



Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования

КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ

ИНСТИТУТ ФИЗИКИ

Криптографические методы защиты информации

Лабораторная работа №1

Программная реализация шифра замены

Выполнил
Глазков Андрей
студент 4 курса
группы 06-952

Казань – 2023

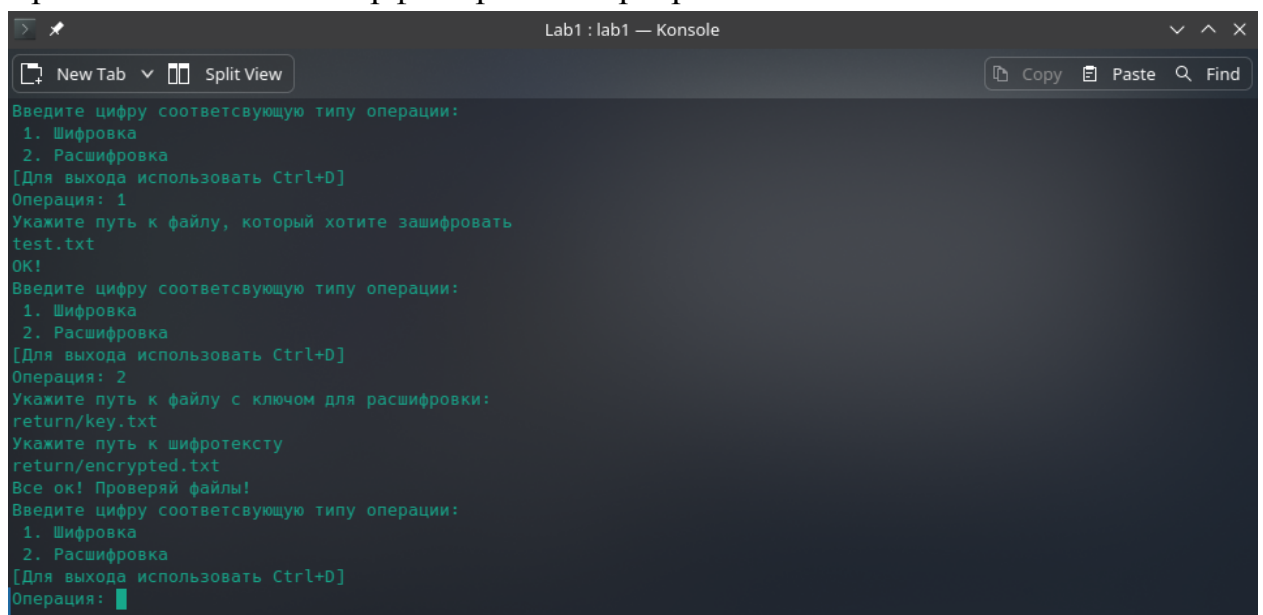
Цель работы

- Изучение шифрования и расшифрования текста с помощью шифра замены;
- Разработка программы, реализующая шифр замены;
- Расшифровка текста, выданного преподавателем, с помощью частотного криптоанализа;

Ход работы

Задание 1. Написать программу, реализующую шифр моноалфавитной замены.

1. Программа разработана на языке C++. Код программы представлен в Приложении №1. Интерфейс работы программы:



```
Lab1 : lab1 — Konsole
New Tab  Split View  Copy Paste Find
Введите цифру соответствующую типу операции:
1. Шифровка
2. Расшифровка
[Для выхода использовать Ctrl+D]
Операция: 1
Укажите путь к файлу, который хотите зашифровать
test.txt
OK!
Введите цифру соответствующую типу операции:
1. Шифровка
2. Расшифровка
[Для выхода использовать Ctrl+D]
Операция: 2
Укажите путь к файлу с ключом для расшифровки:
return/key.txt
Укажите путь к шифротексту
return/encrypted.txt
Все ок! Проверь файлы!
Введите цифру соответствующую типу операции:
1. Шифровка
2. Расшифровка
[Для выхода использовать Ctrl+D]
Операция: █
```

Рис. 1

2. Во время выполнения операции шифрования программа просит указать путь к шифруемому файлу и проверяет его наличие. Если такой существует, то происходит генерация пар алфавита замены, за это отвечают функция `generationReplAlphabet_` и вспомогательная функция `random_` (генерации рандомного числа в диапазоне от low до high):

```

std::random_device rd;

std::mt19937 gen(rd());

int random_(int low, int high)
{
    std::uniform_int_distribution<> dist(low, high);

    return (dist(gen));
}

std::map<wchar_t, wchar_t> generationReplAlphabet_()
{
    std::wstring alpha =
L"ЙФЯЦЫЧУВСКАМЕПИНРТГОЬШЛБЩДЗЖЮЭХЪабвгдежзийклмнопрстуфхцчщъыьэюя ,.!?";

    std::map<wchar_t, wchar_t> retAlpha;

    for (auto c : alpha)
    {
        retAlpha[c] = L'0';
    }

    int sz_alpha = alpha.size() - 1;

    for (auto it = retAlpha.begin(), ite = retAlpha.end(); it != ite; ++it)
    {
        int index_remove_c = random_(0, sz_alpha);

        (*it).second = alpha[index_remove_c];

        alpha.erase(std::remove(alpha.begin(), alpha.end(),
alpha[index_remove_c]), alpha.end());

        sz_alpha--;
    }

    return retAlpha;
}

```

3. Далее программа создает файл с ключем key.txt в директории return. За это отвечает функция `keyFile_`

```
void keyFile_(std::map<wchar_t, wchar_t>& key)
{
    std::wofstream f("./return/key.txt", std::ios::trunc);

    if (f.is_open()) {
        f.imbue(std::locale("ru_RU.UTF-8"));

        for (auto curr : key) {
            f << curr.first << "-" << curr.second << std::endl;
        }

        f.close();
    }

    else {
        throw std::runtime_error("Не удалось создать файл для записи key.txt");
    }
}
```

