ДНІПРОВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ОЛЕСЯ ГОНЧАРА

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

КАФЕДРА КОМП’ЮТЕРНИХ ТЕХНОЛОГІЙ

**Лабораторна робота №9**

«Обробка текстової інформації»

**з курсу «Програмування»**

**Варіант №14**

Виконав:

студент групи ПА-22-2

Овдієнко Андрій Володимирович

Дніпро, 2023

**Зміст**

[**Зміст** 2](#_Toc134615058)

[**1.Завдання до лабораторної роботи №9** 3](#_Toc134615059)

[**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”** 3](#_Toc134615060)

[**Завдання 2 “Рядкові функції”** 3](#_Toc134615061)

[**Завдання 3 “Обробка текстової інформації, що міститься у файлах”** 3](#_Toc134615062)

[**2.Опис розв’язку** 5](#_Toc134615063)

[**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”** 5](#_Toc134615064)

[**Завдання 2 “Рядкові функції”** 6](#_Toc134615065)

[**Функція “memmove”** 6](#_Toc134615066)

[**Функція “strspn( )”** 7](#_Toc134615067)

[**Завдання 3 “Обробка текстової інформації, що міститься у файлах”** 8](#_Toc134615068)

[**3.Опис інтерфейсу програми (керівництво користувача)** 10](#_Toc134615069)

[**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”** 10](#_Toc134615070)

[**Завдання 2 “Рядкові функції”** 11](#_Toc134615071)

[**Завдання 3 “Обробка текстової інформації, що міститься у файлах”** 14](#_Toc134615072)

[**Додаток (коди програм)** 16](#_Toc134615073)

[**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”** 16](#_Toc134615074)

[**Завдання 2 “Рядкові функції”** 23](#_Toc134615075)

[**Завдання 3 “Обробка текстової інформації, що міститься у файлах”** 34](#_Toc134615076)

**1.Завдання до лабораторної роботи №9**

**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”**

Виконати задані дії із введенним текстом.

Індивідуальне завдання варіанта 14:

“Вивести на екран слова з речення, які вміщують в собі символи, які повторюються, причому під словом вивести на екран символ, який повторюється.”

**Завдання 2 “Рядкові функції”**

Написати програму, в якій були б розроблені 2 функції, що імітують дії стандартних рядкових функцій (див. індивідуальне завдання).

Вибір функцій здійснюється за допомогою пунктів меню.

Індивідуальне завдання варіанта 14:

“memmove( ), strspn( )”.

**Завдання 3 “Обробка текстової інформації, що міститься у файлах”**

За допомогою текстового редактора створити файл, що містить текст. Довжина рядка тексту не повинна перевищувати 80 символів. Це вхідний файл.

Програма повинна

інформацію вхідного файлу записати у вихідний файл;

реалізувати дії, вказані в індивідуальному завданні і занести відповідну інформацію у вихідний файл.

Ім'я вхідного файлу задає користувач в командному рядку або під час виконання програми.

Ім'я вихідного файлу: перші символи - не більше трьох символів з імені вхідного файлу, а решти

символів \_out.

Вихідний файл повинен мати розширення .dat.

Програма повинна забезпечувати по запиту користувача: 1) виведення тексту вхідного файлу на екран дисплея; 2) виведення тексту вихідного файлу на екран дисплея.

Закінчені послідовності дій оформити у вигляді функцій. Всі необхідні дані для функцій повинні передаватися їм як параметри. Використання глобальних змінних у функціях не допускається.

Для вводу і виводу інформації (і для консолі, і для текстових файлів) застосувати засоби форматованого вводу/виводу.

Визначення функцій і головну функцію розмістити в двох окремих файлах.

Індивідуальне завдання варіанта 14:

Визначити кількість слів, що містять заданий користувачем символ. Вивести ці слова.

**2.Опис розв’язку**

**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”**

Даний алгоритм призначений для пошуку слів у введеному тексті, які містять повторювані букви.

На початку програми встановлюється константа MAX\_TEXT\_LENGTH, яка визначає максимальну довжину тексту, та створюється масив char з назвою text довжиною MAX\_TEXT\_LENGTH.

