Министерство образования и науки РФ

ФГБОУ ВО ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Кафедра «Информационная безопасность систем и технологий»

ОТЧЕТ

о лабораторной работе №1

СОЗДАНИЕ МНОГОМОДУЛЬНЫХ ПРОЕКТОВ НА ЯЗЫКЕ С++

Дисциплина: Технологии и методы программирования

Группа: 18ПИ1

Выполнил: Нестеров И.С.

Количество баллов:

Дата сдачи:

Принял: к.т.н., доцент Лупанов М. Ю.

Пенза 2019

1 Цель работы

1.1 Освоить процесс создания многомодульных проектов.

2 Задание к лабораторной работе

2.1 Модифицировать рассмотренную выше программу для работы с текстами на русском языке.

2.2 Разработать многомодульную программу, реализующую шифр табличной маршрутной перестановки. В качестве ключа взять количество столбцов таблицы. Маршрут записи: по горизонтали слева направо, сверху вниз. Маршрут считывания: сверху вниз, справа налево. При этом необходимо: Разработать UML-диаграмму вариантов использования. Спроектировать шифр табличной маршрутной перестановки в виде класса и построить для него диаграмму классов. Установку ключа выполнять в конструкторе. Разработать диаграммы деятельности (допускается разработка блок-схем алгоритмов вместо диаграмм деятельности) для методов зашифрования и расшифрования. Реализовать шифр табличной маршрутной перестановки в виде отдельного модуля. Спроектировать пользовательский интерфейс программы и реализовать его в главном модуле.

3 Результаты работы

3.1 Изменение в файле modAlphaCipher.h представлено на Рисунке 1. Полный текст модифицированной программы представлен в Приложении А.

