аргументы

s	Зашифрованный текст для проверки

Возвращает

Валидный зашифрованный текст

Исключения

cipher_error	Если текст пуст или содержит недопустимые символы

Требования к шифртексту:

- Должен содержать только русские буквы в верхнем регистре
- Должен включать букву Ё
- Не должен содержать пробелов, знаков препинания, цифр

Аргументы

s	Зашифрованный текст для проверки

Возвращает

Валидный зашифрованный текст

Исключения

cipher_error	Если текст пуст или содержит недопустимые символы

Проверяет что шифртекст содержит только русские прописные буквы, включая букву Ё. Не допускает наличие других символов.

4.2.3.6 getValidKey()

wstring modAlphaCipher::getValidKey (
const wstring & s) [private]

Проверяет и нормализует ключ шифрования

Аргументы

s	Исходный ключ

Возвращает

Валидный ключ в верхнем регистре

Исключения

cipher_error	Если ключ пуст или содержит недопустимые символы

Требования к ключу:

- Не должен быть пустым
- Должен содержать только русские буквы
- Автоматически преобразует строчные буквы в прописные

Аргументы

s	Исходный ключ

Возвращает

Валидный ключ в верхнем регистре

Исключения

cipher_error	Если ключ пуст или содержит недопустимые символы

Выполняет следующие проверки:

- Наличие ключа (не пустая строка)
- Соответствие символов русскому алфавиту
- Автоматическое преобразование строчных букв в прописные

4.2.3.7 getValidOpenText()

wstring modAlphaCipher::getValidOpenText (
const wstring & s) [private]

Проверяет и нормализует открытый текст

Аргументы

s	Исходный открытый текст

Возвращает

Валидный открытый текст в верхнем регистре

Исключения

cipher_error	Если текст пуст или не содержит русских букв

Особенности обработки:

- Не-русские символы игнорируются
- Все русские буквы преобразуются к верхнему регистру
- Поддерживаются как прописные, так и строчные русские буквы

Аргументы

s	Исходный открытый текст

Возвращает

Валидный открытый текст в верхнем регистре

Исключения

cipher_error	Если текст пуст или не содержит русских букв

Фильтрует текст, оставляя только русские буквы и преобразуя их к верхнему регистру. Игнорирует не-русские символы.

4.2.3.8 to_upper()

wstring modAlphaCipher::to_upper (
const wstring & s) [private]

Преобразует строку к верхнему регистру

Аргументы

s	Исходная строка

Возвращает

Строка в верхнем регистре

Преобразует русские строчные буквы в прописные, включая специальную обработку для буквы Ё.

Аргументы

s	Исходная строка

Возвращает

Строка в верхнем регистре

Преобразует русские строчные буквы в прописные, включая специальную обработку для буквы Ё. Не-русские символы остаются без изменений.

Объявления и описания членов классов находятся в файлах:

- modAlphaCipher.h
- modAlphaCipher.cpp

Глава 5

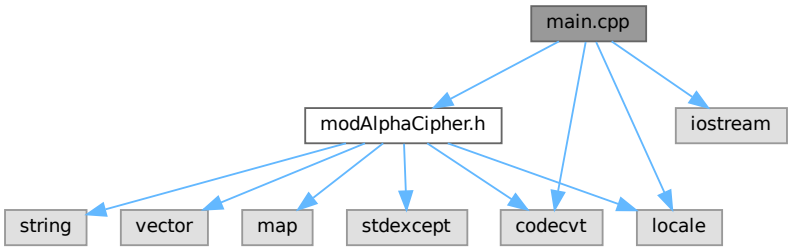
Файлы

5.1 Файл main.cpp

Демонстрационная программа для тестирования шифра Гронсвельда

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

Граф включаемых заголовочных файлов для main.cpp:



Функции

- wstring string_to_wstring (const string &str)

  Преобразует std::string в std::wstring.
- string wstring_to_string (const wstring &wstr)

  Преобразует std::wstring в std::string.
- void check (const string &Text, const string &key, const bool destructCipherText=false)

  Тестирует работу шифра с заданными параметрами
- int main ()

  Главная функция программы

5.1.1 Подробное описание

Демонстрационная программа для тестирования шифра Гронсвельда

Содержит функции для тестирования работы класса modAlphaCipher с различными входными данными и обработкой исключений.

Автор

Жбанчиков

Дата

2025

Версия

1.0

5.1.2 Функции

5.1.2.1 check()

```
void check (
            const string & Text,
            const string & key,
            const bool destructCipherText = false )
```

Тестирует работу шифра с заданными параметрами

Аргументы

Text	Открытый текст для шифрования
key	Ключ шифрования
destructCipherText	Флаг для тестирования обработки ошибок (опционально)

Выполняет полный цикл шифрования и расшифрования:

1. Проверяет входные данные
2. Создает объект шифратора
3. Шифрует текст
4. При необходимости модифицирует шифртекст для тестирования ошибок
5. Расшифровывает текст
6. Выводит результаты

Заметки

Если destructCipherText = true, модифицирует шифртекст для тестирования обработки некорректных данных при расшифровании.

5.1.2.2 main()

int main ( )

Главная функция программы

Возвращает

Код возврата: 0 при успешном выполнении

Выполняет серию тестов шифра с различными входными данными:

- Корректные тексты и ключи
- Пустые ключи
- Тексты с цифрами и английскими буквами
- Намеренно испорченные шифртексты

Тестовые сценарии покрывают основные случаи использования и обработку исключительных ситуаций.

