

Министерство образования и науки РФ  
Санкт-Петербургский политехнический университет Петра Великого  
Институт компьютерных наук и кибербезопасности  
Высшая школа «Компьютерных технологий и информационных систем»

ОТЧЕТ  
по дисциплине «Практикум по программированию»  
**Лабораторная работа № 3**

**Выполнил:**

Студент гр. з5130902/20001

Д.Л. Рязанцев

**Проверил**

Ст. преподаватель

А.М. Журавская

Санкт-Петербург  
2024 г.

## Задание

В соответствии с вариантом **19**:

Считать текст из файла. Распечатать все первые слова предложений. Для первых двух слов посчитать количество вхождений в текст.

Исходные данные: файлы, приложенные к заданию.

## Код программы/Листинг программы

### CMakeLists.txt:

```
cmake_minimum_required(VERSION 3.0.0)
project(lab3
  DESCRIPTION "Лабораторная работа 3"
  HOMEPAGE_URL "github.com/whs31/education"
  LANGUAGES CXX
)

set(CMAKE_CXX_STANDARD 20)

add_executable(${PROJECT_NAME})
target_sources(${PROJECT_NAME} PRIVATE main.cc)
target_include_directories(${PROJECT_NAME} PRIVATE ${CMAKE_CURRENT_SOURCE_DIR})
```

### main.cc:

```

/*
 * Лабораторная работа 3
 * Студент: Рязанцев Дмитрий
 */

#include <cmath>
#include <concepts>
#include <iostream>
#include <format>
#include <regex>
#include <vector>
#include <fstream>
#include <filesystem>

using std::string;
using std::string_view;
using std::vector;
namespace fs = std::filesystem;

namespace
{
    [[nodiscard]] auto first_words_of_sentences(string const& text) -> vector<string>
    {
        using std::sregex_token_iterator;
        using std::sregex_iterator;
        using std::smatch;
        using std::regex;

        auto const pattern = regex(R"([\.\!\?\][\s]*([A-Z][\w]+)*)");
        auto result = vector<string>();
        result.push_back(text.substr(0, text.find_first_of(" \t\n") + 1));
        for(auto it = sregex_iterator(text.begin(), text.end(), pattern); it !=
sregex_iterator(); ++it)
            result.push_back(it->str(1));
        return result;
    }

    [[nodiscard]] auto count_word(string const& text, string_view const word) ->
size_t // NOLINT(*-no-recursion)
    {
        auto const pos = text.find(word.data());
        if(pos == string::npos)
            return 0;
        return 1 + count_word(text.substr(pos + word.size()), word);
    }
}

auto main() -> int
{
    using std::cin;

```

```

using std::cout;
using std::cerr;
using std::endl;

auto const path = []() -> fs::path {
    cout << "Enter path to text file: " << endl;
    auto p = fs::path();
    cin >> p;
    auto joined = fs::current_path() / p;
    if(not exists(joined)) {
        cerr << std::format("File {} doesn't exist", joined.string()) << endl;
        std::exit(1);
    }
    return joined;
}();

if(path.extension() != ".txt") {
    cerr << "File must have .txt extension" << endl;
    std::exit(1);
}

auto const text = [](fs::path const& filepath) -> string {
    using std::ifstream;
    using std::istreambuf_iterator;

    auto handle = ifstream(filepath);
    if(handle.fail()) {
        cerr << std::format("Can't open file {}", filepath.string()) << endl;
        std::exit(1);
    }
    return {istreambuf_iterator<char>(handle), istreambuf_iterator<char>()};
}(path);

auto is_all_whitespace = [](string const& word) -> bool { return
word.find_first_not_of(" \t\n") == string::npos; };
if(text.empty() or is_all_whitespace(text)) {
    cerr << "File is empty" << endl;
    std::exit(1);
}

auto join = [=](vector<string> const& words) -> string {
    auto result = string();
    for(auto const& word : words)
        result += word.empty() or is_all_whitespace(word) ? "" : word + " ";
    return result;
};

const auto [first, second] = [](string const& txt) -> std::pair<string, string>
{
    auto const pos = txt.find_first_of(" \t\n");
    auto const second_pos = txt.find_first_of(" \t\n", pos + 1);

