Министерство науки и высшего образования РФ
ФГБОУ ВО ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Кафедра «Информационная безопасность систем и технологий»

ОТЧЁТ
о лабораторной работе №4
Документирование программы.

Дисциплина: Технологии и методы программирования
Группа:18ПИ1
Выполнил: Новиков Д.О.
Количество баллов:
Дата сдачи:
Проверил: к.т.н., доцент Лупанов М.Ю.

Пенза, 2019

1 Цель работы

1.1 Освоить документирование программы на языке C++ с использованием программы Doxygen.

2 Задания к практической работе

2.1 Сформировать блоки документирования для ранее разработанных модулей.

2.2 Сформировать документацию в форматах HTML и PDF.

3 Результат выполнения работы

3.1 Были сформированы блоки документирования для модулей modAlphaCipher и Perestanovka, разработанных в результате выполнения лабораторных работ № 1, № 2, № 3. Код заголовочного файла modAlphaCipher.h для модуля modAlphaCipher:

```
#pragma once
#include <vector>
#include <codecvt>
#include <string>
#include <map>
/** @file
 * @author Нестеров И.С.
 * @version 1.0
 * @date 11.06.2019
 * @copyright ИБСТ ПГУ
 * @warning Работа студента.
 * @brief Заголовочный файл для модуля modAlphaCipher
 */
class modAlphaCipher
{
private:
    std::wstring numAlpha =
                L"АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ";///<
Русский алфавит по порядку.
              std::map   <wchar_t,int>   alphaNum;///<
Ассоциативный массив "номер по символу"
    std::vector <int> key;///< Ключ
    /**
     * @brief Валидация ключа.
     * @param [in] s Ключ
```

```
         * @return Обработанный ключ.
         */
        std::wstring getValidKey(const std::wstring & s);
        /**
         * @brief Валидация открытого текста.
         * @param [in] s Открытый текст.
         * @return Обработынний открытый тест.
         */
        std::wstring getValidOpenText(const std::wstring
& s);
        /**
                * @brief Валидация текста, требующего
расшифровки.
         * @param [in] s Текст, требующий расшифровки.
         * @return Шифр-текст.
         */
                  std::wstring getValidCipherText(const
std::wstring & s);
        /**
         * @brief Преобразование "строка-вектор"
         * @param [in] s Строка, требущая конвертации в
целочисленный вектор.
         * @return Целочисленный вектор.
         */
        std::vector<int> convert(const std::wstring& s);
        /**
         * @brief Преобразование "вектор-строка.
         * @param [in] v Вектор, требующий преобразования
в строку.
         * @return Строка.
         */
        std::string convert(const std::vector<int>& v);
    public:
        /**
         *@brief Пустой конструктор для установки ключа.
         *@detail Конструктор запрещён.
         */
        modAlphaCipher()=delete;
        /**
         *@brief Конструктор для установки ключа.
         *@details Устанавливает ключ, с помощью которого
будет осуществляться шифрование и расшифрование.
         *@param [in] skey Строка-ключ. Должна состоять
из букв русского алфавита в верхнем регистре. Не должна
```

```
быть пустой. Все символы в нижнем регистре буду
автоматически преобразованы в верхний.
          *@throw cipher_error, если строка пустая или
содержит символы не русского алфавита или ключ
вырожденный.
         */
        modAlphaCipher(const std::wstring& skey);
        /**
              *@brief Метод шифрования текста шифром
Гронсфельда.
         *@param [in] open_text Открытый текст.Не должен
быть пустой строкой. Должен содержать только символы
русского алфавита. Строчные символы автоматически
преобразуются к прописным. Все не-буквы удаляются
         *@throw cipher_error, если строка пустая.
         */
              std::string encrypt(const std::wstring&
open_text);
        /**
              *@brief Метод шифрования текста шифром
Гронсфельда.
              *@param [in] open_text Текст, требующий
расшифровки.Не должен быть пустой строкой. Должен
содержать только символы русского алфавита в верхнем
регистре.
             *@throw cipher_error, если строка пустая,
содержит символы не русского алфавита или символы в
нижнем регистре.
         */
              std::string decrypt(const std::wstring&
cipher_text);
    };
    class cipher_error: public std::invalid_argument
    {
    public:
             explicit cipher_error (const std::string&
what_arg):
            std::invalid_argument(what_arg) {}
        explicit cipher_error (const char* what_arg):
            std::invalid_argument(what_arg) {}
    };
```