4. Следующий этап является ключевым, на нем считываются данные с входного файла, происходит замена букв в соответствии с ключом и записывается в файл esnrypted.txt, в директории return. За это отвечает функция `cryptoFile_`:

```

void cryptoFile_(std::map<wchar_t, wchar_t>& key, std::wofstream&
file)
{
    std::wofstream f("./return/encrypted.txt", std::ios::trunc);

    if (f.is_open()) {
        f.imbue(std::locale("ru_RU.UTF-8"));

        std::wstring line;
        while(std::getline(file, line))
        {
            for (auto curr : line)
            {
                if (key.count(curr))
                {
                    f << key[curr];
                } else {
                    f << curr;
                }
            }
            f << std::endl;
        }
        f.close();
    } else {
        throw std::runtime_error("Не удалось открыть файл
encrypted.txt");
    }
}

```

5. Во время расшифровки требуется указать путь к файлу с ключом и шифротексту. Сначала считывается файл key.txt и на его основе генерируется map. За это отвечает функция `scanKey_`:

```

std::map<wchar_t, wchar_t> scanKey_()
{
    std::cout << "Укажите путь к файлу с ключом для расшифровки:\n";
    std::string keyPath;
    std::cin >> keyPath;

    std::wifstream f(keyPath);
    std::map<wchar_t, wchar_t> retKey;
    if (f.is_open()) {
        f.imbue(std::locale("ru_RU.UTF-8"));
        std::wstring line;
        while (std::getline(f, line)) {
            if (line.begin() != line.end() - 1) {
                retKey[*line.end() - 1] = *line.begin();
            }
        }
    }
    else {
        throw std::runtime_error("Не удалось открыть файл с ключевой комбинацией");
    }
    return retKey;
}

```

6. Последний этап расшифровки – произведение замены букв и запись в файл decrypted.txt, в директории return. За это отвечает функция `createDecrypted_`:

```

void createDecrypted_(std::map<wchar_t, wchar_t>& key) {
    std::cout << "Укажите путь к шифротексту\n";
    std::string pathTxt;
    std::cin >> pathTxt;

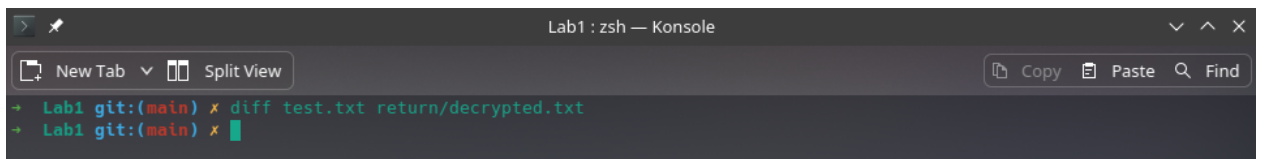
    std::wifstream fileIn(pathTxt);
    if (fileIn.is_open()) {
        fileIn.imbue(std::locale("ru_RU.UTF-8"));

        std::wofstream fileOut("./return/decrypted.txt",
std::ios::trunc);
        if (fileOut.is_open()) {
            fileOut.imbue(std::locale("ru_RU.UTF-8"));

            std::wstring line;
            while (std::getline(fileIn, line)) {
                for (auto it = line.begin(), ite = line.end(); it !=
ite; ++it) {
                    if (key.count(*it)) {
                        fileOut << key[*it];
                    } else {
                        fileOut << *it;
                    }
                }
            }
            fileOut.close();
        } else {
            throw std::runtime_error("Не удалось создать файл
decrypted.txt");
        }
        fileIn.close();
        std::cout << "Все ок! Проверяй файлы!\n";
    } else {
        throw std::runtime_error("Не удалось открыть исходный
файл");
    }
}

```

7. Результат работы (рис. 1) находится в приложениях №2-5. Проверили, что расшифрованный текст соответствует исходному:



```
Lab1 : zsh — Konsole
New Tab Split View
Copy Paste Find
+ Lab1 git:(main) * diff test.txt return/decrypted.txt
+ Lab1 git:(main) * 
```

Рис. 2

Вывод по 1 заданию

При выполнении работы была реализована программа, реализующая шифр замены. На примере произведения В.В. Маяковского “Во весь голос” была проверена её работоспособность. Зашифрованный текст совпал с расшифрованным.

Задание 2. Расшифровать текст, выданный преподавателем, с помощью частотного криптоанализа.

1. Написали скрипт на python для частотного анализа исходного текста (Приложение №6) и получили следующий результат:

```
→ text_analysis git:(main) x python inform.py
Введите имя файла: variant5.txt
Общее количество символов: 919
Количество символа 'г': 140 вероятность: 0.1523
Количество символа 'к': 82 вероятность: 0.0892
Количество символа 'ж': 71 вероятность: 0.0773
Количество символа 'ш': 67 вероятность: 0.0729
Количество символа 'б': 58 вероятность: 0.0631
Количество символа 'о': 54 вероятность: 0.0588
Количество символа 'д': 50 вероятность: 0.0544
Количество символа 'э': 49 вероятность: 0.0533
Количество символа 'у': 36 вероятность: 0.0392
Количество символа 'ъ': 34 вероятность: 0.037
Количество символа 'ю': 30 вероятность: 0.0326
Количество символа 'л': 30 вероятность: 0.0326
Количество символа 'ы': 29 вероятность: 0.0316
Количество символа 'я': 23 вероятность: 0.025
Количество символа 'х': 19 вероятность: 0.0207
Количество символа 'ф': 19 вероятность: 0.0207
Количество символа 'п': 16 вероятность: 0.0174
Количество символа 'щ': 13 вероятность: 0.0141
Количество символа 'с': 13 вероятность: 0.0141
Количество символа 'а': 12 вероятность: 0.0131
Количество символа 'н': 12 вероятность: 0.0131
Количество символа 'ц': 12 вероятность: 0.0131
Количество символа 'ч': 10 вероятность: 0.0109
Количество символа 'в': 8 вероятность: 0.0087
Количество символа 'т': 6 вероятность: 0.0065
Количество символа 'е': 5 вероятность: 0.0054
Количество символа ' ': 5 вероятность: 0.0054
Количество символа 'й': 4 вероятность: 0.0044
Количество символа 'з': 4 вероятность: 0.0044
Количество символа 'м': 4 вероятность: 0.0044
Количество символа 'р': 2 вероятность: 0.0022
Количество символа 'ь': 2 вероятность: 0.0022
→ text_analysis git:(main) x
```