Після цього виконується команда cin.getline(text, MAX\_TEXT\_LENGTH), яка дозволяє ввести текст з клавіатури та зберегти його у масиві text.

Cтворюється масив символів з назвою punctuation, який містить усі можливі розділові знаки, які ми будемо шукати в тексті.

Використовуючи формулу sizeof(punctuation) / sizeof(punctuation[0]) визначаємо довжину масиву punctuation та зберігаємо її в змінну len.

Створюється допоміжний цикл, який перевіряє кожен символ у введеному тексті. Цикл продовжується до тих пір, поки не зустріне символ '\0', який позначає кінець тексту.

У середині цього циклу дані символи порівнюються з кожним елементом масиву punctuation за допомогою другого циклу. Якщо символ збігається з елементом масиву punctuation, то він замінюється на пробіл, і цикл завершується за допомогою команди break.

Отже, на кожній ітерації допоміжного циклу перевіряється, чи є поточний символ розділовим знаком. Якщо він є розділовим знаком, то символ являє собою елемент масиву punctuation, і він замінюється на пробіл.

Після того, як цикли опрацьовують усі символи тексту, ми отримуємо текст зі зміненими розділовими знаками на пробіли.

Далі виконується основний цикл, який перебирає всі символи у масиві text до кінця тексту (за допомогою символу '\0'). У циклі ми шукаємо слово, що починається з вказівника word\_start.

Виконується додатковий цикл, який перевіряє, чи не є наступний символ пробілом. Якщо він є пробілом, то вказівник word\_start збільшується. Якщо наступного символу немає або він не є пробілом, то цикл завершується, і ми знаходимо кінець слова.

Із символа word\_start до символа word\_end перевіряється наявність повторюваних букв та записуємо цю інформацію у булевий масив repeated\_letters.

За допомогою циклу знову перевіряється масив repeated\_letters на наявність елементів, і якщо такі є, то слово виводиться на екран разом з переліком повторюваних букв.

На кожній ітерації основного циклу вказівник word\_start переміщується до наступного слова, починаючи з кінця попереднього слова до кінця тексту. Кінець програми.

**Завдання 2 “Рядкові функції”**

**Функція “memmove”**

Алгоритм називається my\_memmove та виконує перенесення блоку пам'яті з src в dest заданого розміру size. Аргументами функції є char\* dest (вказівник на початок області пам'яті, куди буде копіювання), char\* src (вказівник на початок області пам'яті, яку треба скопіювати) та int size (розмір копіювання).

Функція складається з циклу, де кожен байт із вказівника src копіюється в вказівник dest за допомогою оператора присвоєння "=". Цикл працює size разів, що гарантує повне копіювання src в dest.

Таким чином, функція my\_memmove є більш безпечною та більш гнучкою альтернативою бібліотечній функції memmove(). Наприклад, вона може використовуватися вмісто функцій memcpy() або strcpy() у випадку, коли потрібно копіювати область пам'яті, яка перетинається з іншими, тому що my\_memmove переносить дані байт за байтом та не східчується з розмірами масивів.

**Функція “strspn( )”**

Aлгоритм називається my\_strspn та виконує пошук у рядку str1 довжини найбільшого початкового підрядка, який містить тільки символи з рядка str2. Аргументами функції є char\* str1 (вказівник на рядок для пошуку) та char\* str2 (вказівник на рядок символів, які ми шукатимемо в str1).

Функція перевіряє на наявність покажчики str1 та str2, якщо вони пусті - повертається -1.

Далі функція пройшлається по рядку str1 символ за символом. Для кожного символу str1 виконується ітерація по символах рядка str2. Якщо в str2 знайдено відповідний символ, змінюється булева змінна найдене на true, в іншому випадку функція виходить з циклу, щоб знайти довжину str1, яку my\_strspn поверне в результаті роботи.

Останнім кроком функції є повернення значення len1 - довжини найбільшого початкового підрядка, який містить тільки символи з рядка str2.

