

Информационная безопасность.

Лабораторная работа №7.

Подмогильный Иван Александрович.

Содержание

1. Цель работы	5
2. Задание	6
3. Выполнение лабораторной работы	7
4. Выводы	9

Список иллюстраций

3.1. Код 1	7
3.2. Код 2	7
3.3. Код 3	8
3.4. Код 4	8

Список таблиц

1. Цель работы

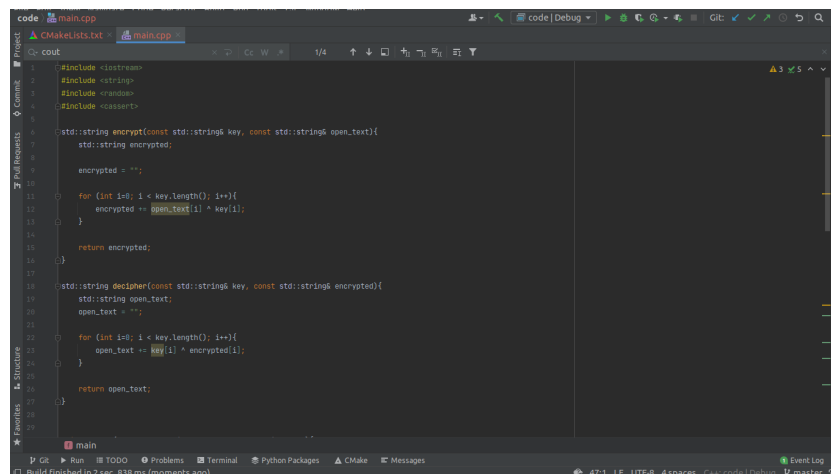
Освоить на практике применение режима однократного гаммирования

2. Задание

- 1) Выполнить пункты из задания по порядку.

3. Выполнение лабораторной работы

Написал код для зашифровки и дешифровки кодов.



```
code main.cpp
CMakeLists.txt main.cpp
cout
#include <iostream>
#include <string>
#include <random>
#include <cassert>

std::string encrypt(const std::string& key, const std::string& open_text){
    std::string encrypted;

    encrypted = "";

    for (int i=0; i < key.length(); i++){
        encrypted += open_text[i] * key[i];
    }

    return encrypted;
}

std::string decipher(const std::string& key, const std::string& encrypted){
    std::string open_text;

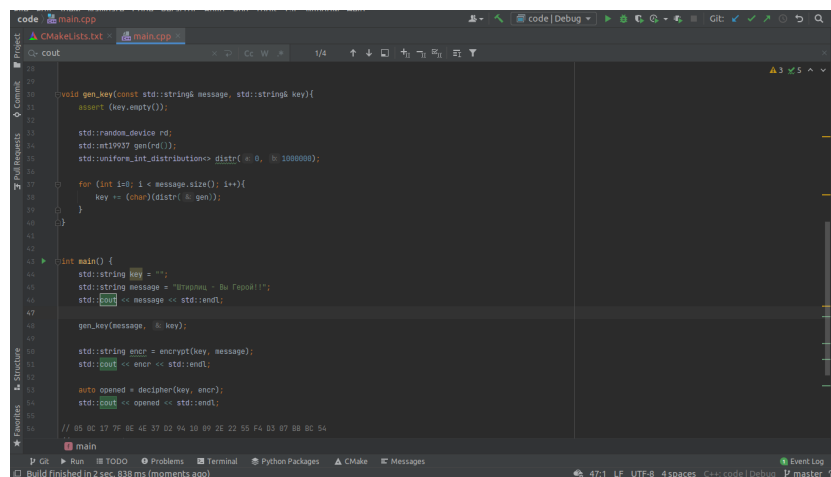
    open_text = "";

    for (int i=0; i < key.length(); i++){
        open_text += encrypted[i] * key[i];
    }

    return open_text;
}

main
GR Run TODO Problems Terminal Python Packages CMake Messages
Build finished in 2 sec, 838 ms (moments ago)
47:1 LF UTF-8 4spaces C++ code [Debug] master
```