рис. 3

$$H(x)=4.36$$

2. Самым часто встречающимся символом является пробел, заменили «Г» на пробел. Затем К заменили на О, т.к. является вторым по частоте. Проанализировав текст в поисках биграмм “НН”, “НО” сделали предположение, что Э – Н. На основе биграмм предположили, что Ш – Е и затем обнаружили триграмму НЕБ, соответственно заменили Б – Т.

3. Основываясь на интуиции и внимательном взгляде заметили схожесть в длине слова ТЕОЭЕТыЕдуОщ и известной части с словом ТЕОРЕТИЧЕСКОЙ. Поэтапно производя замены расшифровали текст:

Расшифрованный текст

СТУДЕНТЫ ФИЗИКО ТЕХНИЧЕСКОГО ИНСТИТУТА СИДЯТ НА СКУЧНЕЙШЕЙ ЛЕКЦИИ ПО ТЕОРЕТИЧЕСКОЙ МЕХАНИКЕ И КАЖДЫЙ ЗАНИМАЕТСЯ СВОИМИ ДЕЛАМИ МЫ НАПРИМЕР С ОДНОГРУППНИКОМ СИДЕЛИ И ЧИТАЛИ РАЗНЫЕ КОМИКСЫ ЛЕКТОР НА ЗАДНЕМ ФОНЕ ОТЧАЯННО РАСПИНАЕТСЯ ПИШЕТ ВСЮ ДОСКУ УЖЕ ИСПИСАЛ ВЫВОДОМ КАКОЙ ТО ФОРМУЛЫ ХОТЯ САМ ОН НЕ РАЗ НАМ ПРИЗНАВАЛСЯ ЧТО ВООБЩЕ ТО ОН МАТЕМАТИК В ТЕОРЕТИЧЕСКОЙ МЕХАНИКЕ НЕ СПЕЦИАЛИЗИРУЕТСЯ А ЭТОТ КУРС ЕГО СУДЬБА ЗАСТАВИЛА ЧИТАТЬ ТАК ВОТ НАДОЕЛО НАМ НАКОНЕЦ ЧИТАТЬ СВОИ КОМИКСЫ И ОДНОГРУППНИК МНЕ ПРЕДЛАГАЕТ А ДАВАЙ РАЗВЛЕЧЕМСЯ Я ЕСТЕСТВЕННО СОГЛАШАЮСЬ И ОН ГРОМКИМ УВЕРЕННЫМ ГОЛОСОМ СООБЩАЕТ ЛЕКТОРУ А У ВАС ВО ВТОРОЙ ФОРМУЛЕ ОШИБКА ЛЕКТОР РЕЗКО ЗАДРАВ ВВЕРХ ГОЛОВУ СО СЛОВАМИ ГДЕ ПЫТАЕТСЯ НА ИСПИСАННОЙ ОГРОМНОЙ ДОСКЕ НАЙТИ ЭТУ НЕСУЩЕСТВУЮЩУЮ ОШИБКУ ТАК ОН БЕДНЯГА МУЧАЛСЯ МИНУТ ПЯТЬ И НАКОНЕЦ БЕРЕТ ТРЯПКУ И СО СЛОВАМИ НЕТ РЕБЯТА ДАВАЙТЕ КА Я ЗАНОВО ПОПРОБУЮ СТИРАЕТ К ЧЕРТОВОЙ БАБУШКЕ ВСЮ СВОЮ ПОЛУТОРАЧАСОВУЮ РАБОТУ

4. Так как текст расшифровывался ручным трудом и ключ не был записан, то написали программу для получения ключа из двух текстов (Приложение №7). Полученный ключ:

```
→ text_analysis git:(main) ✖ cat pairs.txt
-б
а-Б
б-Т
в-Ю
г-
д-С
е-Х
ж-А
з-Ц
й-Щ
к-О
л-В
м-Ф
н-Э
о-И
п-Я
р-Э
с-Ч
т-Ш
у-К
ф-Д
х-П
ц-Г
ч-Ы
ш-Е
щ-Й
ь-Р
ы-У
ь-Ж
э-Н
ю-М
я-Л
```

рис. 4

Энтропия текста $H=4.35$ бит/симв.

Избыточность $R=1-H(M)/H_{\max}=1-4,35/5=0,13$;

Средняя длина слова $(N/N_{\text{с}}) - 1 = (919/140)-1= 5.5642$

Вывод

Энтропия расшифрованного текста составила 4,35 бит/симв., что соответствует показателю в теории (4,35 бит/симв.). Избыточность составила 0,13 является достаточно низким показателем и свидетельствует о высоком качестве шифра замены. На практике было доказано, что данный метод шифрования не является криптостойким.

