Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

Кафедра кібербезпеки та програмного забезпечення

# Дисципліна: Базові методології та технології програмування

**Лабораторна робота №7**

**Тема:** **«ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНИХ ІНФОРМАЦІЇ ЗА СТАНДРТОМ UNICODE»**

|  |
| --- |
| Виконав: ст. гр. КБ-24-1 |
| Жуковська В.В |
| Перевірив: викладач  Коваленко А.С. |
|  |

Кропивницький

2024

**ТЕМА: ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНИХ ІНФОРМАЦІЇ ЗА СТАНДРТОМ UNICODE**

**МЕТА: МЕТА РОБОТИ ПОЛЯГАЄ У НАБУТТІ ҐРУНТОВНИХ ВМІНЬ І ПРАКТИЧНИХ НАВИЧОК СИНТЕЗУ АЛГОРИТМІВ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНОЇ (ТЕКСТОВОЇ) ІНФОРМАЦІЇ У КОДУВАННЯХ UTF-8 І CP866, ЇХ ПРОГРАМНОЇ РЕАЛІЗАЦІЇ МОВОЮ ПРОГРАМУВАННЯ МОВОЮ ПРОГРАМУВАННЯ С (ISO/IEC 9899:2018) ЗАДЛЯ РЕАЛІЗАЦІЇ ПРОГРАМНИХ ЗАСОБІВ У ВІЛЬНОМУ КРОСПЛАТФОРМОВОМУ CODE::BLOCKS IDE.**

розв’язування задачі 7.1:

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <locale.h>

#include <windows.h> // Для правильного відображення тексту в Windows

// Функція підрахунку символу 'й'

int count\_yi(const char \*text) {

int count = 0;

while (\*text) {

if (\*text == 'й') {

count++;

}

text++;

}

return count;

}

// Функція видалення голосних з речення

void remove\_vowels(char \*text) {

const char \*vowels = "аеєиіїоуюяAEIOUaeiou";

char result[256]; // Гарантуємо достатньо місця для нового тексту

int j = 0;

for (int i = 0; text[i] != '\0' && j < 255; i++) { // Обмеження для безпеки

if (!strchr(vowels, text[i])) {

result[j++] = text[i];

}

}

result[j] = '\0';

strcpy(text, result);

}

// Функція перевірки закінчення речення

int has\_valid\_ending(const char \*text) {

size\_t len = strlen(text);

if (len < 2) return 0; // Мінімальна довжина для "//"

return (len >= 2 && strcmp(&text[len - 2], "//") == 0) ||

(len >= 1 && (text[len - 1] == '-' || text[len - 1] == ';'));

}

// Функція видалення роздільних символів

void remove\_ending(char \*text) {

size\_t len = strlen(text);

if (len >= 2 && strcmp(&text[len - 2], "//") == 0) {

text[len - 2] = '\0';

} else if (len >= 1 && (text[len - 1] == '-' || text[len - 1] == ';')) {

text[len - 1] = '\0';

}

}

int main() {

// Встановлення правильного кодування для Windows-консолі

setlocale(LC\_ALL, "");

SetConsoleCP(65001); // Встановлює кодування UTF-8 для введення

SetConsoleOutputCP(65001); // Встановлює кодування UTF-8 для виводу

char input[256]; // Буфер для вводу

printf("Введіть речення: ");

if (!fgets(input, sizeof(input), stdin)) { // Перевірка на NULL (EOF)

printf("Помилка зчитування введення!\n");

return 1;

}

// Видалення символу нового рядка (якщо є)

input[strcspn(input, "\n")] = 0;

// Перевірка коректності вводу

if (!has\_valid\_ending(input)) {

printf("Некоректний формат вводу!\n");

return 1;

}

// Видалення роздільного символу

remove\_ending(input);

// Обробка речення

int yi\_count = count\_yi(input);

if (yi\_count > 0) {

printf("Кількість 'й': %d\n", yi\_count);

} else {

remove\_vowels(input);

printf("Модифіковане речення: %s\n", input);

}

return 0;

}

Вихідний код Zhukovsky-task\_7\_1:

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <locale.h>

#include <windows.h> // Для коректного відображення тексту у Windows-консолі

#define MAX\_LEN 256 // Максимальна довжина введеного речення

/\*\*

\* Функція підраховує кількість символів 'й' у тексті.

\* @param text Вхідний рядок.

\* @return Кількість входжень символу 'й'.

\*/

int count\_yi(const char \*text) {

int count = 0;

while (\*text) {

if (\*text == 'й') {

count++;

}

text++;

}

return count;

}

/\*\*

\* Функція видаляє голосні з вхідного рядка.

\* @param text Вхідний рядок, у якому видаляються голосні.

\*/

void remove\_vowels(char \*text) {

const char \*vowels = "аеєиіїоуюяAEIOUaeiou"; // Голосні української та англійської мов

char result[MAX\_LEN] = {0}; // Ініціалізований буфер для нового тексту

int j = 0;

// Видаляємо голосні, залишаючи лише приголосні

for (int i = 0; text[i] != '\0' && j < MAX\_LEN - 1; i++) {

if (!strchr(vowels, text[i])) {

result[j++] = text[i];

}

}

result[j] = '\0';

strncpy(text, result, MAX\_LEN); // Копіюємо відфільтрований рядок назад

}

/\*\*

\* Функція перевіряє, чи має рядок правильний завершальний символ (//, - або ;)

\* @param text Вхідний рядок.

