**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.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 2372 |  | Коваленко М.А. |
| Преподаватель |  | Глущенко А. Г. |

Санкт-Петербург

2022

**Цель работы.**

Научиться работать с классов string, выводить и вводить текст с клавиатуры, выполнять различные операции с текстом и находить подстроку в строке.

**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.

Класс string обладает широким функционалом:

* функция compare() сравнивает одну часть строки с другой;
* функция length() определяет длину строки;
* функции find() и rfind() служат для поиска подстроки в строке (отличаются функции лишь направлением поиска);
* функция erase() служит для удаления символов;
* функция replace() выполняет замену символов;
* функция insert() необходима, чтобы вставить одну строку в заданную позицию другой строки;

Но весь функционал string накладывает и свой негативный отпечаток. Основным недостатком string в сравнении с типом char является замедленная скорость обработки данных.

**Постановка задачи.**

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);

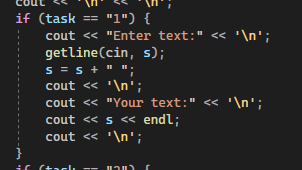
3) Вывести на экран только те слова последовательности, в которых встречаются одинаковые буквы.

4) Вывести на экран ту же последовательность, заменив во всех словах первую букву соответствующей прописной буквой.

5) Найти подстроку в строке через линейный поиск.

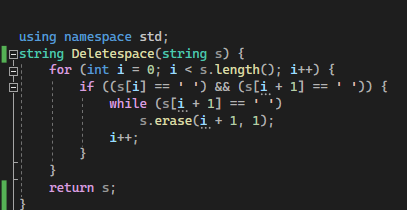
**Выполнение работы.**

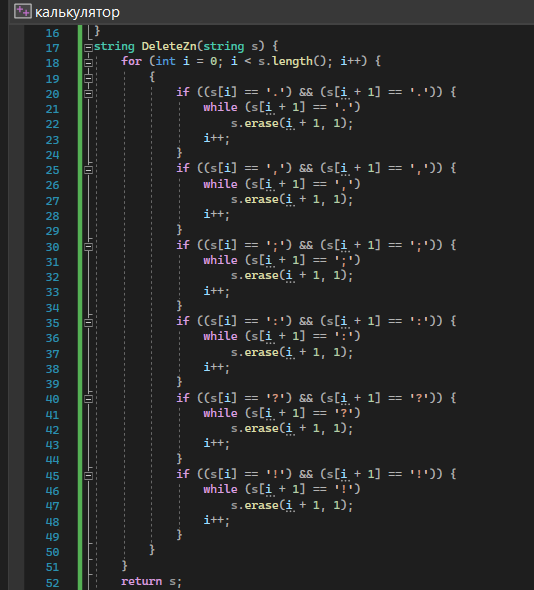
Вводим текст с клавиатуры

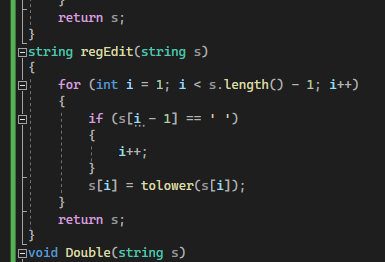


Далее мы должны отредактировать наш текст, то есть исправить все ошибки

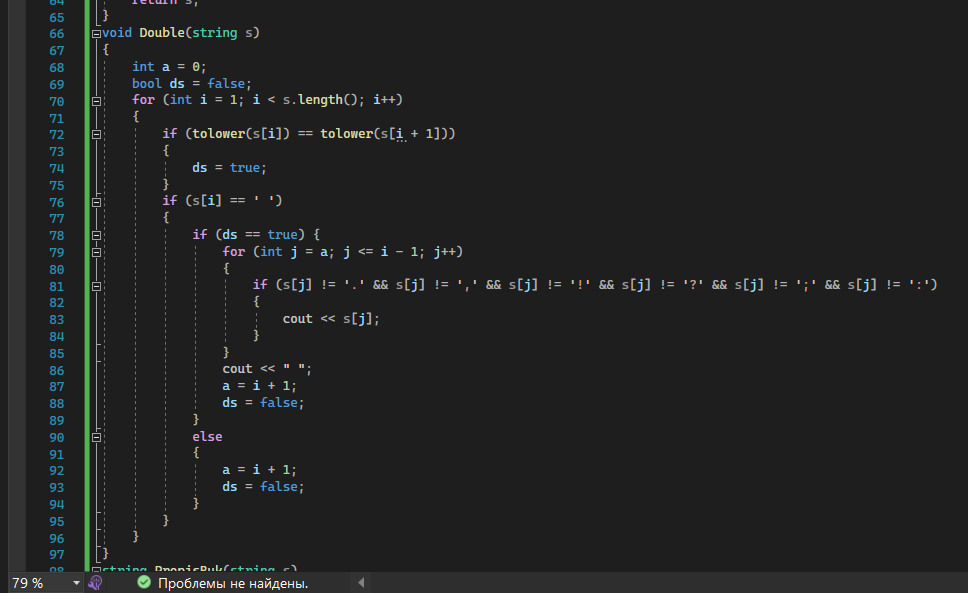
Выполняем это с помощью следующих функций:

 Функция удаления лишних пробелов

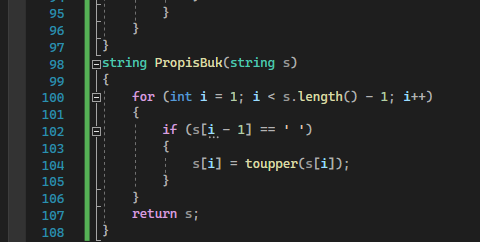
 Функция удаления лишних знаков препинания

 Функция исправления регистров букв

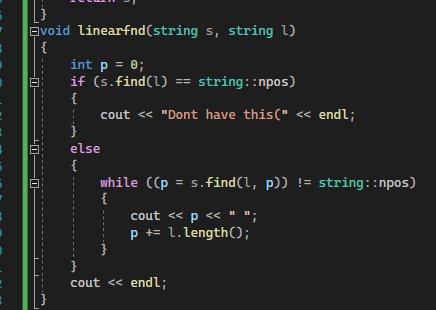
Далее мы выводим на экран те слова, в которых идут подряд несколько одинаковых букв



Дальше мы должны вывести текст, в котором все слова будут начинаться с большой буквы



Дальше мы находим подстроку в строке с помощью линейного поиска



**Выводы:**

Я научился выводить и вводить текстовые строки класс string, научился редактировать исходный текст, выполнять различные задания с текстом и находить подстроку в строке.

Приложение А

Полный код программы

#include <iostream>

#include <string>

#include <fstream>

using namespace std;

string Deletespace(string s) {

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

if ((s[i] == ' ') && (s[i + 1] == ' ')) {

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

s.erase(i + 1, 1);

i++;

}

}

return s;

}

void linearfnd(string s, string l)

{

int p = 0;

if (s.find(l) == string::npos)

{

cout << "Dont have this(" << endl;

}

else

{

while ((p = s.find(l, p)) != string::npos)

{

cout << p << " ";

p += l.length();

}

}

cout << endl;

}

string DeleteZn(string s) {

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

{

if ((s[i] == '.') && (s[i + 1] == '.')) {

while (s[i + 1] == '.')

s.erase(i + 1, 1);

i++;

}

if ((s[i] == ',') && (s[i + 1] == ',')) {

while (s[i + 1] == ',')

s.erase(i + 1, 1);

i++;

}

if ((s[i] == ';') && (s[i + 1] == ';')) {

while (s[i + 1] == ';')

s.erase(i + 1, 1);

i++;

}

if ((s[i] == ':') && (s[i + 1] == ':')) {

while (s[i + 1] == ':')

s.erase(i + 1, 1);

i++;

}

if ((s[i] == '?') && (s[i + 1] == '?')) {

while (s[i + 1] == '?')

s.erase(i + 1, 1);

i++;

}

if ((s[i] == '!') && (s[i + 1] == '!')) {

while (s[i + 1] == '!')

s.erase(i + 1, 1);

i++;

}

}

}

return s;

}

string regEdit(string s)

{

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

{

if (s[i - 1] == ' ')

{

i++;

}

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

}

return s;

}

void Double(string s)

{

int a = 0;

bool ds = false;

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

{

if (tolower(s[i]) == tolower(s[i + 1]))

{

ds = true;

}

if (s[i] == ' ')

{

if (ds == true) {

for (int j = a; j <= i - 1; j++)

{

if (s[j] != '.' && s[j] != ',' && s[j] != '!' && s[j] != '?' && s[j] != ';' && s[j] != ':')

{

cout << s[j];

}

}

cout << " ";

a = i + 1;

ds = false;

}

else

{

a = i + 1;

ds = false;

}

}

}

}

string PropisBuk(string s)

{

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

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

{

if (s[i - 1] == ' ')

{

s[i] = toupper(s[i]);

}

}

return s;

}

int main() {

string s;

while (true) {

string task;

cout << "Select task:\n"

"1. Enter text\n"

"2. Fixed text\n"

"3. Identical letters\n"

"4. Uppercase letter\n"

"5. Substrings\n";

getline(cin, task);

cout << '\n' << '\n';

if (task == "1") {

string task2;

cout << "Select way:\n"

"1. Keyboard\n"

"2. File\n";

getline(cin, task2);

if (task2 == "1") {

cout << "Enter text:" << '\n';

getline(cin, s);

s = s + " ";

cout << '\n';

cout << "Your text:" << '\n';

cout << s << endl;

cout << '\n';

}

if (task2 == "2") {

ifstream file("C://123.txt");

char a;

while (!file.eof())

{

file.get(a);

s.push\_back(a);

}

cout << "Your text:" << '\n';

cout << s << '\n';

}

}

if (task == "2") {

s = DeleteZn(s);

s = Deletespace(s);

s = regEdit(s);

s = DeleteZn(s);

cout << "Corect text:" << '\n';

cout << s << endl;

cout << '\n';

}

if (task == "3") {

cout << "Words with the same letters: ";

Double(s);

cout << endl;

}

if (task == "4") {

cout << "Text, in which all the words with a capital letter:" << '\n';

s = PropisBuk(s);

cout << s << endl;

}

if (task == "5") {

string l;

cout << "Enter word: ";

cin >> l;

linearfnd(s, l);

}

}

}