Приложение №1

namespaces.hpp

```
#pragma once

# include <map>
# include <algorithm>
# include <fstream>
# include <iostream>

namespace encryption {

    void crypto();

} // encryption

namespace decryption {

    void decrypto();

} // decryption
```

main.cc

```
#include <iostream>

#include "../inc/namespaces.hpp"

int main()

{

    std::setlocale(LC_ALL, "ru_RU.UTF-8");

    std::cout << "\x1B[2J\x1B[H";

    while (std::cin.good())

    {

        std::cout << "Введите цифру соответствующую типу операции:\n 1. Шифровка\n 2. Расшифровка\n[Для выхода использовать Ctrl+D]\nОперация: ";

        int op = -1;

        std::cin >> op;

        if (op == 1) {

            encryption::crypto();

        } else if (op == 2) {

            decryption::decrypto();

        }

    }

}
```

```

    } else if (std::cin.good()) {

        std::cout << "\x1B[2J\x1B[HDруг, ну как же ты мог ошибиться?\n";

    } else {

        std::cout << "\x1B[2J\x1B[HДо встречи!\n";

    }

}

return 0;

}

```

encryption.cc

```

#include "../inc/namespaces.hpp"

#include <random>

namespace {

    std::random_device rd;

    std::mt19937 gen(rd());

    int random_(int low, int high)

    {

        std::uniform_int_distribution<> dist(low, high);

        return (dist(gen));

    }

    std::map<wchar_t, wchar_t> generationReplAlphabet_()

    {

        std::wstring alpha =
L"ЙФЯЦЧУВСКАМЕПИНРТГОЫШЛЩДЗЖЮЭХЪабвгдезийклмнопрстуфхцщтыъёя ,.!?";

        std::map<wchar_t, wchar_t> retAlpha;

        for (auto c : alpha)

        {

            retAlpha[c] = L'0';

        }

        int sz_alpha = alpha.size() - 1;

        for (auto it = retAlpha.begin(), ite = retAlpha.end(); it != ite; ++it)

```

```

    {
        int index_remove_c = random_(0, sz_alpha);

        (*it).second = alpha[index_remove_c];

        alpha.erase(std::remove(alpha.begin(), alpha.end(), alpha[index_remove_c]),
alpha.end());

        sz_alpha--;
    }

    return retAlpha;
}

void cryptoFile_(std::map<wchar_t, wchar_t>& key, std::wifstream& file)
{
    std::wofstream f("./return/encrypted.txt", std::ios::trunc);

    if (f.is_open()) {

        f.imbue(std::locale("ru_RU.UTF-8"));

        std::wstring line;

        while(std::getline(file, line))
        {
            for (auto curr : line)
            {
                if (key.count(curr))
                {
                    f << key[curr];
                } else {
                    f << curr;
                }
            }

            f << std::endl;
        }

        f.close();
    } else {

        throw std::runtime_error("Не удалось открыть файл encrypted.txt");
    }
}

```

```

void keyFile_(std::map<wchar_t, wchar_t>& key)
{
    std::wofstream f("./return/key.txt", std::ios::trunc);

    if (f.is_open()) {

        f.imbue(std::locale("ru_RU.UTF-8"));

        for (auto curr : key) {

            f << curr.first << "-" << curr.second << std::endl;

        }

        f.close();

    }

    else {

        throw std::runtime_error("Не удалось создать файл для записи key.txt");

    }

}

)

namespace encryption {

    void crypto()

    {

        std::cout << "Укажите путь к файлу, который хотите зашифровать\n";

        std::string path;

        std::cin >> path;

        std::wifstream file(path);

        file.imbue(std::locale("ru_RU.UTF-8"));

        if (file.is_open())

        {

            std::map<wchar_t, wchar_t> key = generationReplAlphabet_();

            try {

                keyFile_(key);

                cryptoFile_(key, file);

                file.close();

                std::cout << "OK!\n";

            } catch (std::exception& e) {

```

```

        std::cerr << e.what();

    }

    } else {

        std::cerr << "Файл не был открыт.";

    }

}

} // encryption

```

decryption.cc

```

#include "../inc/namespaces.hpp"

namespace {

    std::map<wchar_t, wchar_t> scanKey_()

    {

        std::cout << "Укажите путь к файлу с ключом для расшифровки:\n";

        std::string keyPath;

        std::cin >> keyPath;

        std::wifstream f(keyPath);

        std::map<wchar_t, wchar_t> retKey;

        if (f.is_open()) {

            f.imbue(std::locale("ru_RU.UTF-8"));

            std::wstring line;

            while (std::getline(f, line)) {

                if (line.begin() != line.end() - 1) {

                    retKey[*line.end() - 1] = *line.begin();

                }

            }

        }

        else {

            throw std::runtime_error("Не удалось открыть файл с ключевой комбинацией");

        }

        return retKey;

    }

}

```



```

void createDecrypted_(std::map<wchar_t, wchar_t>& key) {

    std::cout << "Укажите путь к шифротексту\n";

    std::string pathTxt;

    std::cin >> pathTxt;

    std::wifstream fileIn(pathTxt);

    if (fileIn.is_open()) {

        fileIn.imbue(std::locale("ru_RU.UTF-8"));

        std::wofstream fileOut("./return/decrypted.txt", std::ios::trunc);

        if (fileOut.is_open()) {

            fileOut.imbue(std::locale("ru_RU.UTF-8"));

            std::wstring line;

            while (std::getline(fileIn, line)) {

                for (auto it = line.begin(), ite = line.end(); it != ite; ++it) {

                    if (key.count(*it)) {

                        fileOut << key[*it];

                    } else {

                        fileOut << *it;

                    }

                }

            }

            fileOut.close();

        } else {

            throw std::runtime_error("Не удалось создать файл decrypted.txt");

        }

        fileIn.close();

        std::cout << "Все ок! Проверяй файлы!\n";

    } else {

        throw std::runtime_error("Не удалось открыть исходный файл");

    }

}

}

namespace decryption {