```

```

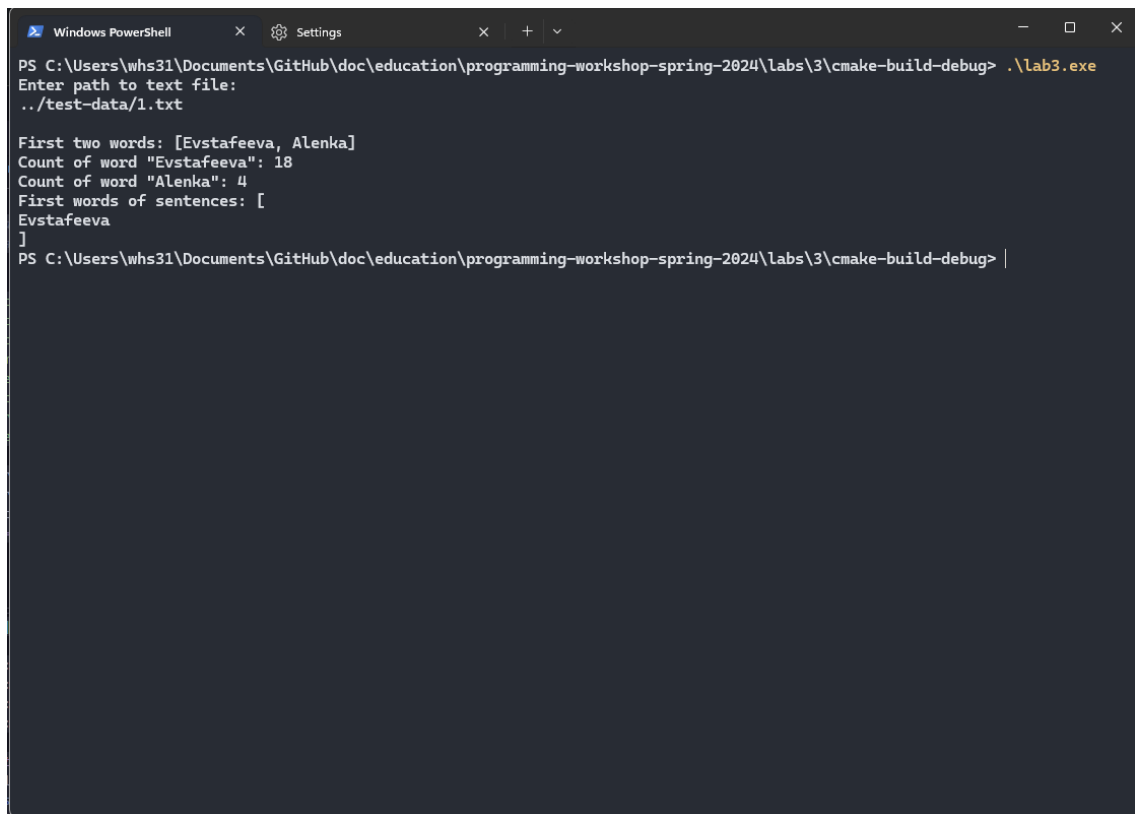
    if(pos == string::npos)
        return {"", ""};
    return {txt.substr(0, pos), txt.substr(pos + 1, second_pos - pos - 1)};
}(text.substr(text.find_first_not_of(" \t\n"), text.size() -
text.find_first_not_of(" \t\n"))));

if(first.empty() or second.empty()) {
    cerr << "File doesn't contain two words" << endl;
    std::exit(1);
}

cout << endl;
cout << std::format("First two words: [{}, {}]\n", first, second);
cout << std::format("Count of word \"{}\": {}\n", first, count_word(text,
first));
cout << std::format("Count of word \"{}\": {}\n", second, count_word(text, sec-
ond));
cout << std::format("First words of sentences: [\n{}\n]\n",
join(first_words_of_sentences(text)));
return 0;
}