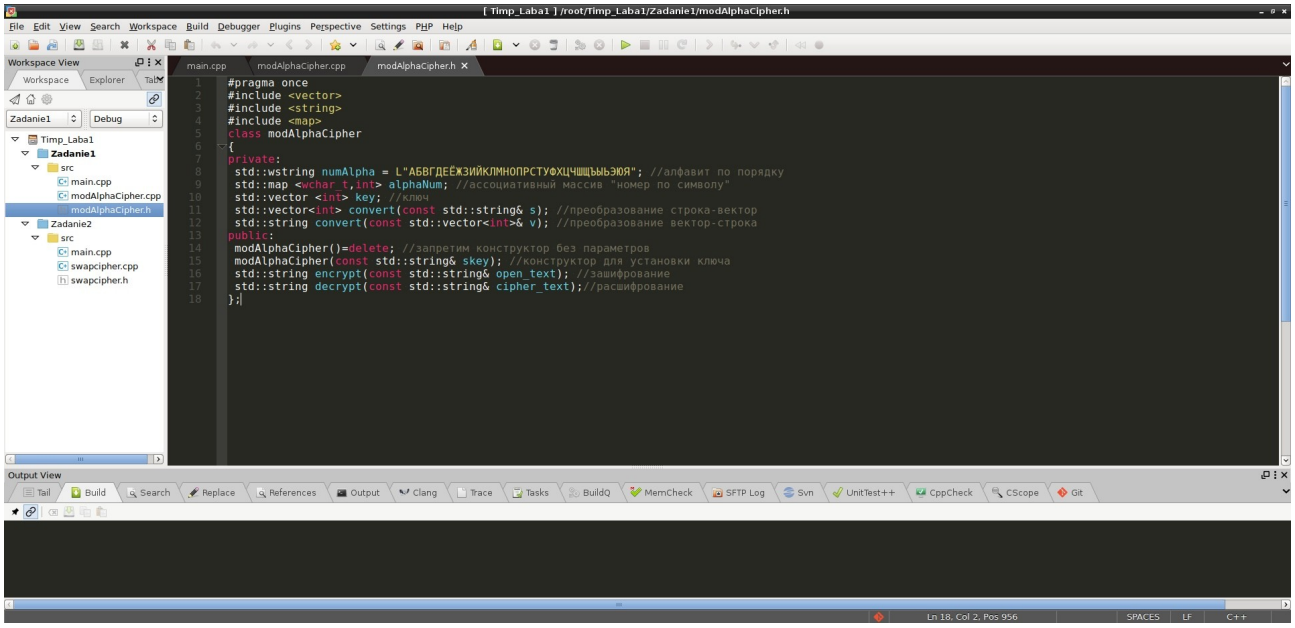

Рисунок 1 - файл modAlphaCipher.h

3.2 Файл swapcipher.h представлен на Рисунке 2. UML-диаграмма вариантов использования представлена на Рисунке 3. Диаграмма классов для шифрования представлена на Рисунке 4. Алгоритм работы методов зашифрования и расшифрования представлены на Рисунке 5 и Рисунке 6 соответственно. Полный текст модифицированной программы представлен в Приложении Б.