5.1.2.3 string_to_wstring()

wstring string_to_wstring (
const string & str)

Преобразует std::string в std::wstring.

Аргументы

str	Строка для преобразования

Возвращает

Wide string в кодировке UTF-8

Используется для корректной работы с русскими символами.

5.1.2.4 wstring_to_string()

string wstring_to_string (
const wstring & wstr)

Преобразует std::wstring в std::string.

Аргументы

wstr	Wide string для преобразования

Возвращает

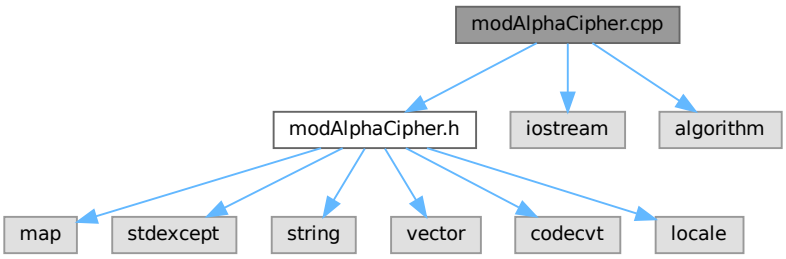
Строка в кодировке UTF-8

Используется для вывода wide string в стандартный поток.

5.2 Файл modAlphaCipher.cpp

Реализация класса шифрования методом Гронсвельда

#include "modAlphaCipher.h"
#include <iostream>
#include <algorithm>
Граф включаемых заголовочных файлов для modAlphaCipher.cpp:



Переменные

- wstring_convert< codecvt_utf8< wchar_t >, wchar_t > codec

  Глобальный конвертер для преобразования между std::string и std::wstring.

5.2.1 Подробное описание

Реализация класса шифрования методом Гронсвельда

Содержит реализацию всех методов класса modAlphaCipher. Включает преобразование текста, валидацию данных и алгоритмы шифрования/расшифрования.

Автор

Жбанчиков

Дата

2025

Версия

1.0

5.2.2 Переменные

5.2.2.1 codec

wstring_convert<codecvt_utf8<wchar_t>, wchar_t> codec

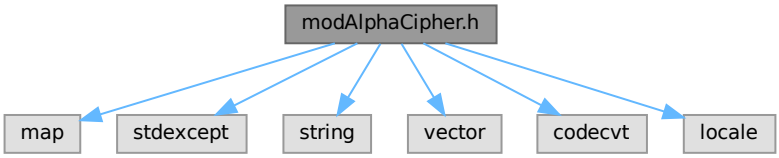
Глобальный конвертер для преобразования между std::string и std::wstring.

Используется для работы с русскими символами в UTF-8 кодировке. Обеспечивает корректное преобразование между многобайтовыми и широкими строками.

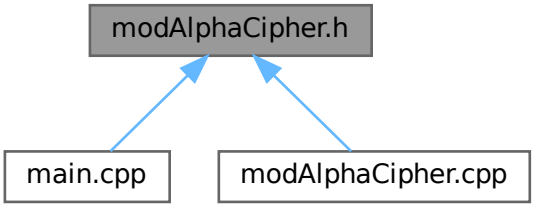
5.3 Файл modAlphaCipher.h

Заголовочный файл для класса шифрования методом Гронсвельда

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



Граф файлов, в которые включается этот файл:



Классы

- class cipher_error

  Класс исключений для ошибок шифрования
- class modAlphaCipher

  Класс для шифрования и расшифрования текста методом Гронсвельда

5.3.1 Подробное описание

Заголовочный файл для класса шифрования методом Гронсвельда

Реализует шифрование и расшифрование текста с использованием модифицированного шифра Гронсвельда для русского алфавита. Класс обеспечивает симметричное шифрование, где каждый символ открытого текста сдвигается на соответствующую позицию в ключе.

Автор

Анохин

Дата

2025

Версия

1.0

5.4 modAlphaCipher.h

См. документацию.

```
00001
00013 #pragma once
00014 #include <map>
00015 #include <stdexcept>
00016 #include <string>
00017 #include <vector>
00018 #include <codecvt>
00019 #include <locale>
00020 using namespace std;
00021
00031 class cipher_error : public invalid_argument
00032 {
00033 public:
00038     explicit cipher_error(const string& what_arg)
00039         : invalid_argument(what_arg)
00040     {
00041     }
00042
00047     explicit cipher_error(const char* what_arg)
00048         : invalid_argument(what_arg)
00049     {
00050     }
00051 };
00052
00071 class modAlphaCipher
00072 {
00073 private:
00074     wstring numAlpha = L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";
00075     map<wchar_t, int> alphaNum;
00076     vector<int> key;
00077
00086     vector<int> convert(const wstring& s);
00087
```

```
00096     wstring convert(const vector<int>& v);
00097
00109     wstring getValidKey(const wstring& s);
00110
00122     wstring getValidOpenText(const wstring& s);
00123
00135     wstring getValidCipherText(const wstring& s);
00136
00145     wstring to_upper(const wstring& s);
00146
00147 public:
00152     modAlphaCipher() = delete;
00153
00164     modAlphaCipher(const wstring& skey);
00165
00178     wstring encrypt(const wstring& open_text);
00179
00192     wstring decrypt(const wstring& cipher_text);
00193 };
```

Предметный указатель