```

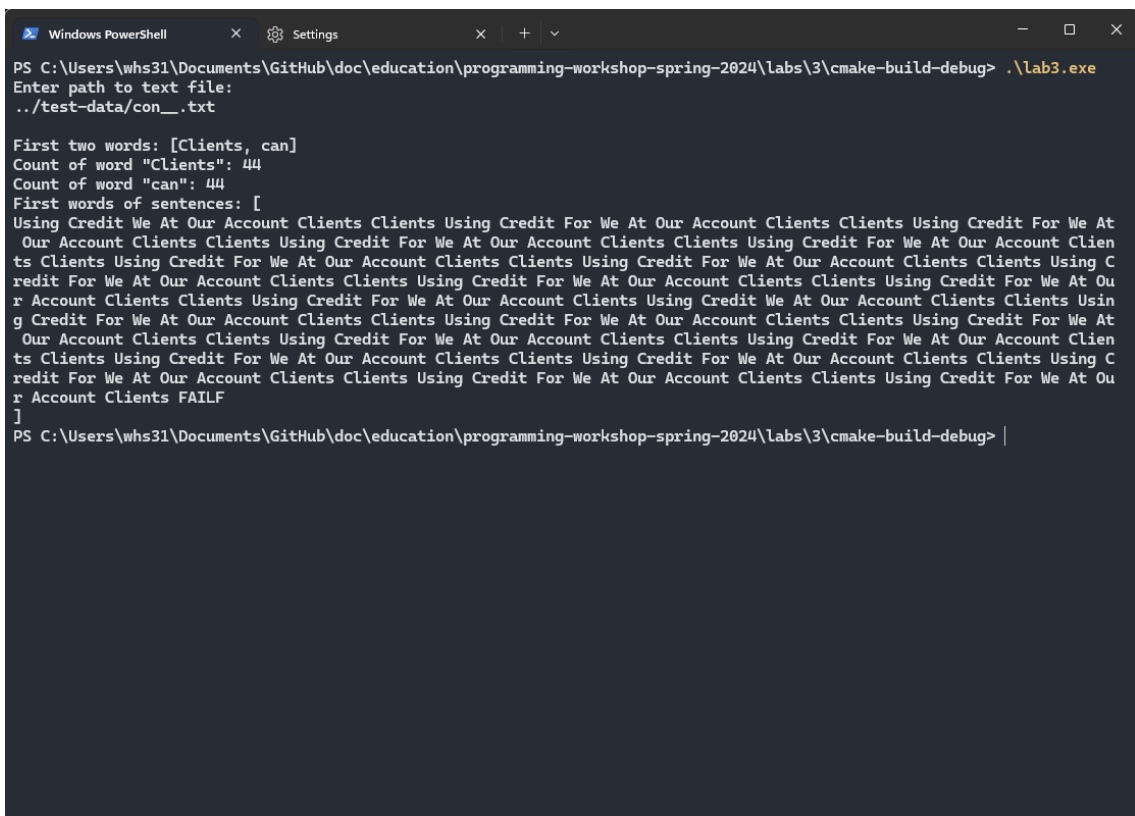
# Пример работы программы



```
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> .\lab3.exe
Enter path to text file:
../test-data/1.txt

First two words: [Evstafeeva, Alenka]
Count of word "Evstafeeva": 18
Count of word "Alenka": 4
First words of sentences: [
Evstafeeva
]
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> |
```

