**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра информационных систем**

отчет

**по практической работе №4**

**по дисциплине «Программирование»**

Тема: Текстовые строки как массивы символов

|  |  |  |
| --- | --- | --- |
| Студент гр. 2373 |  | Герасимов А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2023

**Цель работы.**

Изучение представления текстовых строк в программах. Работа со строками как с массивами символов. Операции над строками. Знакомство с классом **std::string**.

**Основные теоретические положения.**

Текстовые строки представляются с помощью одномерных массивов символов. В языке **C++** текстовая строка представляет собой набор символов, обязательно заканчивающийся нулевым символом (‘\0’). Поэтому, если вы хотите создать текстовый массив для хранения 10 (N) символов, нужно выделить память под 11 (N+1) символов.

Объявленный таким образом массив может использоваться для хранения текстовых строк, содержащих не более 10 символов. Нулевой символ позволяет определить границу между содержащимся в строке текстом и неиспользованной частью строки.

Класс string предназначен для работы со строками типа char, которые представляют собой строчку с завершающим нулем (символ ‘\0’). Класс string был введен как альтернативный вариант для работы со строками типа char.

Чтобы использовать возможности класса **string**, нужно подключить библиотеку <string> и пространство имен **std**. Объявление же переменной типа **string** осуществляется схоже с обычной переменной:

**Постановка задачи.**

Необходимо написать программу, которая:

1) С клавиатуры или с файла (пользователь сам может выбрать способ ввода) вводится последовательность, содержащая от 1 до 50 слов, в каждом из которых от 1 до 10 строчных латинских букв и цифр. Между соседними словами произвольное количество пробелов. За последним символом стоит точка.

2) Необходимо отредактировать входной текст:

- удалить лишние пробелы;

- удалить лишние знаки препинания (под «лишними» подразумевается несколько подряд идущих знаков (обратите внимание, что «…» - корректное использование знака) в тексте);

- исправить регистр букв, если это требуется (пример некорректного использования регистра букв: пРиМЕр);

3) Выполнить задание по варианту: вывести на экран слова последовательности в обратном порядке.

4) Выполнить задание по варианту: вывести на экран ту же последовательность, заменив во всех словах первую букву соответствующей прописной буквой.

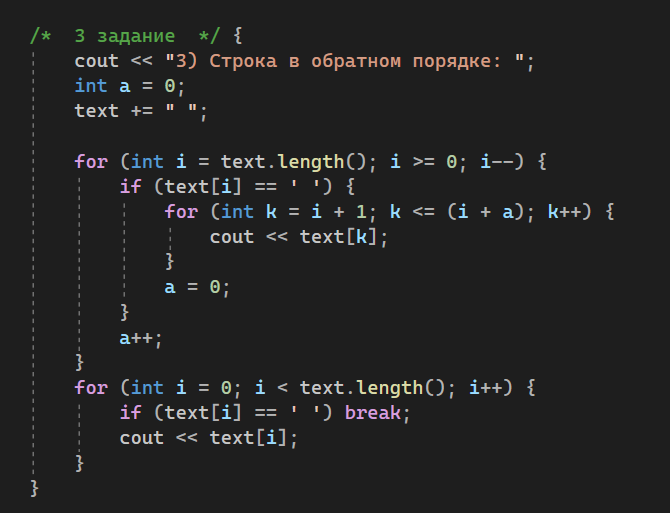
5) Необходимо найти все подстроки, которую введёт пользователь в имеющейся строке. Реализуйте два алгоритма: первый алгоритма – Линейный поиск, а второй алгоритм согласно вашему номеру в списке. Четные номера должны реализовать алгоритм КНМ, а нечетные – Бойера-Мура.

**Выполнение работы.**

Код программы представлен в приложении А.

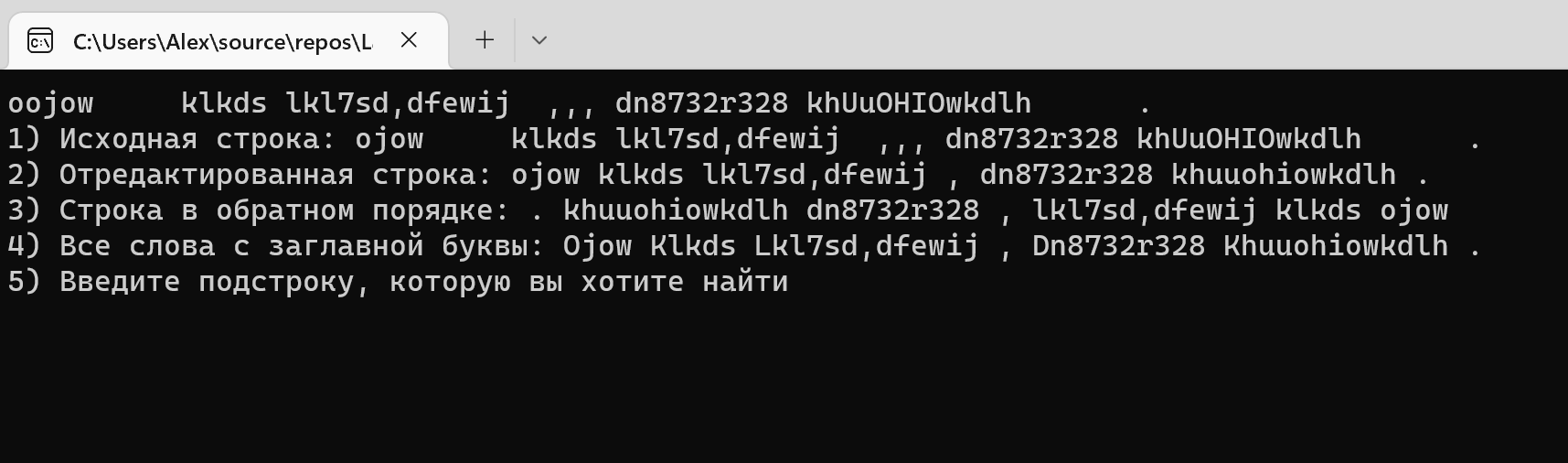
**Блок описания кода и использованных алгоритмов:**