Код заголовочного файла Perestanovka.h для модуля Perestanovka:

```
/** @file
```

```
     * @author Нестеров И.С.
     * @version 1.0
     * @date 11.06.2019
     * @copyright ИБСТ ПГУ
     * @warning Работа студента.
     * @brief Заголовочный файл для модуля Perestanovka
     */
    #pragma once
    #include <string>
    #include <stdexcept>
    /** @brief Шифрование методом табличной маршрутной
перестановки.
     * @details Ключ устанавливается в конструкторе.
     * Для зашифровывания и расшифровывания предназначены
методы shifr и rashifr.
     * @warning Реализация только для английского языка.
     */
    class Perestanovka{
    private:
        int k;///< Ключ
        std::string getValidOpenText(const std::string &
s);///< Метод проверки открытого текста.
        std::string getValidCipherText(const std::string
& s);///< Метод проверки зашифрованного текста.
    public:
    /** @brief Конструктор без параметра.
     * @details Конструктор запрещён.
     */
        Perestanovka()=delete;
    /** @brief Конструктор для установки ключа.
     * @param [in] k - ключ, целое, положительное число.
     * @throw cipher_error, если ключ меньше или равен 1.
     */
        Perestanovka(const int k);
    /**
      *   @brief   Метод   для   шифрования   текста   методо
маршрутной табличной перестановки.
      *   @details   Запись   в   таблицу   происходит   слев
направо, сверху вниз. Считывание из таблицы сверху вниз,
справа налево
      * @param [in] t Открытый текст. Не должен быть
пустой строкой. Текст не должен быть меньше или равен
длине ключа. Все не-буквы будут автоматически удалены.
     * @return Зашифрованный текст.
```

```
     * @throw cipher_error, если строка пустая или меньше
или равна длине ключа.
     */
        std::string  shifr(const std::string& t);
    /**
     * @brief Метод для расшифровки зашифрованного текста
по известному ключу.
      *   @param   [in]   z   Зашифрованный   текст.   Должен
содержать только символы английского алфавита в верхнем
регистре. Строка не должна быть пустой.
     * @return Расшифрованный текст.
      *   @throw   cipher_error,   если   строка   пустая   или
встречена не английская буква в верхнем регистре.
     */
        std::string  rashifr(const std::string& z);
    };

    class cipher_error: public std::invalid_argument
    {
    public:
             explicit   cipher_error   (const   std::string&
what_arg):
            std::invalid_argument(what_arg) {}
        explicit cipher_error (const char* what_arg):
            std::invalid_argument(what_arg) {}
    };
```

3.2 Была сформирована документация в формате HTML и PDF. Документация для модулей modAlphaCipher и Perestanovka представлена в приложении А и приложении Б.

4. Вывод

В результате выполнения лабораторной работы были изучены основные возможности пакета Doxygen. Было освоено документирование в стиле Doxygen. Были получены практические навыки по редактированию конфигурационного файла, формированию документации в форматах HTML и PDF, созданию блоков документирования в программе.

Лабораторная работа 4

Нестеров И.С
Версия 1.1
Вт 18 Июн 2019

Оглавление

Table of contents

Иерархический список классов

Иерархия классов

Иерархия классов.

invalid_argument
- cipher_error 5
- cipher_error 5

modAlphaCipher 6
Perestanovka 8

Алфавитный указатель классов

Классы

Классы с их кратким описанием.

cipher_error 5
modAlphaCipher 6
Perestanovka (Шифрование методом табличной маршрутной перестановки) 8

Список файлов

Файлы

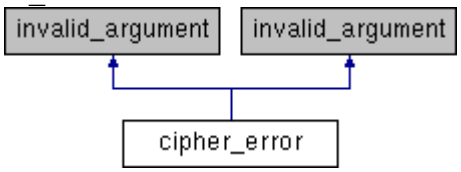
Полный список файлов.

modAplhaCipher.h (Заголовочный файл для модуля modAlphaCipher) 10
perestanovka.h (Заголовочный файл для модуля Perestanovka) 11

Классы

Класс cipher_error

`#include <modAplhaCipher.h>`
Граф наследования:cipher_error:



Открытые члены

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

Конструктор(ы)

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

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

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

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

Объявления и описания членов классов находятся в файлах:

- **modAplhaCipher.h**
- **perestanovka.h**

Класс modAlphaCipher

`#include <modAplhaCipher.h>`

Открытые члены

- **modAlphaCipher** ()=delete
  Пустой конструктор для установки ключа. @detail Конструктор запрещён.

- **modAlphaCipher** (const std::wstring &skey)
  Конструктор для установки ключа.

- std::string **encrypt** (const std::wstring &open_text)
  Метод шифрования текста шифром Гронсфельда.

