МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«ПЕРМСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

ЭЛЕКТРОТЕХНИЧЕСКИЙ ФАКУЛЬТЕТ

КАФЕДРА «ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АВТОМАТИЗИРОВАННЫХ СИСТЕМ»

**ОТЧЁТ**

**«ЛАБОРАТОРНАЯ №6: ОБРАБОТКА СТРОКИ»**

Дисциплина: «Программирование»

Выполнил:

Студент группы ИВТ-21-2б

Безух Владимир Сергеевич

Проверил:

Доцент кафедры ИТАС

Полякова Ольга Андреевна

Пермь, 2022

Содержание

[1. Постановка задачи 3](#_Toc97732349)

[2. Анализ задачи 4](#_Toc97732350)

[3. Описание переменных 5](#_Toc97732351)

[4. Исходный код 6](#_Toc97732352)

[5. Анализ результатов 7](#_Toc97732353)

# Постановка задачи

Обработать строку. Считать до конца строки или до появления символа «точка» все слова, начинающиеся с буквы. Отсортировать полученные слова.

# 

# Анализ задачи

Последовательно считываем символы.

Если символ равен ‘\n’ или ‘.’ — прекращаем считывание.

Если символ — пробел — формируем слово.

Проверяем, если слово только началось, является ли первый символ буквой или числом.

Формируем вектор подходящих слов.

Сортируем найденные слова по алфавиту.

# 

# Описание переменных

**std::string word;** — хранение слова.

**std::vector<std::string> words;** — хранение слов.

**char c;** — хранение символа.

**bool flag\_correct\_word** **= false;** — проверка корректности слова.

# Исходный код

#include <vector>

#include <cctype>

#include <string>

#include <iostream>

#include <algorithm>

int main()

{

std::string word;

std::vector<std::string> words;

char c;

bool flag\_correct\_word = false, flag\_first\_element = false;

while (std::cin.get(c)) {

if (c == '.') break;

if (c == '\n') {

if (word.length() != 0)

words.emplace\_back(std::string(word));

break;

}

if (c == ' ') {

if (word.length() != 0) {

words.emplace\_back(std::string(word));

word.clear();

}

flag\_first\_element = false;

continue;

}

if (flag\_first\_element == false) {

if (isalpha(c)) {

word.push\_back(c);

flag\_correct\_word = true;

}

else

flag\_correct\_word = false;

flag\_first\_element = true;

continue;

}

if (flag\_correct\_word)

word.push\_back(c);

}

for (auto word : words)

std::cout << word << " ";

std::cout << '\n';

sort(words.begin(), words.end());

for (auto word : words)

std::cout << word << " ";

}

# Анализ результатов

Строка считывается посимвольно и обрабатывается через флаги и эвристики. Набор подходящих слов лексикографически сортируется (рис. 1).

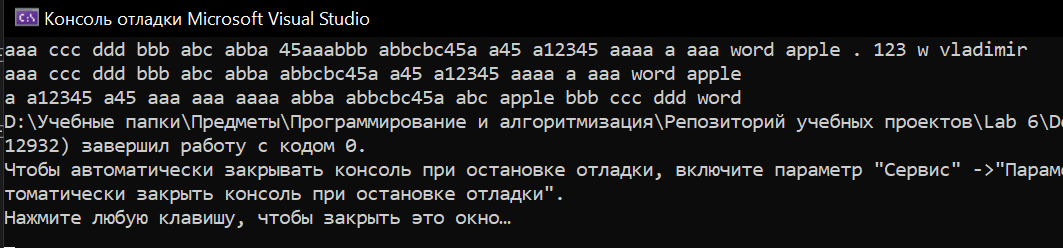


Рисунок — Результаты выполнения программы