1. Пользователь вводит строку или считывает её с файла.
2. Эта строка редактируется с помощью прописанных функций: 1 - удаляет лишние пробелы, 2 - удаляет лишние знаки препинания. 3 – исправляет регистры букв (если первая буква заглавная, то считается, что она и должна такой быть).
3. Слова выводятся в обратном порядке согласно программе:



1. В уже отредактированной строке во всех словах первая буква заменяется на заглавную. Делается это с помощью соответствующей функции.
2. Линейным поиском находится подстрока введённая пользователем.

**Блок скриншотов работы программы**



**Вывод.**

Во время выполнения данной работы были изучены строки, массивы символов и принципы работы с ними. Также был получен навык работы со строками.

Приложение А

рабочий код

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

string text;

void DeleteSpace(string txt) {

for (int i = 0; i < txt.size(); i++) {

if (txt[i] == ' ' && txt[i + 1] == ' ')

while (txt[i + 1] == ' ')

txt.erase((i + 1), 1);

}

text = txt;

}

void DeleteSign(string txt) {

for (int i = 0; i <= txt.size(); i++) {

if ((txt[i] == '.' || txt[i] == ',' || txt[i] == ';' || txt[i] == ':' || txt[i] == '!' || txt[i] == '?') && (txt[i + 1] == '.' || txt[i + 1] == ',' || txt[i + 1] == ';' || txt[i + 1] == ':' || txt[i + 1] == '!' || txt[i + 1] == '?'))

if (txt[i] == '.' && txt[i + 1] == '.' && txt[i + 2] == '.' && txt[i + 3] == ' ')

i += 3;

else {

while (txt[i + 1] == '.' || txt[i + 1] == ',' || txt[i + 1] == ';' || txt[i + 1] == ':' || txt[i + 1] == '!' || txt[i + 1] == '?')

txt.erase((i + 1), 1);

}

}

text = txt;

}

void Register(string txt) {

for (int i = 0; i < txt.size(); i++) {

if (txt[i] == ' ')

i += 2;

txt[i] = tolower(txt[i]);

}

text = txt;

}

void UpFirst(string txt) {

txt[0] = toupper(txt[0]);

for (int i = 0; i < txt.size(); i++) {

if (txt[i] == ' ')

txt[i + 1] = toupper(txt[i + 1]);

}

text = txt;

}

void Search(string txt) {

string key;

bool check;

int count = 0;

cin >> key;

for (int i = 0; i < txt.size(); i++) {

if (txt[i] == key[0]) {

check = true;

for (int j = 1; j < key.size(); j++) {

if (txt[i + j] != key[j]) {

check = false;

}

}

if (check == true) {

count++;

cout << "Позиция " << i << "\n";

}

}

}

if (count == 0) {

cout << "По запросу ничего не найдено";

}

else {

cout << "Всего найдено " << count << " соответствий";

}

}

int main() {

setlocale(0, "");

/\* 1 задание \*/ {

cin.ignore();

getline(cin, text);

cout << "1) Исходная строка: " << text << "\n";

/\* 2 задание \*/ {

DeleteSpace(text);

DeleteSign(text);

Register(text);

cout << "2) Отредактированная строка: " << text << "\n";

}

/\* 3 задание \*/ {

cout << "3) Cтрока в обратном порядке: ";

int a = 0;

text += " ";

for (int i = text.length(); i >= 0; i--) {

if (text[i] == ' ') {

for (int k = i + 1; k <= (i + a); k++) {

cout << text[k];

}

a = 0;

}

a++;

}

for (int i = 0; i < text.length(); i++) {

if (text[i] == ' ') break;

cout << text[i];

}

}

/\* 4 задание \*/ {

UpFirst(text);

cout << "\n4) Все слова с заглавной буквы: " << text << "\n";

}

/\* 5 задание \*/ {

cout << "5) Введите подстроку, которую вы хотите найти \n";

Search(text);

}

cout << "\n";

}

}