```

```
void decrypto()
{
    try {
        std::map<wchar_t, wchar_t> key = scanKey_();

        createDecrypted_(key);
    } catch (std::exception& e) {
        std::cerr << e.what();
    }
}
```

Приложение №2

Уважаемые товарищи потомки! Роясь в сегодняшнем окаменевшем говне, наших дней изучая потёмки, вы, возможно, спросите и обо мне. И, возможно, скажет ваш учёный, кроя эрудицией вопросов рой, что жил-де такой певец кипячёной И ярый враг воды сырой. Профессор, снимите очки-велосипед! Я сам расскажу о времени и о себе. Я, ассенизатор и водовоз, революцией мобилизованный и призванный, ушёл на фронт из барских садоводств поэзии — бабы капризной. Засадил садик мило, дочка, дачка, вода и гладь — сама садик я садила, сама буду поливать. Кто стихами льёт из лейки, кто кропит, набравши в рот — кудреватые Митрейки, мудреватые Кудрейки — кто их к чёрту разберёт! Нет на прорву карантина — мандолинят из-под стен: «Тара-тина, тара-тина, т-эн-н...» Неважная честь, чтоб из этаких роз мои изваяния высились по скверам, где харкает туберкулёз, где блядь с хулиганом да сифилис. И мне агитпроп в зубах навяз, и мне бы строчить романсы на вас — доходней оно и прелестней. Но я себя смирял, становясь на горло собственной песне. Слушайте, товарищи потомки, агитатора, горлана-главаря. Заглуша поэзии потоки, я шагну через лирические томики, как живой с живыми говоря. Я к вам приду в коммунистическое далекó не так, как песенно-есененный провитязь. Мой стих дойдёт через хребты веков и через головы поэтов и правительств. Мой стих дойдёт, но он дойдёт не так, — не как стрела в амурно-лировой охоте, не как доходит к нумизмату стёршийся пятак и не как свет умерших звёзд доходит. Мой стих трудом громаду лет прорвёт и явится весомо, грубо, зримо, как в наши дни вошёл водопровод, сработанный ещё рабами Рима. В курганах книг, похоронивших стих, железки строк случайно обнаруживая, вы с уважением ощупывайте их, как старое, но грозное оружие. Я ухо словом не привык ласкать; ушку девическому в завиточках волоска с полупохабщины не разалеться тронуту. Парадом развернув моих страниц войска, я прохожу по строчечному фронту, Стихи стоят свинцово-тяжело, готовые и к смерти и к бессмертной славе. Поэмы замерли, к жерлу прижав жерло нацеленных зияющих заглавий. Оружия любимейшего готовая рвануться в гике, застыла кавалерия острот, поднявши рифм отточенные пики. И все поверх зубов вооружённые войска, что двадцать лет в победах пролетали, до самого последнего листка я отдаю тебе, планеты пролетарий. Рабочего громады класса враг — он враг и мой, отъявленный и давний. Велели нам идти под красный флаг года труда и дни недоеданий. Мы открывали Маркса каждый том, как в доме собственном мы открываем ставни, но и без чтения мы разбирались в том, в каком идти, в каком сражаться стане. Мы диалектику учили не по Гегелю. Бряцанием боёв она врывалась в стих, когда под пулями от нас буржуи бегали, как мы когда-то бегали от них. Пускай за гениями безутешною вдовой плетётся слава в похоронном марше — умри, мой стих, умри, как рядовой, как безымянные на штурмах мёрли наши! Мне наплевать на бронзы многопудье, мне наплевать на мраморную слизь. Сочтёмся славою — ведь мы свои же люди, — пускай нам общим памятником будет построенный в боях социализм. Потомки, словарей проверьте поплавки: из Леты выплывут остатки слов таких, как «проституция», «туберкулёз», «блокада». Для вас, которые здоровы и ловки, поэт вылизывал чахоткины плевки шершавым языком плаката. С хвостом годов я становлюсь подобием чудовищ ископаемо-хвостатых. Товарищ жизнь, давай быстрее протопаем, протопаем по пятилетке дней остаток. Мне и рубля не накопили строчки, краснодеревщики не слали мебель на дом. И кроме свежевывытой сорочки, скажу по совести, мне ничего не надо. Явившись в

Це Ка Ка идущих светлых лет, над бандой поэтических рвачей и выжиг я подыму, как
большевистский партбилет, все сто томов моих партийных книжек.

Приложение №3

-Ж	Т-Р	й-в
!-з	У-Т	к-д
, -Ы	Ф-З	л-В
. -М	Х-л	м-Г
?-Э	Ц-п	н-Ь
А-П	Ч-с	о-я
Б-Я	Ш-ц	п-Ш
В-И	Щ-.	р-,
Г-Ы	Ъ-м	с-б
Д-Б	Ы-к	т-Е
Е-ч	Ь-т	у-Щ
Ж-й	Э-ж	ф-?
З-Ф	Ю-Й	х-У
И-Ч	Я-н	ц-ф
Й-р	а-г	ч-Ъ
К-ъ	б-А	ш-Д
Л-Ц	в-С	щ-Н
М-ш	г-!	ъ-Э
Н-щ	д-х	ы-е
О-ю	е-и	ь-Ь
П-а	ж-К	э-
Р-Х	з-Ю	ю-у
С-О	и-о	я-Л