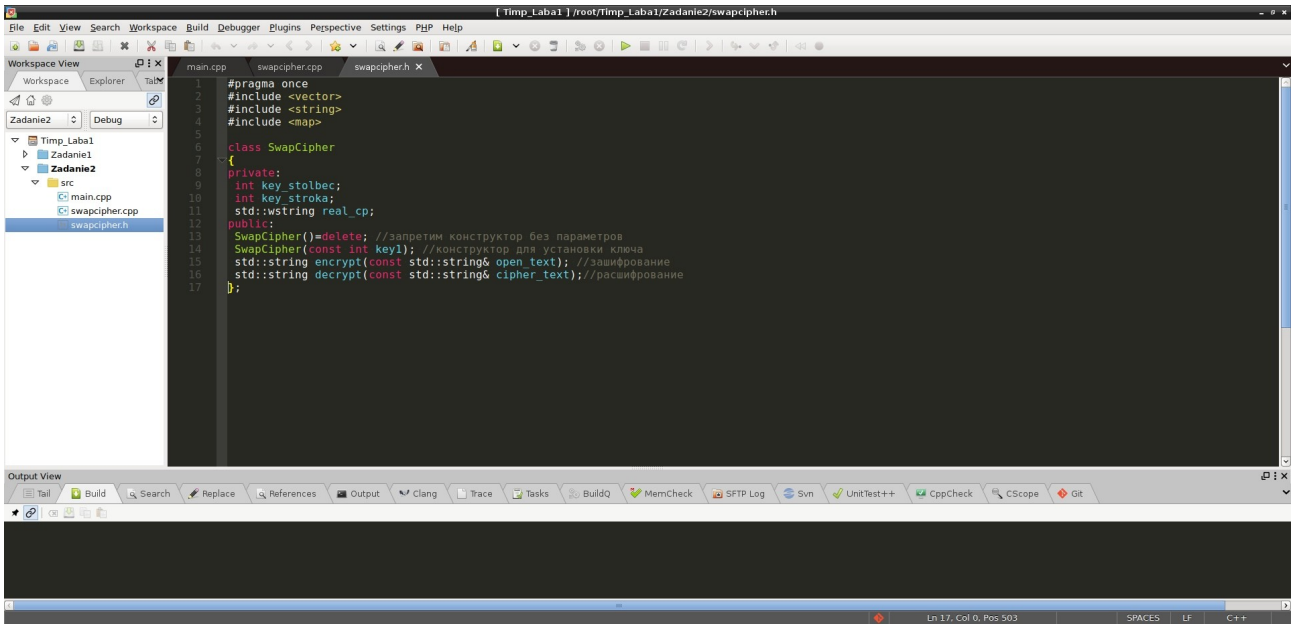

Рисунок 2 - файл swapcipher.h

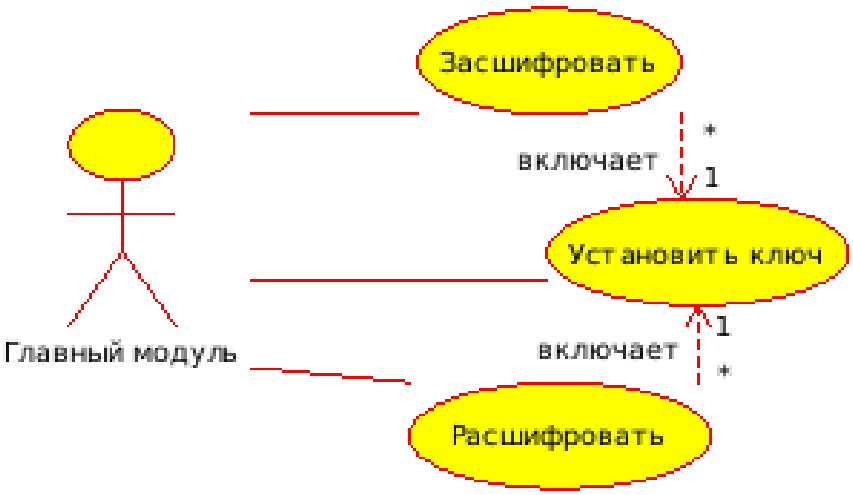

Рисунок 3 - UML-диаграмма вариантов использования

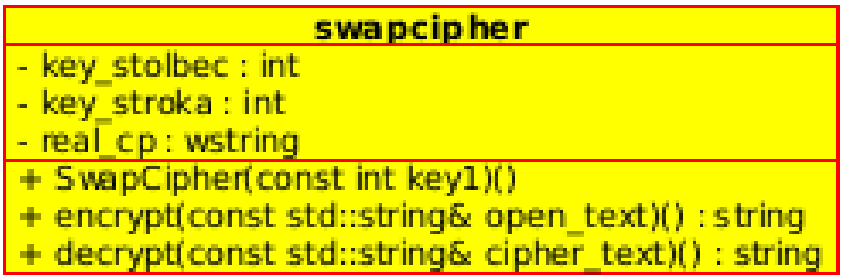

Рисунок 4 - Диаграмма классов

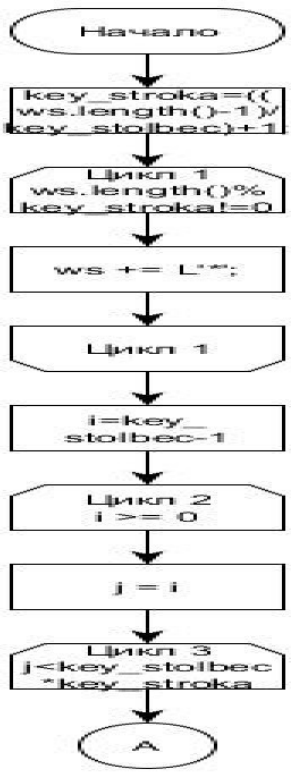

Рисунок 5 - Блок-схема для зашифрования

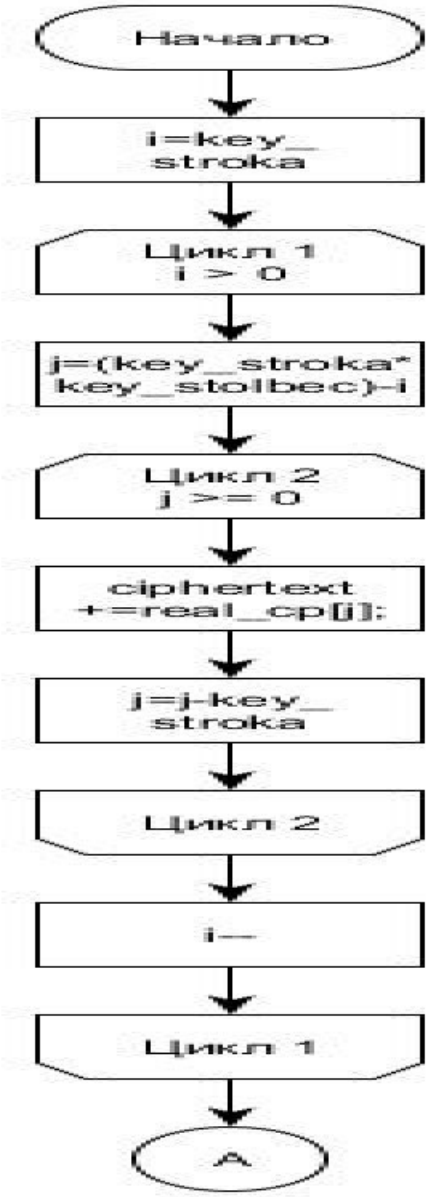

Рисунок 6 - Блок-схема для расшифрования

4 Вывод

В результате выполнения работы были изучены многомодульные проекты языка Си++, а также был разработан и написан многомодульный проект маршрутной перестановки, и получены практические навыки в создании многомодульных проектов.