Рис. 3.1.: Код 1



```
code main.cpp
CMakeLists.txt main.cpp
cout
void gen_key(const std::string& message, std::string& key){
    assert(!key.empty());

    std::random_device rd;
    std::mt19937 gen(rd());
    std::uniform_int_distribution<> distr(0, 1000000);

    for (int i=0; i < message.size(); i++){
        key += (char)distr(gen);
    }
}

int main() {
    std::string key = "";
    std::string message = "Привет - бу! (good!)";
    std::cout << message << std::endl;

    gen_key(message, key);

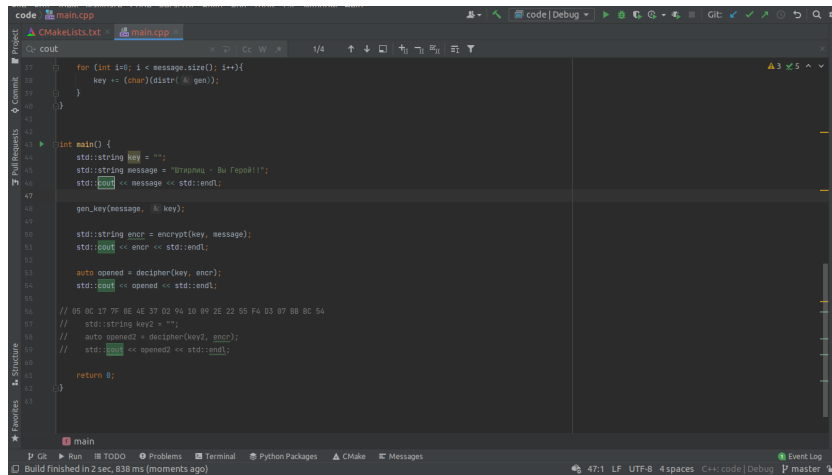
    std::string encr = encrypt(key, message);
    std::cout << encr << std::endl;

    std::string open = decipher(key, encr);
    std::cout << open << std::endl;

    // 05 06 17 7F 8E 4E 37 02 94 20 89 25 22 55 F4 03 07 88 8C 54

main
GR Run TODO Problems Terminal Python Packages CMake Messages
Build finished in 2 sec, 838 ms (moments ago)
47:1 LF UTF-8 4spaces C++ code [Debug] master
```

Рис. 3.2.: Код 2



```
code main.cpp
CMakeLists.txt main.cpp
cout
for (int i=0; i < message.size(); i++){
    key += (char)(distri(& gen));
}

int main() {
    std::string msg = "";
    std::string message = "Brupnea - Bu fepohl!";
    std::cout << message << std::endl;

    gen_key(message, & key);

    std::string encr = encrypt(key, message);
    std::cout << encr << std::endl;

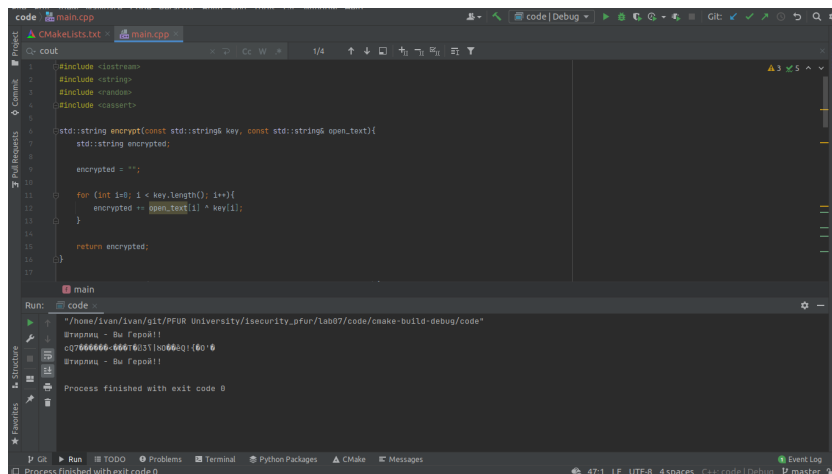
    auto opened = decipher(key, encr);
    std::cout << opened << std::endl;

    // 05 0C 17 7F 8E 4E 37 02 04 10 09 1E 22 55 F4 03 07 88 8C 94
    // std::string key2 = "";
    // auto opened2 = decipher(key2, encr);
    // std::cout << opened2 << std::endl;

    return 0;
}
```

Рис. 3.3.: Код 3

Результат кодировки и декодировки:



```
code main.cpp
CMakeLists.txt main.cpp
cout
#include <iostream>
#include <string>
#include <random>
#include <cassert>

std::string encrypt(const std::string key, const std::string open_text){
    std::string encrypted;

    encrypted = "";
    for (int i=0; i < key.length(); i++){
        encrypted += open_text[i] ^ key[i];
    }

    return encrypted;
}

int main() {
    // ... (code from previous image) ...
}
```

Run: code

"/home/ivan/ivan/git/PPUR University/Iscurity_pfur/lab07/code/cmake-build-debug/code"

Brupnea - Bu fepohl!

c076666666-6667631hc666q!60 6

Brupnea - Bu fepohl!

Process finished with exit code 0

Рис. 3.4.: Код 4

4. Выводы

Освоил на практике применение режима однократного гаммирования