Таким чином, функція my\_strspn є альтернативою функції strspn бібліотеки С++. Обидві функції виконують пошук найбільшого збігу між двома рядками, але my\_strspn більш гнучка, тому що вона дозволяє виконувати пошук в довільних рядках, а не тільки в рядках, які є нуль-термінованими масивами символів C.

**Завдання 3 “Обробка текстової інформації, що міститься у файлах”**

Допоміжний алгоритм призначений для заміни знаків пунктуації в тексті (розділових знаків крім крапок, комами, крапками з комами, знаками питання, знаками оклику, круглими, квадратними та фігурними дужками) на пробіли. Також він знаходить у введеному тексті всі слова, які містять задану користувачем букву, після чого виводить їх на екран, а також кількість таких слів.

Алгоритм починається з оголошення масиву символів, який містить список розділових знаків, що замінюються на пробіли. Масив цих знаків розділових знаків перебирається двічі: з якщо знак, який має бути змінений, знаходиться у списку, то його замінюється на пробіл.

Далі переходимо до читання тексту з клавіатури (з файлу) та знаходження слів, що містять користувачем задану букву в середині слова. Текст розбивається на окремі слова з допомогою функції "strtok()", яка використовується з рядком-роздільником, що включає пробіли, переноси рядків та символи возврату каретки. Якщо слово містить користувачем задану букву, то лічильник слів збільшується на одиницю, а саме слово виводиться на екран.

Результати виконання програми виводяться на екран користувачем, а сама програма завершується.

**3.Опис інтерфейсу програми (керівництво користувача)**

**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”**

Enter the text and the program will display words that have repeating characters, as well as all characters that are repeated in this word.

WARNING. Must be no more than 1 000 000 characters.

Read text from file? (y/n) n

Enter rhe text.

I'm on the road, empty and cold. To a distant destination, I don't know. Been thinking about you, we back in days of all. It's hard to admit it, I still miss you, miss you so. Flashbacks of our memories, the past is my enemy. And I'm drowning in a sad melody. Flashbacks of our memory, the past is my enemy. It keeps holding, holding on me. Come break the silence.

Word: distant | Repeating letters: t.

Word: destination | Repeating letters: i, n, t.

Word: Been | Repeating letters: e.

Word: thinking | Repeating letters: i, n.

Word: all | Repeating letters: l.

Word: still | Repeating letters: l.

Word: miss | Repeating letters: s.

Word: miss | Repeating letters: s.

Word: Flashbacks | Repeating letters: a, s.

Word: memories | Repeating letters: e, m.

Word: enemy | Repeating letters: e.

Word: drowning | Repeating letters: n.

Word: Flashbacks | Repeating letters: a, s.

Word: memory | Repeating letters: m.

Word: enemy | Repeating letters: e.

Word: keeps | Repeating letters: e.

Word: silence | Repeating letters: e.

**Завдання 2 “Рядкові функції”**

Enter the text and the program will display words that have repeating characters, as well as all characters that are repeated in this word.

WARNING. Must be no more than 1 000 000 characters.

Read text from file? (y/n) n

This program presents two artificial functions: "memmove()", "strspn()".

The "memmove()" function in C++ copies a given number of bytes from one area of memory to another, including the possibility of area intersections. This function allows you to safely copy data to memory locations that may overlap with the original memory area in case the data is copied from right to left.

The "strspn()" function in C++ finds the length of the initial substring of the string str, which consists only of characters from the string accept. That is, if str begins with a substring consisting only of the characters of the specified accept string, then strspn will return the length of that substring.

To use them you need to press:

"m" - if you want to use only the function "memmove()";

"s" - if you want to use only the function "strspn()";

"b" - if you want to use both functions.

"0" - end the program.

b

Enter the text for implementation the function "memmove()".

Who made up all the rules? We follow them like fools, believe them to be true, don't care to think them through.

Enter the word 1 for implementation the function "strspn()".

who

Enter the word 2 for implementation the function "strspn()".

whose

Array content: "Who made up all the rules? We follow them like fools, believe them to be true, don't care to think them through."

Program use the function "memmove()".

A copy of the contents of the previous array: "Who made up all the rules? We follow them like fools, believe them to be true, don't care to think them through."

Program use the function "my\_memmove()".