Приложение №4

ТСгКгиГеиЖЕяСг,оНоЖШяЕяГдозЖХяЛбЬЖСЖби!яхьЛДьГЖядгГиьСДиГЖ!яСьиыЖ
ыгДоУЖхьивЖоЮЩЪгЛЖШяЕёГдоыЖСеыЖСяЮГяКьяыЖбШ,ябоЕиЖоЖяАяЖГьиМЖ
ЧыЖСяЮГяКьяыЖбдгКиЕЖСгДЖЩЪёьевыЖд,яЛЖ
,ЩхофоивЖСяШ,ябяСЖ,явыЖЪЕяЖКоВ-хиЖЕгдявЖШиСифЖдоШЛТЬёьавЖЧЖЛ,евЖС,
г!ЖСяхеЖбе,явМЖа,я?иббя,ыЖбьоГоЕиЖяЪдо-СиВябоШихзЖнЖбгГЖ,гббдгКЩЖяЖС,
иГиьЖоЖяЖбиАиМЖныЖгббьоЮГЕя,ЖоЖСяхяСяЮыЖ,иСяВуфоивЖГяАоВоЮяСгь
евЖоЖШ,оЮСгьевыЖЩДёВЖыгЖ?,яьЕЖоЮЖАг,бдоУЖбгхяСяхбЕСЖШя
ЮооЖ—ЖАгАеЖдгШ,оЮьявМЖФгбгхоВгЖбгходЖГоВяыЖхяЪдгыЖхгЪдгыЖСяхьЖо
Ж!ВгхьЖ—ЖбгГгЖбгходЖЛЖбгхоВгыЖбгГгЖАЩхЩЖШяВоСгЕьМЖЪЕяЖбЕоУгГоЖ
ВьёЕЖоЮЖВивдоыЖдЕяЖд,яШоЕыЖыгА,гСдоЖСЖ,яЕЖ—ЖдЩх,иСгЕеиЖшоЕ,ивдоы
ЖГЩх,иСгЕеиЖыЩх,ивдоЖ—ЖдЕяЖоУЖдЖЪё,ЕЩЖ,гЮАи,ёЕзЖщиЕЖыгЖШ,я,СЩЖ
дг,гьЕоыгЖ—ЖГгхяВоьЛЕЖоЮ-ШяхЖбЕиь:Ж«Рг,г-ЕоыгыЖЕг,г-ЕоыгыЖЕ-
ь-ь...»ЖщиСгКыгЛЖЪибЕыЖЪЕяАЖоЮЖ
ЕгдоУЖ,яЮЖГяоЖоЮСгЛьолЖСебоВобьЖШяЖбдСи,гГыЖ!хиЖУг,дгиЕЖЕЩАи,дЩВё
ЮыЖ!хиЖАВЛхьЖбЖУЩВо!гьяГЖхгЖбо?оВобМЖЧЖГыЖГ!оЕШ,яШЖСЖЮЩАгУ
ЖыгСЛЮыЖоЖГыЖАеЖбЕ,яЪоЕьЖ,яГгьбеЖыгЖСгбЖ—ЖхяУяхьивЖьяяЖоЖШ,иВиБ
ЕьивМЖщяЖЛЖбиАЛЖбГо,ЛВыЖбЕгьяСЛбьЖыгЖ!я,ВяЖбяАбЕСиььявЖШибьиМЖОВ
ЩДгвЕиыЖЕяСг,оНоЖШяЕяГдоыЖг!оЕГЕя,гыЖ!я,Вгыг-!ВгСг,ЛМЖФг!ВЩДгЖШя
ЮооЖШяЕядоыЖЛЖдГ!ыЩЖЪи,иОЖВо,оЪибдоиЖЕяГодоыЖдгдЖКоСявЖбЖКоСеГо
Ж!яСя,ЛМЖнЖдЖСгГЖШ,охЩЖСЖдяГГЩьобЕоЪибдяиЖхгВидяЖыиЖЕгдыЖдгдЖШ
ибиьья-ибиьибьевЖШ,яСоЕЛЮбМЖшявЖбЕоУЖхявхёЕЖЪи,иОЖУ,иАЕеЖСидяСЖоЖ
Ъи,иОЖ!яВяСеЖШя
ЕяСЖоЖШ,гСоЕиВьбЕСМЖшявЖбЕоУЖхявхёЕыЖьяЖьяЖхявхёЕЖыиЖЕгдыЖ—Жыи
ЖдгдЖбЕ,иВгЖСЖгГЩ,ья-Во,яСявЖяУяЕиыЖыиЖдгдЖхяУяхоЕЖдЖьЩГоЮГЕЩЖбЕ
ё,ДовбЛЖШЛЕгдЖоЖыиЖдгдЖбСиЕЖЩГи,ДоУЖЮСёЮхЖхяУяхоЕМЖшявЖбЕоУЖЕ,
ЩхяГЖ!,яГгхЩЖВиЕЖШ,я,СёЕЖоЖЛСоЕблЖСибяГыиЖ!,ЩАяыЖЮ,оГыиЖдгдЖСЖь
гДоЖхьЖСяДёВЖСяхяШ,яСяхыЖб,гАяЕгьевЖиНёЖ,гАгГоЖХоГгМЖИЖдЩ,!гыгУЖд
ьо!ыиЖШяУя,яьбСдоУЖбЕоУыЖКиВиЮдоЖбЕ,ядЖбВЩЪгвьяЖяАыг,ЩКоСгЛыЖСеЖб
ЖЩСгКиьоиГЖяНЩШеСгвЕиЖоУыЖдгдЖбЕг,яиыЖьяЖ!,яЮыиЖя,ЩКоиМЖнЖЩУя
ЖбВяСяГЖыиЖШ,оСедЖВгбдгЕь;ЖЩДдЩЖхиСоЪибдяГЩЖСЖЮгСоЕяЪдгУЖСяВяб
дгЖбЖШяВЩШяУгАНоьеЖыиЖ,гЮгВиЕьблЖЕ,яьЩЕЩМЖаг,гхяГЖ,гЮСи,ьЩСЖГяо
УЖбЕ,гьфЖСявбдгыЖЛЖШ,яУяКЩЖШяЖбЕ,яЪиЪьяГЩЖ?,яьЕЩыЖОЕоУоЖбЕяЛЕ
ЖбСоьфяСя-ЕЛКиВяыЖ!яЕяСеиЖоЖдЖбГи,ЕоЖоЖдЖАиббГи,ЕьявЖбВгСиМЖая
ГеЖЮгГи,ВоыЖдЖКи,ВЩЖШ,оКгСЖКи,ВяЖыгфиВиььеУЖЮолуНоУЖЮг!ВгСовМЖ
ю,ЩКолЖВуАогивДи!яЖ!яЕяСгЛЖ,СгыЩЕьблЖСЖ!одиыЖЮгбЕеВгЖдгСгВи,олЖяб
Е,яЕыЖШяхьЛСдоЖ,о?ГЖяЕЕяЪиььеиЖШодоМЖЧЖСбиЖШяСи,УЖЮЩАяСЖСяя,Щ
КёььеиЖСявбдгыЖЪЕяЖхСгхфгЕьЖВиЕЖСЖШяАихгУЖШ,яВиЕгВоыЖххяЖбГя!яЖШ
ябВихьи!яЖВобЕдгЖЛЖяЕхгуЖЕиАиыЖШВгыЕеЖШ,яВиЕг,овМЖХгАяЪи!яЖ!,яГгхе
ЖдВгббгЖС,г!Ж—ЖьяЖС,г!ЖоЖГявыЖяЕэлСВиььеВЖоЖхгСьовМЖИиВиВоЖыгГЖох
ЕоЖШяхЖд,гбьевЖ?Вг!Ж!яхгЖЕ,ЩхгЖоЖхьЖыиыхгьовМЖшеЖяЕд,еСгВоЖшг,дбг
ЖдгКхевЖЕяГыЖдгдЖСЖхяГиЖбяАбЕСиььяГЖГеЖяЕд,еСгГЖбЕгСьобьЖьяЖоЖАиЮ
ЖЪЕиьолЖГеЖ,гЮАо,гВобьЖСЖЕяГыЖСЖдгдгЖохЕоыЖСЖдгдгЖб,гКгЕьблЖбЕг
ьиМЖшеЖхогВидЕодЩЖЩЪоВоЖыиЖШяЖЫи!иВуМЖЯ,ЛфгьоиГЖАяёСЖяыгЖС,еСг
ВгбьЖСЖбЕоУыЖдя!хгЖШяхЖШЩВЛГоЖяЕЖыгЖАЩ,КЩоЖАи!гВоыЖдгдЖГеЖдя

