**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра вычислительной техники**

отчет

**по лабораторной работе №1**

**по дисциплине «Программирование»**

Тема: Строка и подстрока

|  |  |  |
| --- | --- | --- |
| Студент гр. 4373 |  | Матвеев А.В. |
| Преподаватель |  | Глущенко |

Санкт-Петербург

2024

**Цель работы.**

Знакомство со строками и подстроками.

#include <iostream>

#include <string>

#include <cctype>

using namespace std;

const int MAX\_LENGTH = 256; // Максимальная длина строки

void removeExtraSpaces(char\* str) {

char temp[MAX\_LENGTH];

int j = 0;

bool inSpace = false;

for (int i = 0; str[i]; i++) {

if (isspace(str[i])) {

if (!inSpace) {

temp[j++] = ' '; // Сохраняем один пробел

inSpace = true;

}

}

else {

temp[j++] = str[i];

inSpace = false;

}

}

temp[j] = '\0';

for (int i = 0; i < MAX\_LENGTH; i++) {

str[i] = temp[i];

}

}

void removeExtraPunctuation(char\* str) {

char temp[MAX\_LENGTH];

int j = 0;

bool inPunct = false;

for (int i = 0; str[i]; i++) {

if (ispunct(str[i])) {

if (!inPunct) {

temp[j++] = str[i];

inPunct = true;

}

}

else {

temp[j++] = str[i];

inPunct = false;

}

}

temp[j] = '\0';

for (int i = 0; i < MAX\_LENGTH; i++) {

str[i] = temp[i];

}

}

void correctCase(char\* str) {

bool firstChar = true;

for (int i = 0; str[i]; i++) {

if (isalpha(str[i])) {

if (firstChar) {

str[i] = toupper(str[i]); // Первая буква заглавная

firstChar = false;

}

else {

str[i] = tolower(str[i]); // Остальные буквы строчные

}

}

/\*else {

firstChar = true; // Если не буква, следующий символ может быть заглавным

}\*/

}

}

bool hasDuplicateLetters(const char\* word) {

int letterCount[26] = { 0 }; // Для букв от 'a' до 'z'

for (int i = 0; word[i]; i++) {

if (isalpha(word[i])) {

int index = tolower(word[i]) - 'a';

letterCount[index]++;

if (letterCount[index] > 1) {

return true; // Если буква встречается больше одного раза

}

}

}

return false;

}

void printWordsWithDuplicates(const char\* sequence) {

char word[11]; // Максимальная длина слова 10 + 1 для нуль-терминатора

int index = 0;

for (int i = 0; sequence[i]; i++) {

if (isspace(sequence[i]) || sequence[i] == '.') {

if (index > 0) {

word[index] = '\0'; // Завершаем слово

if (hasDuplicateLetters(word)) {

cout << word << endl; // Выводим слово с одинаковыми буквами

}

index = 0; // Сбрасываем индекс для следующего слова

}

}

else {

word[index++] = sequence[i];

}

}

// Проверка последнего слова, если оно не заканчивается пробелом

if (index > 0) {

word[index] = '\0';

if (hasDuplicateLetters(word)) {

cout << word << endl;

}

}

}

void removeCharsAndDigits(const char\* sequence, const char\* charsToRemove) {

for (int i = 0; sequence[i]; i++) {

// Проверяем, является ли текущий символ частью charsToRemove

bool toRemove = false;

for (int j = 0; charsToRemove[j]; j++) {

if (sequence[i] == charsToRemove[j]) {

toRemove = true;

break;

}

}

// Если символ не нужно удалять, выводим его

if (!toRemove) {

cout << sequence[i];

}

// Если текущий символ - пробел, выводим его

if (isspace(sequence[i])) {

cout << ' ';

}

}

cout << endl; // Переход на новую строку после завершения

}

void findSubstrings(const char\* mainString, const char\* substring) {

int mainLength = strlen(mainString);

int subLength = strlen(substring);

// Перебираем все возможные позиции в основной строке

for (int i = 0; i <= mainLength - subLength; i++) {

// Сравниваем подстроку с текущей частью основной строки

bool found = true;

for (int j = 0; j < subLength; j++) {

if (mainString[i + j] != substring[j]) {

found = false;

break;

}

}

// Если подстрока найдена, выводим позицию

if (found) {

std::cout << "Найдена подстрока \"" << substring << "\" на позиции: " << i << std::endl;

}

}

}

int main() {

setlocale(LC\_ALL, "Russian");

char input[MAX\_LENGTH];

char charsToRemove[100];

char substring[MAX\_LENGTH];

cout << "Введите последовательность (до 50 слов, заканчивается точкой): ";

cin.getline(input, MAX\_LENGTH);

// Удаление точки в конце

if (input[strlen(input) - 1] == '.') {

input[strlen(input) - 1] = '\0';

}

// Удаление лишних пробелов и знаков препинания

removeExtraSpaces(input);

removeExtraPunctuation(input);

correctCase(input);

// Вывод отредактированной строки

std::cout << "Отредактированный текст: " << input << std::endl;

cout << "Слова с одинаковыми буквами: " << endl;

printWordsWithDuplicates(input);

cout << "Введите набор букв и цифр для удаления: ";

cin.getline(charsToRemove, sizeof(charsToRemove));

removeCharsAndDigits(input, charsToRemove);

cout << "Введите подстроку для поиска: ";

cin.getline(substring, MAX\_LENGTH);

findSubstrings(input, substring);

return 0;

}