A copy of the contents of the previous array: "Who made up all the rules? We follow them like fools, believe them to be true, don't care to think them through."

Array content word 1: "who"

Array content word 2: "whose"

Program use the function "strspn()".

Number of identical elements in two rows: 3.

Program use the function "my\_strspn()".

Number of identical elements in two rows: 3.

Back to menu? (y/n)

n

End of program.

**Завдання 3 “Обробка текстової інформації, що міститься у файлах”**

The length of the text line mustn't exceed 80 characters.

Read text from file? (y/n)

n

Enter the text.

When you get older, plainer, saner, will you remember all the danger we came from?

Error: input string too long.

Enter the text.

When you get older, plainer, saner, will you remember all

Enter a letter to search for: e

When

get

older

plainer

saner

remember

Number of words that contain the character "e": 6.

Write data to a file? (y/n) y

The name of the output file will end with "\_out." and extension ".dat".

The first 1-3 characters can be taken from the names of the input file:

"1" - copy the first character of the input file;

"2" - copy the first two characters of the input file;

"3" - copy the first three characters of the input file;

"w" - write your own 1-3 characters.

w

Enter the first 1-3 letters of the file name: abc

Output file name: "abc\_out.dat".

Display the contents of the output file? (y/n)y

Text: "When you get older, plainer, saner, will you remember all"

Search letter "e".

Word:

When

get

lder

lainer

aner

emember

Number of words that contain the character "e": 6.

**Додаток (коди програм)**

**Завдання 1 “Обробка текстової інформації, що міститься у масиві символів”**

#include <iostream>

#include <fstream>

using namespace std;

void readTextFromFile(char\* filename, char\* lines, int len)

{

ifstream fin(filename);

if (!fin.is\_open()) {

cout << "\n\tError opening file!\n";

return;

}

if (fin.peek() == ifstream::traits\_type::eof()) {

cout << "\n\tFile is empty!\n";

fin.close();

return;

}

const int MAX\_LEN = 1000000;

char buf[MAX\_LEN+1];

int idx = 0;

while (fin.getline(buf, MAX\_LEN+1)) {

if (fin.fail()) {

cout << "\n\tFile read error!\n";

fin.close();

return;

}

int len2 = strlen(buf);

if (idx + len2 + 1 >= len) {

cout << "\n\tError: buffer exceeded!\n";

fin.close();

return;

}

memcpy(lines + idx, buf, len2);

idx += len2;

lines[idx++] = '\n';

}

lines[idx] = '\0';

fin.close();

cout << lines << "\n\n";

}

int main(int argc, char \*argv[])

{

const int MAX\_TEXT\_LENGTH = 1000001; // 1 000 000

char text[MAX\_TEXT\_LENGTH];

cout << "\n\tEnter the text and the program will display words that have repeating characters, as well as all characters that are repeated in this word.\n\tWARNING. Must be no more than 1 000 000 characters.\n\n\tRead text from file? (y/n) ";

bool x = false;

char choice;

while (x == false)

{

cin >> choice;

switch(choice)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

char choice2;

char filename[101];

if (choice == 'y')

{

x = false;

cout << "\n\tIs the filename specified on the command line? (y/n) ";

while (x == false)

{

cin >> choice2;

switch(choice2)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice2 == 'y')

{

if (argc == 2)

{

strcpy(filename, argv[1]);

}

else

{

cout << "\n\tThe file name was not specified. Enter file name (full file name): ";

cin >> filename;

}

}

if (choice2 == 'n')

{

cout << "\n\tEnter the file name (full file name): ";

cin >> filename;

}

readTextFromFile(filename, text, MAX\_TEXT\_LENGTH);

cout << "\n\t" << text;

}

if (choice == 'n')

{

cin.ignore();

cout << "\n\tEnter rhe text.\n\t";

cin.getline(text, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the text.\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

}

cout << "\n\n";

char punctuation[] = {'.', ',', ';', ':', '!','?', '(', ')', '[', ']', '{', '}', '<', '>'};

int len = sizeof(punctuation) / sizeof(punctuation[0]);

for (int i = 0; text[i] != '\0'; i++)

