МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Лабораторная работа №2

по дисциплине

«Информатика и программирование»

Классы и шифры

Группа: **АБс-822**

Студент: **Соленый Н.Е.**

Преподаватель:

НОВОСИБИРСК

2019

**Задача:**

Реализовать интерфейс для теста алгоритмов шифрования на выбор пользователя. Реализовать и сами алгоритмы на C++, применив ООП. Для секретной информации использовать атрибуты private. Задачу выполнить для следующих алгоритмов: RSA, шифр Шамира, шифр Эль-Гамаля, система Диффи-Хеллмана.

UPD: В ходе работы помимо основной задачи была решена задача обмена данными через канал двойной связи (прямой канал и обратный), реализованный элементарно через два бинарных файла, задача не оказалась тривиальной: блок-схема и описание подводных камней прилагаются.

**Структура проекта**

**Интерфейс взаимодействия с пользователем**

#include <iostream>

#include <typeinfo>

#include "shifres.hpp"

using namespace std;

enum class Type {SHAMIR=1, DIFFI\_HELLMAN, RSA, EL\_GHAMAL};

template <typename Enumeration>

auto as\_integer(Enumeration const value) -> typename std::underlying\_type<Enumeration>::type {

return static\_cast<typename std::underlying\_type<Enumeration>::type>(value);

}

template<>

constexpr auto as\_integer<Type>(Type const value) -> typename std::underlying\_type<Type>::type {

int n = static\_cast<typename std::underlying\_type<Type>::type>(value);

if (n < 1 || n > 4)

throw std::runtime\_error("Incorrect input");

return n;

}

Type chooseEncode() {

int t;

inputerror:

cout << "1 - Shamir`s\n"

"2 - Diffi-Hellman`s\n"

"3 - RSA\n"

"4 - El-Ghamal`s\n"

"Enter type of simulated encryption algorythm: ";

cin >> t;

cout << endl;

try {

Type encode = static\_cast<Type>(t);

cout << as\_integer(encode) << endl;

return encode;

}

catch (std::exception &e) {

cerr << e.what() << endl;

goto inputerror;

}

}

void emulateRecipient(Type type) {

init("rec", "dis"); /// this is a crutch. it shouldn't be here, but we have it

switch (type) {

case Type::DIFFI\_HELLMAN: {

Diffi\_Hellman session;

cout << "\nMy shared key is " << session.sharedKey <<

". Just reporting, nothing more. I won`t tell you my secret key." << endl;

cout << "processing..." << endl;

session.recipient\_protocol();

cout << "\nI`m abonent B. My secret number is " << session.getEvaluatedNumber() <<

"; abonent A has equal secret number" << endl;

break;

}

case Type::EL\_GHAMAL: {

El\_Ghamal session;

session.recipient\_protocol();

break;

}

case Type::SHAMIR: {

Shamir session;

session.recipient\_protocol();

break;

}

case Type::RSA: {

RSA session;

session.recipient\_protocol();

break;

}

}

}

void emulateDispatcher(Type type) {

init("dis", "rec"); /// this is a crutch. it shouldn't be here, but we have it

switch (type) {

case Type::DIFFI\_HELLMAN: {

Diffi\_Hellman session;

cout << "\nMy shared key is " << session.sharedKey <<

". Just reporting, nothing more. I won`t tell you my secret key." << endl;

cout << "processing..." << endl;

session.dispatcher\_protocol();

cout << "\nI`m abonent A. My secret number is " << session.getEvaluatedNumber() <<

"; abonent B has equal secret number" << endl;

break;

}

case Type::EL\_GHAMAL: {

El\_Ghamal session;

session.dispatcher\_protocol();

break;

}

case Type::SHAMIR: {

Shamir session;

session.dispatcher\_protocol();

break;

}

case Type::RSA: {

RSA session;

session.dispatcher\_protocol();

break;

}

}

}

void chooseMode(Type type) {

int m;

err:

cout << "1 - Emulate recipient\n"

"2 - Emulate dispatcher to encrypt file (shared key of any recipient required)\n"

"Choose mode: ";

cin >> m;

cout << endl;

switch(m) {

case DISPATCHER:

emulateDispatcher(type);

break;

case RECIPIENT:

emulateRecipient(type);

break;

default:

goto err;

}

};

int main() {

Type type = chooseEncode();

chooseMode(type);

system("pause");

return 0;

}

Выбор, какого клиента эмулировать, отправителя или получателя. Выбор алгоритма для теста. Защита ввода. Пользователю нужно запустить два экземпляра /\*\*TODO сделат через терминал вз-ие \*\*/ программы и дождаться выполнения (передачи файла или удачного соединения).

**Описание классов алгоритмов через наследование**

**Реализация алгоритмов**

**Реализация канала двойной связи**

**Реализация блокировки файловых потоков**