Приложение А

Текст модифицированной программы

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

std::locale loc("ru_RU.UTF-8");
std::wstring_convert<std::codecvt_utf8<wchar_t>, wchar_t> codec;
using namespace std;
modAlphaCipher::modAlphaCipher(const std::string& skey)
{
 for (unsigned i=0; i < numAlpha.size(); i++)
    alphaNum[numAlpha[i]]=i;
 key = convert(skey);
}

std::string modAlphaCipher::encrypt(const std::string& open_text)
{
 std::vector<int> work = convert(open_text);
 for(unsigned i=0; i < work.size(); i++)
    work[i] = (work[i] + key[i % key.size()]) % alphaNum.size();
 return convert(work);
}

std::string       modAlphaCipher::decrypt(const       std::string&
cipher_text)
{
 std::vector<int> work = convert(cipher_text);
 for(unsigned i=0; i < work.size(); i++)
     work[i] = (work[i] + alphaNum.size() - key[i % key.size()]) %
alphaNum.size();
 return convert(work);
}
```

```
inline std::vector<int> modAlphaCipher::convert(const std::string&
s)
{
 std::wstring ws = codec.from_bytes(s); // перекодируем из UTF-8 в
UTF-32
 std::vector<int> result;
 for(auto c:ws)
    result.push_back(alphaNum[c]);
 return result;
}

inline std::string modAlphaCipher::convert(const std::vector<int>&
v)
{
 std::string result;
  std::wstring ws = codec.from_bytes(result); // перекодируем из
UTF-8 в UTF-32
 for(auto i:v)
    ws.push_back(numAlpha[i]);
 result = codec.to_bytes(ws);
 return result;
}
```

Приложение Б

Текст программы табличной маршрутной перестановки

```
#include "swapcipher.h"
#include <locale>
#include <codecvt>
std::locale loc("ru_RU.UTF-8");
std::wstring_convert<std::codecvt_utf8<wchar_t>, wchar_t> codec;

SwapCipher::SwapCipher(const int key1)
{
    key_stolbec = key1;
}

std::string SwapCipher::encrypt(const std::string& open_text)
{
    std::string result;
    std::wstring ws = codec.from_bytes(open_text);
    int i = 0, j = 0;
    key_stroka = ((ws.length() - 1) / key_stolbec) + 1;
    while(ws.length() % key_stroka != 0)
        ws += L'*';
    for (i = key_stolbec - 1; i >= 0; i--)
             for (j = i; j < key_stolbec*key_stroka; j = j +
key_stolbec)
            real_cp += ws[j];
    ws = L"";
    for (i = 0; i < key_stolbec*key_stroka; i++)
          if (((real_cp[i] >= L'A') && (real_cp[i] <= L'Я')) ||
real_cp[i] == L'Ё')
              ws += real_cp[i];
    result = codec.to_bytes(ws);
    return result;
}
```

```
std::string SwapCipher::decrypt(const std::string& cipher_text)
{
    std::string result;
    std::wstring ws;
    std::wstring ciphertext;
    int i = 0, j = 0;
    for (i = key_stroka; i > 0; i--)
         for (j = (key_stroka * key_stolbec) - i; j >= 0; j = j -
key_stroka)
            ciphertext += real_cp[j];
    for (i = 0; i < key_stolbec*key_stroka; i++)
          if (((ciphertext[i] >= L'A') && (ciphertext[i] <= L'Я'))
|| ciphertext[i] == L'Ё')
               ws += ciphertext[i];
    result = codec.to_bytes(ws);
    return result;
}
```