{

for (int j = 0; j < len; j++)

{

if (text[i] == punctuation[j])

{

text[i] = ' ';

break;

}

}

}

char\* word\_start = text;

while (\*word\_start != '\0')

{

while (\*word\_start == ' ' || \*word\_start == '\n' || \*word\_start == '\t')

{

word\_start++;

}

char\* word\_end = word\_start;

while (\*word\_end != '\0' && \*word\_end != ' ' && \*word\_end != '\n' && \*word\_end != '\t')

{

word\_end++;

}

if (word\_start == word\_end)

{

word\_start = word\_end;

continue;

}

bool repeated\_letters[256] = {0};

for (char\* letter = word\_start; letter < word\_end; letter++)

{

if (repeated\_letters[\*letter])

{

continue;

}

for (char\* next\_letter = letter + 1; next\_letter < word\_end; next\_letter++)

{

if (\*letter == \*next\_letter)

{

repeated\_letters[\*letter] = true;

break;

}

}

}

bool has\_repeated\_letters = false;

for (size\_t i = 0; i < 256; i++)

{

if (repeated\_letters[i])

{

has\_repeated\_letters = true;

break;

}

}

if (has\_repeated\_letters)

{

cout << "\tWord: ";

for (char\* print\_letter = word\_start; print\_letter < word\_end; print\_letter++)

{

cout << \*print\_letter;

}

cout << " | Repeating letters: ";

for (size\_t i = 0; i < 256; i++)

{

if (repeated\_letters[i])

{

cout << static\_cast<char>(i) << ", ";

}

}

cout << "\b\b.\n\n";

}

word\_start = word\_end;

}

cout << "\n\n";

return 0;

}

**Завдання 2 “Рядкові функції”**

#include <iostream>

#include <fstream>

using namespace std;

void readTextFromFile(char\* filename, char\* lines, int len) {

ifstream fin(filename);

if (!fin.is\_open()) {

cout << "\n\tError opening file!\n";

return;

}

if (fin.peek() == ifstream::traits\_type::eof()) {

cout << "\n\tFile is empty!\n";

fin.close();

return;

}

const int MAX\_LEN = 1000000;

char buf[MAX\_LEN+1];

int idx = 0;

while (fin.getline(buf, MAX\_LEN+1)) {

if (fin.fail()) {

cout << "\n\tFile read error!\n";

fin.close();

return;

}

int len2 = strlen(buf);

if (idx + len2 + 1 >= len) {

cout << "\n\tError: buffer exceeded!\n";

fin.close();

return;

}

memcpy(lines + idx, buf, len2);

idx += len2;

lines[idx++] = '\n';

}

lines[idx] = '\0';

fin.close();

cout << "\n\t" << lines << "\n\n";

}

void my\_memmove(char\* dest, char\* src, int size)

{

for (int i=0; i<size; i++)

{

dest[i] = src[i];

}

}

int my\_strspn(char\* str1, char\* str2)

{

if (!str1 || !str2)

{ // проверка на nullptr

return -1;

}

int len1 = 0;

while (str1[len1]) {

bool found = false;

for (int i = 0; str2[i]; i++) {

if (str1[len1] == str2[i]) {

found = true;

break;

}

}

if (!found) {

break;

}

len1++;

}

return len1;

}

int main(int argc, char \*argv[])

{

const int MAX\_TEXT\_LENGTH = 1000001; // 1 000 000

char text[MAX\_TEXT\_LENGTH];

char text1[MAX\_TEXT\_LENGTH];

char text2[MAX\_TEXT\_LENGTH];

cout << "\n\tEnter the text and the program will display words that have repeating characters, as well as all characters that are repeated in this word.\n\tWARNING. Must be no more than 1 000 000 characters.\n\n\tRead text from file? (y/n) ";

bool x = false;

char choice;

while (x == false)

{

cin >> choice;

switch(choice)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

for(;;)