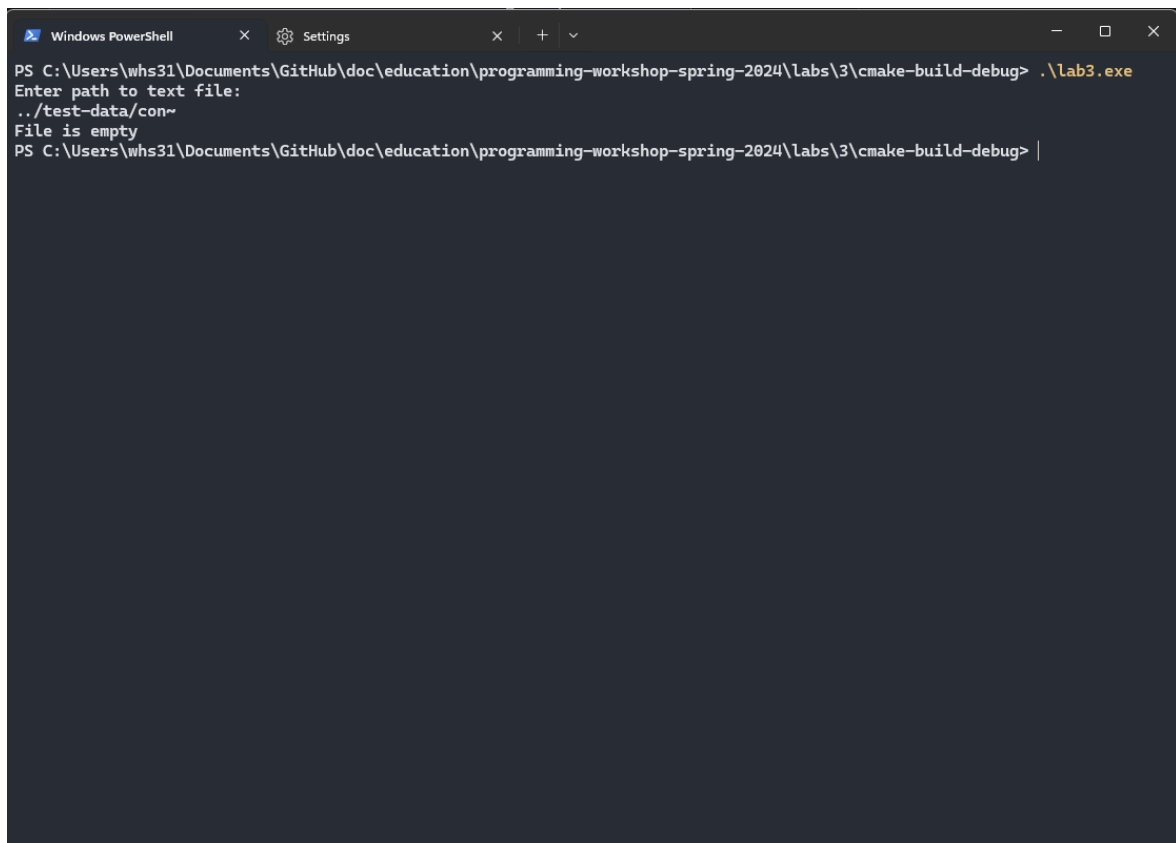
Рисунок 1 – Правильная работа программы на малом объеме текста



```
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> .\lab3.exe
Enter path to text file:
../test-data/con_.txt

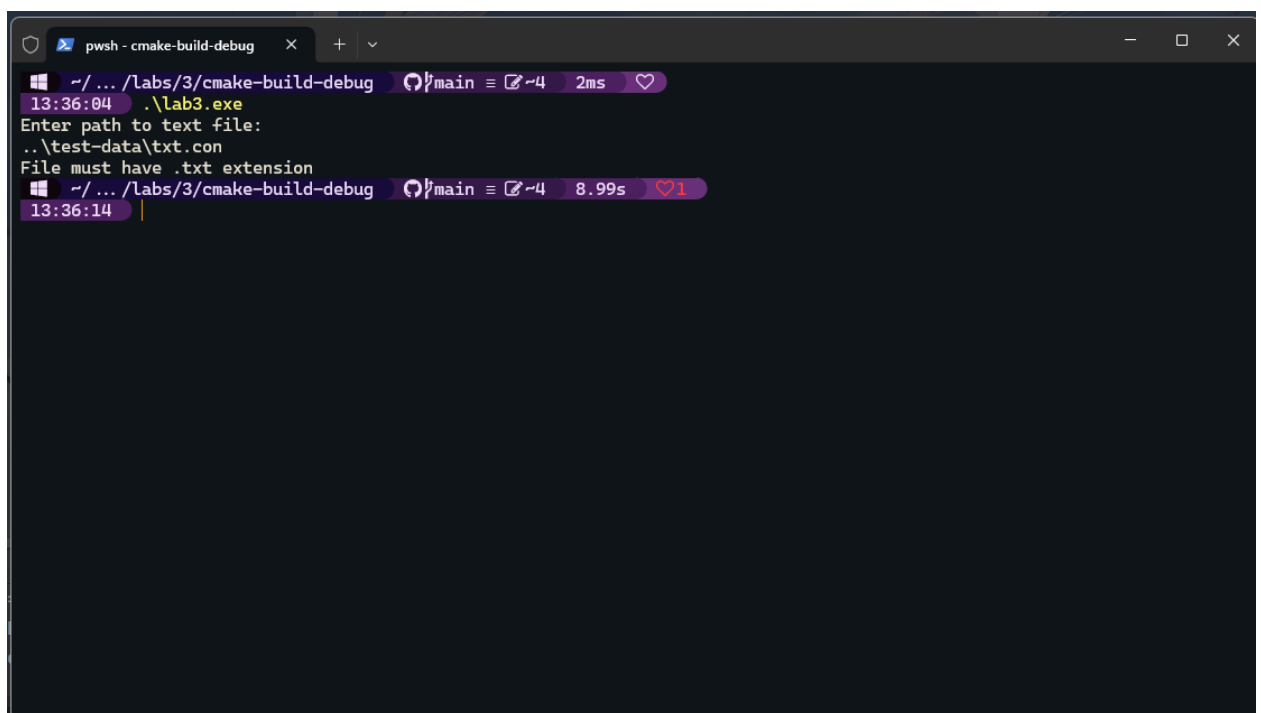
First two words: [Clients, can]
Count of word "Clients": 444
Count of word "can": 444
First words of sentences: [
Using Credit We At Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using Credit For We At
Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using Credit For We At Our Account Clie
ts Clients Using Credit For We At Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using C
redit For We At Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using Credit For We At Ou
r Account Clients Clients Using Credit For We At Our Account Clients Using Credit We At Our Account Clients Clients Usin
g Credit For We At Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using Credit For We At
Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using Credit For We At Our Account Clie
ts Clients Using Credit For We At Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using C
redit For We At Our Account Clients Clients Using Credit For We At Our Account Clients Clients Using Credit For We At Ou
r Account Clients FAILF
]
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> |
```

