**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. 4373 | Репков А.А |  |
| Преподаватель | Глущенко А. Г. |  |

Санкт-Петербург

2024

**Цель работы.**

Изучение работы строк и их внутреннего представления в памяти; сортировки; получение практических навыков работы со строками.

**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.

При определении строковых переменных их можно инициализировать конкретными значениями с помощью строковых литералов:

char S1[15] = “This is text”;

char S2[] = “Пример текста”;

Последние два элемента переменной  просто не используются, а строка  автоматически подстраивается под длину инициализирующего текста.

При работе со строками можно обращаться к отдельным символам строки как в обычном одномерном массиве с помощью индексов:

cout << S1[0]; // На экране будет выведен символ ‘T’

Если строка формируется при помощи цикла (или иного способа), то необходимо в ее конец обязательно записать нулевой символ '\0'.

**Постановка задачи.**

Необходимо написать программу, которая реализует поставленную задачу:

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:

|  |  |
| --- | --- |
| 1 | Вывести на экран слова последовательности в обратном порядке. |
| 2 | Вывести на экран слова последовательности в алфавитном порядке. |
| 3 | Вывести на экран слова последовательности, не содержащие цифр. |
| 4 | Вывести на экран только те слова последовательности, в которых встречаются одинаковые буквы. |
| 5 | Вывести на экран только те слова последовательности, в которых первая буква слова встречается в этом слове еще раз. |
| 6 | После окончания ввода последовательности вывести на экран сначала все слова, содержащие только буквы, затем слова, содержащие только цифры, а потом слова, содержащие и буквы, и цифры. |

4) Выполнить задание по варианту:

|  |  |
| --- | --- |
| 1 | Вывести на экран ту же последовательность, заменив во всех словах первую букву соответствующей прописной буквой. |
| 2 | Вывести на экран количество символов в каждом слове исходной последовательности. |
| 3 | Вывести на экран ту же последовательность, удалив из всех слов заданный набор букв и (или) цифр. |
| 4 | Вывести на экран ту же последовательность, заменив во всех словах цифры на буквы латинского алфавита, номера которых в алфавите равны заменяемой цифре. |
| 5 | Вывести на экран ту же последовательность, переместив все цифры, содержащиеся в словах, в конец соответствующих слов. |
| 6 | Вывести все слова исходной последовательности на экран вертикально. |
| 7 | Вывести на экран  все слова последовательности в две или три колонки (в зависимости от количества слов) с выравниванием слов по правой границе колонки. |

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 КМП, а нечетные – Бойера-Мура. (\*)

**Выполнение работы**

#include <iostream>

#include <sstream>

#include <string>

#include <vector>

#include <cctype>

#include <algorithm>

using namespace std;

string cleanText(const string& input) {

string result;

bool inSpace = true;

char lastChar = '0';

for (char c : input) {

if (isspace(c)) {

if (!inSpace) {

result += ' ';

inSpace = true;

}

}

else {

if (ispunct(c)) {

if (lastChar != c) {

result += c;

lastChar = c;

}

}

else {

result += c;

lastChar = c;

inSpace = false;

}

}

}

return result;

}

string fixCase(const string& text) {

string result;

stringstream stream(text);

string word;

while (stream >> word) {

if (!word.empty()) {

// Приводим первую букву к верхнему регистру, остальные — к нижнему

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

for (size\_t i = 1; i < word.length(); ++i) {

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

}

result += word + ' ';

}

}

if (!result.empty()) {

result.pop\_back();

}

return result;

}

bool containsDigit(const string& word) {

return any\_of(word.begin(), word.end(), isdigit);

}

void printWordsWithoutDigits(const string& text) {

setlocale(LC\_ALL, "RU");

istringstream stream(text);

string word;

cout << "\n";

cout << "Слова без цифр:\n";

while (stream >> word) {

if (!containsDigit(word)) {

std::cout << word << "\n";

}

}

}

void printWordsWithCapitalFirstLetter(const string& text) {

setlocale(LC\_ALL, "RU");

istringstream stream(text);

string word;

cout << "\n";

cout << "Слова с заглавной первой буквой:\n";

while (stream >> word) {

if (!word.empty()) {

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

cout << word << "\n";

}

}

}

void findSubstring(const string& text, const string& substring) {

size\_t pos = text.find(substring);

if (pos == string::npos) {

cout << "Подстрока не найдена.\n";

}

else {

cout << "Подстрока найдена на позиции: " << pos << "\n";

while ((pos = text.find(substring, pos + 1)) != string::npos) {

cout << "Подстрока найдена на позиции: " << pos << "\n";

}

}

}

int main() {

string input;

setlocale(LC\_ALL, "RU");

// Задание 1

cout << "Введите последовательность слов (до 50 слов, каждое от 1 до 10 символов): ";

getline(cin, input);

if (input.back() == '.') {

input.pop\_back();

}

// Задание 2

string cleanedText = cleanText(input);

//fix registr

string fixedText = fixCase(cleanedText);

// Задание 3

printWordsWithoutDigits(fixedText);

// Задание 4

printWordsWithCapitalFirstLetter(cleanedText);

// Задание 5

string substring;

cout << "\n";

cout << "Введите подстроку для поиска: ";

getline(std::cin >> ws, substring); // Убираем пробелы перед вводом

findSubstring(cleanedText, substring);

return 0;

}

**Выводы.**

Я изучил строки и понял, как они представлены в памяти компьютера, научился работать с различными строками.