{

cout << "\n\tThis program presents two artificial functions: \"memmove()\", \"strspn()\".\n\n\t The \"memmove()\" function in C++ copies a given number of bytes from one area of memory to another, including the possibility of area intersections. This function allows you to safely copy data to memory locations that may overlap with the original memory area in case the data is copied from right to left.\n\n\tThe \"strspn()\" function in C++ finds the length of the initial substring of the string str, which consists only of characters from the string accept. That is, if str begins with a substring consisting only of the characters of the specified accept string, then strspn will return the length of that substring.\n\tTo use them you need to press:\n\t\t\"m\" - if you want to use only the function \"memmove()\";\n\t\t\"s\" - if you want to use only the function \"strspn()\";\n\t\t\"b\" - if you want to use both functions.\n\t\t\"0\" - end the program.\n\t";

char choice3;

x = false;

while (x == false)

{

cin >> choice3;

switch(choice3)

{

case 'm':

x = true;

break;

case 's':

x = true;

break;

case 'b':

x = true;

break;

case '0':

cout << "\n\n\tEnd of program.\n\n";

return 0;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

char filename[101];

if (choice == 'y')

{

char choice2;

x = false;

cout << "\n\tIs the filename specified on the command line? (y/n)\n\t";

while (x == false)

{

cin >> choice2;

switch(choice2)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice2 == 'y')

{

if (argc == 2)

{

strcpy(filename, argv[1]);

}

else

{

cout << "\n\tThe file name was not specified. Enter file name (full file name): ";

cin >> filename;

}

}

if (choice2 == 'n')

{

cout << "\n\tEnter the file name (full file name): ";

cin >> filename;

}

readTextFromFile(filename, text, MAX\_TEXT\_LENGTH);

readTextFromFile(filename, text1, MAX\_TEXT\_LENGTH);

readTextFromFile(filename, text2, MAX\_TEXT\_LENGTH);

}

if (choice == 'n')

{

if (choice3 == 'm')

{

cin.ignore();

cout << "\n\tEnter the text for implementation the function \"memmove()\".\n\t";

cin.getline(text, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the text.\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

}

if (choice3 == 's')

{

cin.ignore();

cout << "\n\tEnter the word 1 for implementation the function \"strspn()\".\n\t";

cin.getline(text1, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the word 1 for implementation the function \"strspn()\".\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

cout << "\n\tEnter the word 2 for implementation the function \"strspn()\".\n\t";

cin.getline(text2, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the word 2 for implementation the function \"strspn()\".\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

}

if (choice3 == 'b')

{

cin.ignore();

cout << "\n\tEnter the text for implementation the function \"memmove()\".\n\t";

cin.getline(text, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the text.\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

cout << "\n\tEnter the word 1 for implementation the function \"strspn()\".\n\t";

cin.getline(text1, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the word 1 for implementation the function \"strspn()\".\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

cout << "\n\tEnter the word 2 for implementation the function \"strspn()\".\n\t";

cin.getline(text2, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the word 2 for implementation the function \"strspn()\".\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

}

}

if (choice3 == 'm')

{

char dest[sizeof(text)];

memmove(dest, text, sizeof(text));

cout << "\n\tArray content: \"" << text << "\"";

cout << "\n\n\tProgram use the function \"memmove()\".\n\tA copy of the contents of the previous array: \"" << dest << "\"\n";

char my\_dest[sizeof(text)];

my\_memmove(my\_dest, text, sizeof(text));

cout << "\n\tProgram use the function \"my\_memmove()\".\n\tA copy of the contents of the previous array: \"" << my\_dest << "\"\n";

}

if (choice3 == 's')

{

cout << "\n\tArray content word 1: \"" << text1 << "\"";

cout << "\n\tArray content word 2: \"" << text2 << "\"";

cout << "\n\n\tProgram use the function \"strspn()\".\n\tNumber of identical elements in two rows: " << strspn(text1, text2) << ".";

cout << "\n\n\tProgram use the function \"my\_strspn()\".\n\tNumber of identical elements in two rows: " << my\_strspn(text1, text2) << ".";

}

if (choice3 == 'b')