- std::string **decrypt** (const std::wstring &cipher_text)
  Метод шифрования текста шифром Гронсфельда.

Конструктор(ы)

modAlphaCipher::modAlphaCipher () [delete]

Пустой конструктор для установки ключа. @detail Конструктор запрещён.

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

Конструктор для установки ключа.

Устанавливает ключ, с помощью которого будет осуществляться шифрование и расшифрование.

Аргументы:

in	*skey*	Строка-ключ. Должна состоять из букв русского алфавита в верхнем регистре. Не должна

быть пустой. Все символы в нижнем регистре будут автоматически преобразованы в верхний.

Исключения:

cipher_error*,если*	строка пустая или содержит символы не русского алфавита или ключ вырожденный.

Методы

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

Метод шифрования текста шифром Гронсфельда.

Аргументы:

in	*open_text*	Текст, требующий расшифровки.Не должен быть пустой строкой. Должен содержать только символы русского алфавита в верхнем регистре.

Исключения:

cipher_error*,если*	строка пустая, содержит символы не русского алфавита или символы в нижнем регистре.

std::string modAlphaCipher::encrypt (const std::wstring & *open_text*)

Метод шифрования текста шифром Гронсфельда.

Аргументы:

in	*open_text*	Открытый текст.Не должен быть пустой строкой. Должен содержать только символы русского алфавита. Строчные символы автоматически преобразуются к прописным. Все не-буквы удаляются

Исключения:

cipher_error*,если*	строка пустая.

Объявления и описания членов класса находятся в файле:

- **modAplhaCipher.h**

Класс Perestanovka

Шифрование методом табличной маршрутной перестановки.
`#include <perestanovka.h>`

Открытые члены

- **Perestanovka** ()=delete
  Конструктор без параметра.

- **Perestanovka** (const int k)
  Конструктор для установки ключа.

- std::string **shifr** (const std::string &t)
  Метод для шифрования текста методом маршрутной табличной перестановки.

- std::string **rashifr** (const std::string &z)
  Метод для расшифровки зашифрованного текста по известному ключу.

Подробное описание

Шифрование методом табличной маршрутной перестановки.

Ключ устанавливается в конструкторе. Для зашифровывания и расшифровывания предназначены методы shifr и rashifr.

Предупреждения:

Реализация только для английского языка.

Конструктор(ы)

Perestanovka::Perestanovka () `[delete]`

Конструктор без параметра.

Конструктор запрещён.

Perestanovka::Perestanovka (const int *k*)

Конструктор для установки ключа.

Аргументы:

in	*k*	- ключ, целое, положительное число.

Исключения:

cipher_error*,если*	ключ меньше или равен 1.

Методы

std::string Perestanovka::rashifr (const std::string & *z*)

Метод для расшифровки зашифрованного текста по известному ключу.

Аргументы:

in	*z*	Зашифрованный текст. Должен содержать только символы английского алфавита в верхнем регистре. Строка не должна быть пустой.

Возвращает:

Расшифрованный текст.

Исключения:

cipher_error*,если*	строка пустая или встречена не английская буква в верхнем регистре.

std::string Perestanovka::shifr (const std::string & *t*)

Метод для шифрования текста методом маршрутной табличной перестановки.

Запись в таблицу происходит слева направо, сверху вниз. Считывание из таблицы сверху вниз, справа налево

Аргументы:

in	*t*	Открытый текст. Не должен быть пустой строкой. Текст не должен быть меньше или равен длине ключа. Все не-буквы будут автоматически удалены.

Возвращает:

Зашифрованный текст.

Исключения:

cipher_error*,если*	строка пустая или меньше или равна длине ключа.

Объявления и описания членов класса находятся в файле:

- **perestanovka.h**

Файлы

Файл modAplhaCipher.h

Заголовочный файл для модуля **modAlphaCipher**.

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

Классы

- class **modAlphaCipher**
- class **cipher_error**

Подробное описание

Заголовочный файл для модуля **modAlphaCipher**.

Автор:
Нестеров И.С.

Версия:
1.0

Дата:
13.06.2019

Авторство:
ИБСТ ПГУ

Предупреждения:
Работа студента.

Файл perestanovka.h

Заголовочный файл для модуля **Perestanovka**.

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

Классы

- class **Perestanovka**
  Шифрование методом табличной маршрутной перестановки.
- class **cipher_error**

Подробное описание

Заголовочный файл для модуля **Perestanovka**.

Автор:
Нестеров И.С.

Версия:
1.0

Дата:
13.06.2019

Авторство:
ИБСТ ПГУ

Предупреждения:
Работа студента.

Алфавитный указатель

INDEX