!хг-ЕяЖАи!гВоЖяЕЖьоУМЖаЩбдгвЖЮгЖ!иьОЛГоЖАиЮЩЕиДьяуЖСхяСявЖШВиЕё
ЕбЛЖбВгСгЖСЖШяУя,яььяГЖГг,ДиЖ—ЖЩГ,оыЖГявЖбЕоУыЖЩГ,оыЖдгдЖ,ЛхяСяв
ыЖдгдЖАиЮеГЛьейЖыгЖДЕЩ,ГгУЖГё,ВоЖыгДозЖшыиЖыгШВиСгЕБЖыгЖА,яыЮеЖ
Гья!яШЩхьиыЖГыиЖыгШВиСгЕБЖыгЖГ,гГя,ьЩуЖбВоЮбМЖОяЪЕёГбЛЖбВгСяуж—
ЖСихЪЖГеЖбСяоЖКиЖВухыЖ—ЖШЩбдгвЖыгГЖяАНоГЖШгГЛЕьодяГЖАЩхиЕЖ
ШябЕ,яиььевЖСЖАяЛУЖбяфогВоЮбМЖаяЕяГдоыЖбВяСг,ивЖШ,яСи,ЬЕиЖШяШВгСд
о:ЖоЮЖЦиЕеЖСеШВеСЩЕЖябЕгЕдоЖбВяСЖЕгдоУыЖдгдЖ«Ш,ябЕоЕЩфол»ыж«Е
ЩАи,дЩВёЮ»ыж«АВядгхг»МЖБВЛЖСгбыЖдяЕя,еиЖЮхя,яСеЖоЖВяСдоыЖШя
ЕЖСеВоЮеСгВЖЪгУяЕдоьеЖШВиСдоЖДи,ДгСеГЖЛЮедяГЖШВгдгЕгМЖОЖУСябЕя
ГЖ!яхяСЖЛЖбЕгьяСВубЪЖШяхяАоиГЖЪЩхяСоНЖобдяШгиГя-УСябЕгЕеУМЖРяСг,о
НЖКоЮьЫЖхгСгвЖАебЕ,ивЖШ,яЕяШгиГыЖШ,яЕяШгиГЖШяЖШЛЕоВиЕдиЖхьивЖ
ябЕгЕядМЖшыиЖоЖ,ЩАВЛЖыиЖыгдяШоВоЖбЕ,яЪдоыЖд,гбьяхи,иСнодоЖыиЖбВгВо
ЖГиАиВЪЖыгЖхяГМЖЧЖд,яГиЖбСиКиСеГеЕявЖбя,яЪдоыЖбдгКЩЖШяЖбяСибеоыЖ
ГыиЖьоЪи!яЖыиЖыгхяМЖнСоСдобЪЖСЖпиЖыгЖыгЖохЩНоУЖбСиЕВеУЖВиЕыЖыгх
ЖАгъхявЖШя
ЕоЪибдоУЖ,СгЪивЖоЖСеКо!ЖЛЖШяхеГЩыЖдгдЖАяВЪДиСобЕбдовЖШг,ЕАоВиЕы
ЖСбиЖбЕяЖЕяГяСЖГяоУЖШг,ЕовьеУЖдьОкидМ