{

char dest[sizeof(text)];

memmove(dest, text, sizeof(text));

cout << "\n\tArray content: \"" << text << "\"";

cout << "\n\n\tProgram use the function \"memmove()\".\n\tA copy of the contents of the previous array: \"" << dest << "\"\n";

char my\_dest[sizeof(text)];

my\_memmove(my\_dest, text, sizeof(text));

cout << "\n\tProgram use the function \"my\_memmove()\".\n\tA copy of the contents of the previous array: \"" << my\_dest << "\"\n";

cout << "\n\tArray content word 1: \"" << text1 << "\"";

cout << "\n\tArray content word 2: \"" << text2 << "\"";

cout << "\n\n\tProgram use the function \"strspn()\".\n\tNumber of identical elements in two rows: " << strspn(text1, text2) << ".\n";

cout << "\n\n\tProgram use the function \"my\_strspn()\".\n\tNumber of identical elements in two rows: " << my\_strspn(text1, text2) << ".\n";

}

cout << "\n\tBack to menu? (y/n)\n\t";

char choice6;

x = false;

while (x == false)

{

cin >> choice6;

switch(choice6)

{

case 'y':

cout << "\033[2J\033[H";

choice = 'n';

x = true;

break;

case 'n':

cout << "\n\n\tEnd of program.\n\n";

return 0;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

}

}

**Завдання 3 “Обробка текстової інформації, що міститься у файлах”**

#include <iostream>

#include <fstream>

using namespace std;

void readTextFromFile(char\* filename, char\* lines, int len)

{

ifstream fin(filename);

if (!fin.is\_open())

{

cout << "\n\tError opening file!\n";

return;

}

if (fin.peek() == ifstream::traits\_type::eof())

{

cout << "\n\tFile is empty!\n";

fin.close();

return;

}

const int MAX\_LEN = 80;

char buf[MAX\_LEN+1];

int idx = 0;

while (fin.getline(buf, MAX\_LEN+1))

{

if (fin.fail())

{

cout << "\n\tFile read error!\n";

fin.close();

return;

}

int len2 = strlen(buf);

if (idx + len2 + 1 >= len)

{

cout << "\n\tError: buffer exceeded!\n";

fin.close();

return;

}

memcpy(lines + idx, buf, len2);

idx += len2;

lines[idx++] = '\n';

}

lines[idx] = '\0';

fin.close();

cout << "\n\t" << lines << "\n\n";

}

void replacePunctuation(char\* text)

{

char punctuation[] = {'.', ',', ';', ':', '!','?', '(', ')', '[', ']', '{', '}'};

int len = sizeof(punctuation) / sizeof(punctuation[0]);

for (int i = 0; text[i] != '\0'; i++)

{

for (int j = 0; j < len; j++)

{

if (text[i] == punctuation[j])

{

text[i] = ' ';

break;

}

}

}

}

void r1(char\* text) {

char punctuation[] = {' '};

int len = sizeof(punctuation) / sizeof(punctuation[0]);

for (int i = 0; text[i] != '\0'; i++)

{

for (int j = 0; j < len; j++)

{

if (text[i] == punctuation[j])

{

text[i++] = '\n';

text[i] = '\t';

break;

}

}

}

}

void my\_memmove(char\* dest, char\* src, int size)

{

for (int i=0; i<size; i++)

{

dest[i] = src[i];

}

}

void coutfile(char dest[81], char searchChar, int count, char\* word)

{

cout << "\n\tText: \"" << dest << "\"\n\tSearch letter \"" << searchChar << "\".\n\n\tWord:\n\n\t";

cout << word;

cout << "\n\tNumber of words that contain the character \"" << searchChar << "\": " << count << ".";

}

void printtofile(char\* filename, char dest[81], char searchChar, int count, char\* word)

{

ofstream file(filename);

if (!file.is\_open())

{

cerr << "\tError opening file.\n";

return;

}

file << "\n\tText: \"" << dest << "\"\n\tSearch letter \"" << searchChar << "\".\n\n\tWord:\n\n\t";

file << word;

file << "\n\tNumber of words that contain the character \"" << searchChar << "\": " << count << ".";

file.close();

}

int main(int argc, char \*argv[])

{

char filename[101];

char filename2[101];

const int MAX\_TEXT\_LENGTH = 80;

char text[MAX\_TEXT\_LENGTH + 1];

cout << "\n\tThe length of the text line mustn't exceed 80 characters.\n\tRead text from file? (y/n)\n\t";

bool x = false;

char choice;

while (x == false)