Рисунок 2 – Правильная работа программы на большом объеме текста



```
Windows PowerShell
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> .\lab3.exe
Enter path to text file:
..\test-data\con~
File is empty
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> |
```

Рисунок 3 – Сообщение об ошибке при попытке открыть пустой файл



```
pwsh - cmake-build-debug
~/.../labs/3/cmake-build-debug main 2ms
13:36:04 .\lab3.exe
Enter path to text file:
..\test-data\txt.con
File must have .txt extension
~/.../labs/3/cmake-build-debug main 8.99s 1
13:36:14 |
```

Рисунок 4 – Сообщение об ошибке при попытке открыть файл с расширением, отличным от .txt

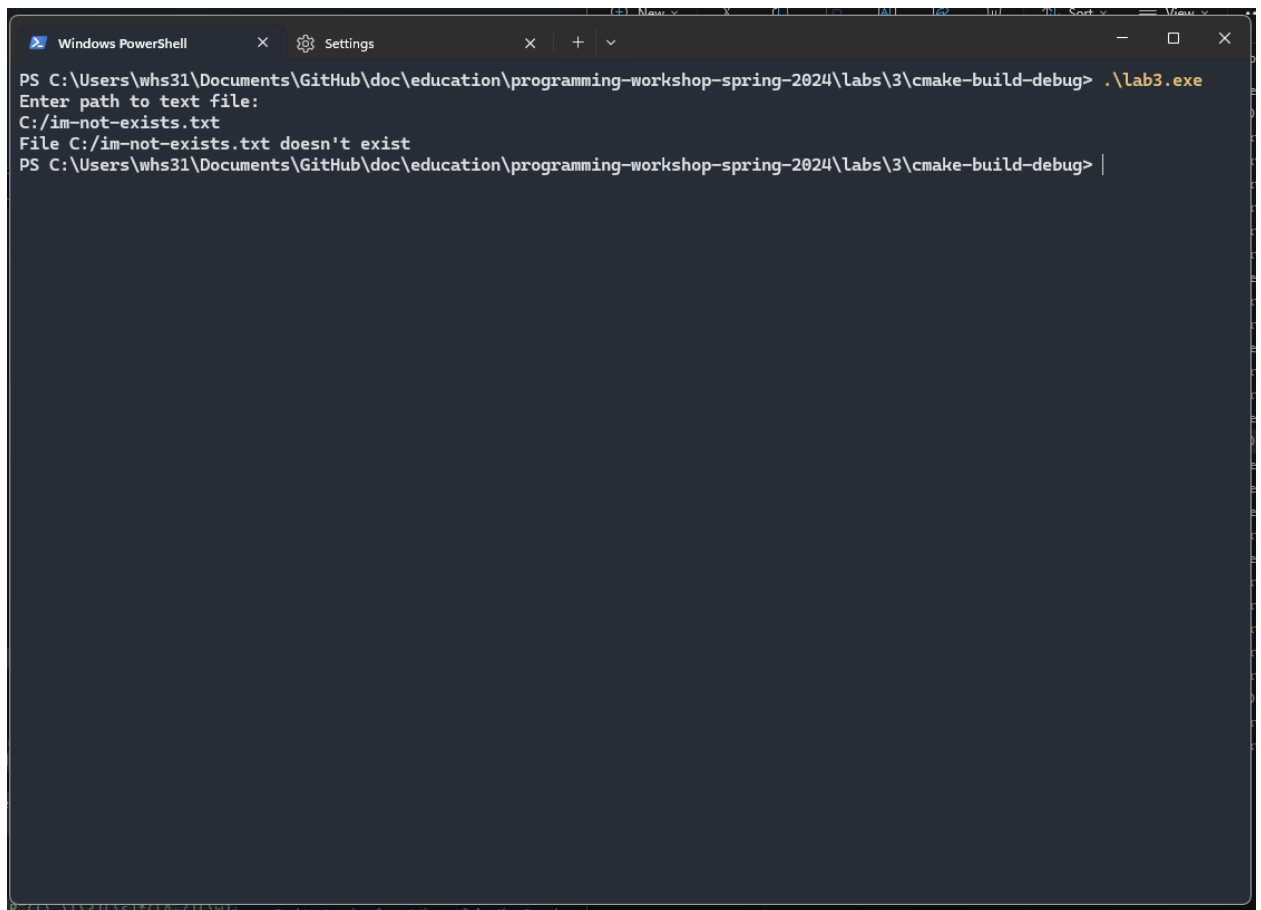
A screenshot of a Windows PowerShell terminal window. The title bar shows 'Windows PowerShell' and 'Settings'. The terminal content shows the following sequence of commands and output:  
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> .\lab3.exe  
Enter path to text file:  
C:/im-not-exists.txt  
File C:/im-not-exists.txt doesn't exist  
PS C:\Users\whs31\Documents\GitHub\doc\education\programming-workshop-spring-2024\labs\3\cmake-build-debug> |  
The terminal has a dark background with light-colored text. The window has standard Windows window controls (minimize, maximize, close) and a settings gear icon.

Рисунок 4 – Сообщение об ошибке при попытке открыть несуществующий файл

## Вывод

В ходе выполнения лабораторной работы я выполнил поставленную задачу по работе с файлами на языке C++. Получены навыки работы с модулем `std::filesystem`, позволяющим работать с файловой системой ОС, файловыми потоками (`std::ifstream`), регулярными выражениями (`std::regex`), базовыми строковыми алгоритмами; навыки форматирования вывода с использованием `std::format`, ввод и вывод в окне терминала с использованием потоковых классов стандартной библиотеки.