**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

|  |  |  |
| --- | --- | --- |
| Студентка гр. 2373 |  | Маркова З. А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**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.

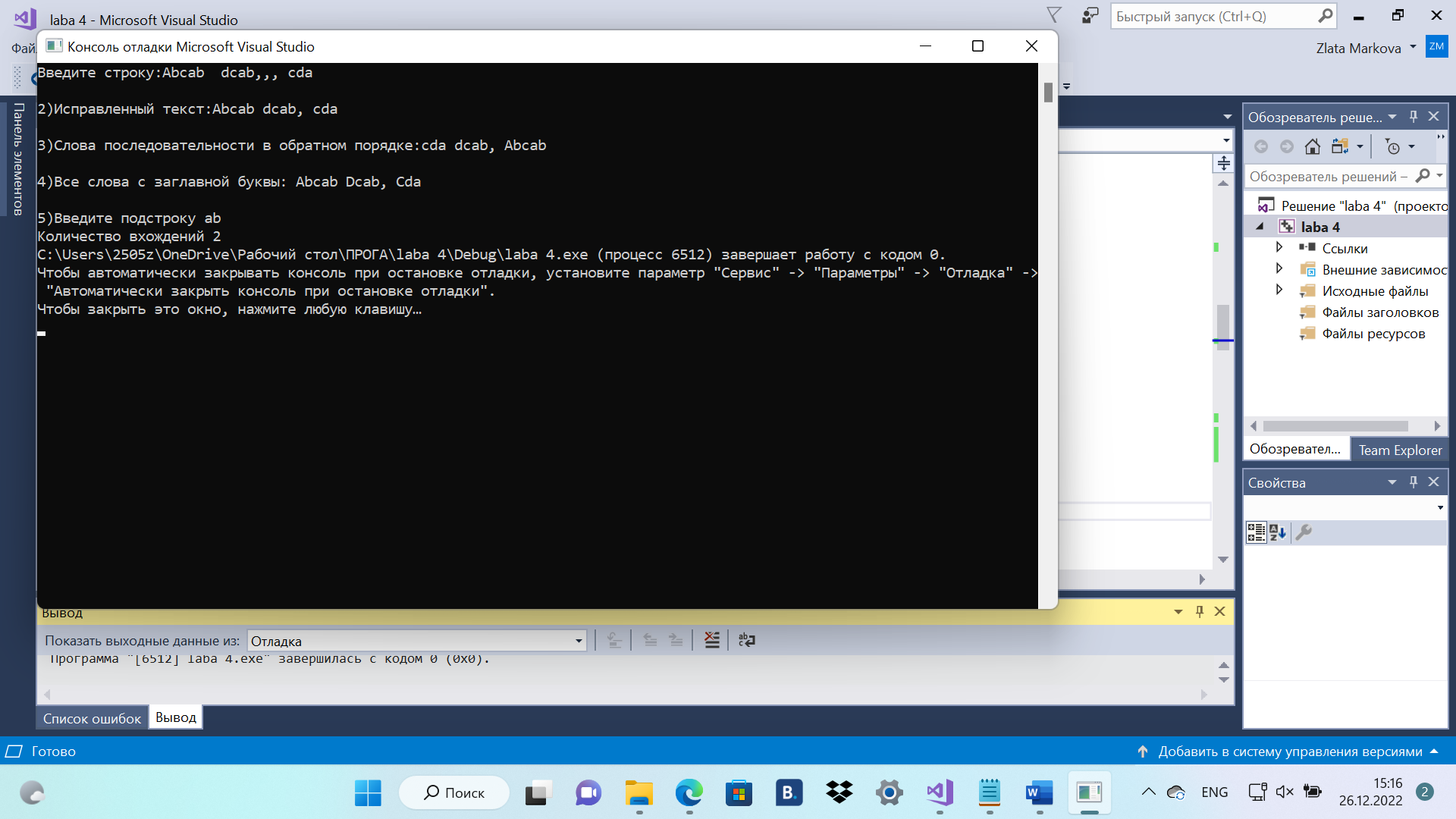
**Выполнение работы.**

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
4. В уже отредактированной строке во всех словах первая буква заменяется на заглавную. Делается это с помощью соответствующей функции.
5. 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>

using namespace std;

#include <fstream>

#include <string>

int main()

{

setlocale(LC\_ALL, "ru");

/\*1)

ifstream File;

File.open("Name.txt");

string S;

while (!File.eof())

{

getline(File, S);

cout << S;

}

File.close();\*/

//2)

string S;

cout << "Введите строку:";

getline(cin, S);

int k = 0;

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

{

if (S[i] == ' ')

{

for (int j = 1; j < S.length() - 1; j++)

{

if (S[i + j] == ' ' )

k++;

if (S[i + j] != ' ')

break;

}

S.erase(i + 1, k);

}

k = 0;

}

//cout << S<<endl;

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

{

if (!(isalnum(S[i])) and S[i] != ' ')

{

for (int j = 1; j < S.length() - 1; j++)

{

if (!(isalnum(S[i + j])) and S[i + j] != ' ')

{

k++;

}

if (isalnum(S[i + j]) or S[i+j]==' ') {

break;

}

}

S.erase(i + 1, k);

}

k = 0;

}

//cout << S;

for (int i = 1; i < S.length(); i++)

{

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

}

cout <<"\n2)Исправленный текст:"<< S << "\n\n";

//3)

string S1;

S1 = S;

cout << "3)Слова последовательности в обратном порядке:";

int l;

while (S1.length() > S1.find(' '))

{

l = S1.rfind(' ');

for (int j = l + 1; j < S1.length(); j++)

{

cout << S1[j];

}

cout << " ";

S1 = S1.erase(l);

}

cout << S1<<"\n\n";

//4)

cout << "4)Все слова с заглавной буквы: ";

string S2;

S2 = S;

int n;

S2[0] = toupper(S2[0]);

for (int i = 0; i < S2.length(); i++)

{

if (S2[i] ==' ')

S2[i+1] = toupper(S2[i+1]);

cout << S2[i];

}

cout << "\n\n";

//5)

string str;

cout << "5)Введите подстроку ";

getline(cin, str);

int count = 0;

int c = 0;

for (int i = 0; i < (S.length() - (str.length() - 1)); i++)

{

if (S[i] == str[0])

{

count++;

for (int j = 1; j < str.length(); j++)

{

if (S[i + j] == str[j])

{

count++;

}

if (S[i + j] != str[j])

break;

}

}

if (count == str.length())

c++;

count = 0;

}

if (c == 0)

cout << "Подстрока не найдена";

else

cout << "Количество вхождений " << c;

}