{

cin >> choice;

switch(choice)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice == 'y')

{

char choice2;

while (x == false)

{

cin >> choice2;

switch(choice2)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice2 == 'y')

{

if (argc == 2)

{

strcpy(filename, argv[1]);

}

else

{

cout << "\n\tThe file name was not specified. Enter file name (full file name): ";

cin >> filename;

}

}

if (choice2 == 'n')

{

cout << "\n\tEnter the file name (full file name): ";

cin >> filename;

}

readTextFromFile(filename, text, MAX\_TEXT\_LENGTH + 1);

cout << "\n\tDisplay input file text on screen? (y/n)\n\t";

char choice3;

x = false;

while (x == false)

{

cin >> choice3;

switch(choice3)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice3 == 'y')

{

cout << text;

}

}

if (choice == 'n')

{

cin.ignore();

cout << "\n\tEnter the text.\n\t";

cin.getline(text, MAX\_TEXT\_LENGTH);

while (cin.fail())

{

cerr << "\n\tError: input string too long.";

cerr << "\n\tEnter the text.\n\t";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(), '\n');

cin.getline(text, MAX\_TEXT\_LENGTH);

}

}

char dest[81];

my\_memmove(dest, text, 81);

replacePunctuation(text);

char searchChar;

cout << "\n\tEnter a letter to search for: ";

cin >> searchChar;

cout << endl;

int count = 0;

char word1[81] = {0};

char \*word = strtok(text, " \r\n"); // используем строку разделителей, включая символ возврата каретки

while (word != NULL)

{

if (strchr(word, searchChar) != NULL)

{

++count;

cout << "\t" << word << endl;

strcat(word1, word);

strcat(word1, " " );

}

word = strtok(NULL, " \r\n"); // также используем строку разделителей второй раз

}

r1(word1);

cout << "\n\tNumber of words that contain the character \"" << searchChar << "\": " << count << ".";

x = false;

cout << "\n\n\tWrite data to a file? (y/n) ";

while (x == false)

{

cin >> choice;

switch(choice)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice == 'y')

{

cout << "\n\tThe name of the output file will end with \"\_out.\" and extension \".dat\".";

cout << "\n\tThe first 1-3 characters can be taken from the names of the input file:";

cout << "\n\t\"1\" - copy the first character of the input file;";

cout << "\n\t\"2\" - copy the first two characters of the input file;";

cout << "\n\t\"3\" - copy the first three characters of the input file;";

cout << "\n\t\"w\" - write your own 1-3 characters.\n\t";

char choice5;

x = false;

while (x == false)

{

cin >> choice5;

switch(choice5)

{

case '1':

strncpy(filename2, filename, 1);

filename2[1] = '\0'; // задаем конец строки

x = true;

break;

case '2':

strncpy(filename2, filename, 2);

filename2[2] = '\0';

x = true;

break;

case '3':

strncpy(filename2, filename, 3);

filename2[3] = '\0';

x = true;

break;

case 'w':

cin.ignore();

cout << "\n\tEnter the first 1-3 letters of the file name: ";

cin >> filename2;

if(!cin || strlen(filename2) > 3 || strlen(filename2) < 1)

{

for (int i=0; i< 101; i++)

{

filename2[i] = 0;

}

strcpy(filename2, "bot");

}

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

char filename3[] = "\_out.dat";

strcat(filename2, filename3);

cout << "\n\tOutput file name: \"" << filename2 << "\".";

printtofile(filename2, dest, searchChar, count, word1);

cout << "\n\tDisplay the contents of the output file? (y/n)";

char choice6;

x = false;

while (x == false)

{

cin >> choice6;

switch(choice6)

{

case 'y':

x = true;

break;

case 'n':

x = true;

break;

default:

cout << "\n\tWhat did you mean?\n\tRetype your input.\n\t";

break;

}

}

if (choice6 == 'y')

{

coutfile(dest, searchChar, count, word1);

}

}

cout << "\n\n";

return 0;

}