Приложение №5

Уважаемые товарищи потомки! Роясь в сегодняшнем окаменевшем говне, наших дней изучая потёмки, вы, возможно, спросите и обо мне. И, возможно, скажет ваш учёный, кроя эрудицией вопросов рой, что жил-де такой певец кипячёной И ярый враг воды сырой. Профессор, снимите очки-велосипед! Я сам расскажу о времени и о себе. Я, ассенизатор и водовоз, революцией мобилизованный и призванный, ушёл на фронт из барских садоводств поэзии — бабы капризной. Засадил садик мило, дочка, дачка, водь и гладь — сама садик я садила, сама буду поливать. Кто стихами льёт из лейки, кто кропит, набравши в рот — кудреватые Митрейки, мудреватые Кудрейки — кто их к чёрту разберёт! Нет на прорву карантина — мандолинят из-под стен: «Тара-тина, тара-тина, т-эн-н...» Неважная честь, чтоб из таких роз мои изваяния высились по скверам, где харкает туберкулёз, где блядь с хулиганом да сифилис. И мне агитпроп в зубах навяз, и мне бы строчить романсы на вас — доходней оно и прелестней. Но я себя смирял, становясь на горло собственной песне. Слушайте, товарищи потомки, агитатора, горлана-главаря. Заглуша поэзии потоки, я шагну через лирические томики, как живой с живыми говоря. Я к вам приду в коммунистическое далекó не так, как песенно-есененный провитязь. Мой стих дойдёт через хребты веков и через головы поэтов и правительств. Мой стих дойдёт, но он дойдёт не так, — не как стрела в амурно-лировой охоте, не как доходит к нумизмату стёршийся пятак и не как свет умерших звёзд доходит. Мой стих трудом громаду лет прорвёт и явится весомо, грубо, зримо, как в наши дни вошёл водопровод, сработанный ещё рабами Рима. В курганах книг, похоронивших стих, железки строк случайно обнаруживая, вы с уважением ощупывайте их, как старое, но грозное оружие. Я ухо словом не привык ласкать; ушку девическому в завиточках волоска с полупохабщины не разалеться тронуту. Парадом развернув моих страниц войска, я прохожу по строчечному фронту, Стихи стоят свинцово-тяжело, готовые и к смерти и к бессмертной славе. Поэмы замерли, к жерлу прижав жерло нацеленных зияющих заглавий. Оружия любимейшего готовая рвануться в гике, застыла кавалерия острот, поднявши рифм отточенные пики. И все поверх зубов

вооружённые войска, что двадцать лет в победах пролетали, до самого последнего листка я отдаю тебе, планеты пролетарий. Рабочего громады класса враг — он враг и мой, отъявленный и давний. Велели нам идти под красный флаг года труда и дни недоеданий. Мы открывали Маркса каждый том, как в доме собственном мы открываем ставни, но и без чтения мы разбирались в том, в каком идти, в каком сражаться стане. Мы диалектику учили не по Гегелю. Бряцанием боёв она врывалась в стих, когда под пулями от нас буржуи бегали, как мы когда-то бегали от них. Пускай за гениями безутешною вдовой плетётся слава в похоронном марше — умри, мой стих, умри, как рядовой, как безымянные на штурмах мёрли наши! Мне наплевать на бронзы многопудье, мне наплевать на мраморную слизь. Сочтёмся славою — ведь мы свои же люди, — пускай нам общим памятником будет построенный в боях социализм. Потомки, словарей проверьте поправки: из Леты выплывут остатки слов таких, как «проституция», «туберкулёз», «блокада». Для вас, которые здоровы и ловки, поэт вылизывал чахоткины плевки шершавым языком плаката. С хвостом годов я становлюсь подобием чудовищ ископаемо-хвостатых. Товарищ жизнь, давай быстрее протопаем, протопаем по пятилетке дней остаток. Мне и рубля не накопили строчки, краснодеревщики не слали мебель на́ дом. И кроме свежeweымытой сорочки, скажу по совести, мне ничего не надо. Явившись в Це Ка Ка идущих светлых лет, над бандой поэтических рвачей и выжиг я подыму, как большевистский партбилет, все сто томов моих партийных книжек.

Приложение №6

```
filename = input("Введите имя файла: ")

with open(filename, 'r') as f:

    text = f.read()

    size = len(text)

    print("Общее количество символов:", size)

    for c in sorted(set(text), key=lambda c: text.count(c), reverse=True):

        count_c = text.count(c)

        print("Количество символа '{}': {} вероятность: {}".format(c, count_c,
round((count_c/size),4)))
```

Приложение №7

```
#include <fstream>

#include <map>

#include <iostream>

int main(int argc, char** argv)

{

    std::wifstream f(argv[1]), fl(argv[2]);

    f.imbue(std::locale("ru_RU.UTF8")), fl.imbue(std::locale("ru_RU.UTF8"));

    std::map<wchar_t, wchar_t> pairs;

    std::wstring s, sl;

    while(std::getline(f,s) && std::getline(fl,sl)) {

        for (auto it = s.begin(), ite = s.end(), it1 = sl.begin(), itel = sl.end(); it
!= ite && it1 != itel; ++it, ++it1) {

            pairs[*it] = *it1;

        }

    }

    fl.close(), f.close();
```

```
std::wofstream fOut("pairs.txt", std::ios::trunc);

fOut.imbue(std::locale("ru_RU.UTF-8"));

for (auto&& curr : pairs) {

    fOut << curr.first << "-" << curr.second << std::endl;

}

fOut.close();

return 0;

}
```