\* @return 1, якщо рядок закінчується правильним символом; 0 — якщо ні.

\*/

int has\_valid\_ending(const char \*text) {

size\_t len = strlen(text);

if (len < 2) return 0; // Мінімальна довжина для "//"

return (len >= 2 && strcmp(&text[len - 2], "//") == 0) ||

(len >= 1 && (text[len - 1] == '-' || text[len - 1] == ';'));

}

/\*\*

\* Функція видаляє спеціальні символи (//, - або ;) з кінця рядка.

\* @param text Вхідний рядок.

\*/

void remove\_ending(char \*text) {

size\_t len = strlen(text);

if (len >= 2 && strcmp(&text[len - 2], "//") == 0) {

text[len - 2] = '\0';

} else if (len >= 1 && (text[len - 1] == '-' || text[len - 1] == ';')) {

text[len - 1] = '\0';

}

}

/\*\*

\* Головна функція програми.

\* Отримує речення від користувача, перевіряє його коректність, обробляє та виводить результат.

\* @return 0 при успішному виконанні.

\*/

int main() {

// Встановлення правильного кодування для Windows-консолі

setlocale(LC\_ALL, "uk\_UA.UTF-8");

SetConsoleCP(65001); // Встановлює кодування UTF-8 для введення

SetConsoleOutputCP(65001); // Встановлює кодування UTF-8 для виводу

char input[MAX\_LEN]; // Буфер для зберігання введеного тексту

printf("Введіть речення: ");

// Зчитуємо введене користувачем речення

if (fgets(input, sizeof(input), stdin) == NULL) {

printf("Помилка зчитування введення!\n");

return 1;

}

// Видаляємо символ нового рядка (\n), якщо він є

size\_t len = strlen(input);

if (len > 0 && input[len - 1] == '\n') {

input[len - 1] = '\0';

}

// Перевіряємо коректність формату речення

if (!has\_valid\_ending(input)) {

printf("Некоректний формат вводу!\n");

return 1;

}

// Видаляємо роздільний символ (//, - або ;)

remove\_ending(input);

// Обробка речення

int yi\_count = count\_yi(input);

if (yi\_count > 0) {

printf("Кількість 'й': %d\n", yi\_count);

} else {

remove\_vowels(input);

printf("Модифіковане речення: %s\n", input);

}

// Очікування натискання Enter перед виходом (щоб консоль не закривалася)

printf("\nНатисніть Enter для виходу...\n");

getchar();

return 0;

}

розв’язування задачі 7.2:

#include <stdio.h>

#include <locale.h>

#include <windows.h> // Для коректного відображення тексту в Windows-консолі

#define SIZE 14 // Кількість елементів у масиві

/\*\*

\* Функція підраховує кількість додатних чисел у масиві.

\* @param arr Вхідний масив чисел.

\* @param size Розмір масиву.

\* @return Кількість додатних чисел.

\*/

int count\_positive\_numbers(int arr[], int size) {

int count = 0;

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

if (arr[i] > 0) {

count++;

}

}

return count;

}

/\*\*

\* Функція обчислює суму додатних чисел у масиві.

\* @param arr Вхідний масив чисел.

\* @param size Розмір масиву.

\* @return Сума додатних чисел.

\*/

int sum\_positive\_numbers(int arr[], int size) {

int sum = 0;

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

if (arr[i] > 0) {

sum += arr[i];

}

}

return sum;

}

/\*\*

\* Функція заповнює масив введеними користувачем числами.

\* @param arr Масив для збереження введених чисел.

\* @param size Розмір масиву.

\*/

void input\_array(int arr[], int size) {

printf("Введіть %d чисел:\n", size);

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

printf("Число №%d: ", i + 1);

while (scanf("%d", &arr[i]) != 1) { // Перевірка введення

printf("Помилка! Введіть ціле число: ");

while (getchar() != '\n'); // Очищення буфера вводу

}

}

}

/\*\*

\* Головна функція програми.

\* Отримує масив чисел від користувача, обробляє його та виводить результати.

\* @return 0 при успішному виконанні.

\*/

int main() {

// Встановлення правильного кодування для Windows-консолі

setlocale(LC\_ALL, "uk\_UA.UTF-8");

SetConsoleCP(65001); // Встановлює кодування UTF-8 для введення

SetConsoleOutputCP(65001); // Встановлює кодування UTF-8 для виводу

int numbers[SIZE]; // Оголошення масиву

input\_array(numbers, SIZE); // Заповнення масиву

// Обчислення кількості та суми додатних чисел

int positive\_count = count\_positive\_numbers(numbers, SIZE);

int positive\_sum = sum\_positive\_numbers(numbers, SIZE);

// Вивід результатів

printf("\nКількість додатних чисел: %d\n", positive\_count);

printf("Сума додатних чисел: %d\n", positive\_sum);

// Очікування Enter перед закриттям

printf("\nНатисніть Enter для виходу...\n");

getchar();

getchar();

return 0;

}

Вихідний код Zhukovsky-task\_7\_2:

#include <stdio.h>

#include <locale.h>

#include <windows.h> // Для правильного відображення тексту у Windows-консолі

#define SIZE 14 // Кількість елементів у масиві

/\*\*

\* Функція підраховує кількість додатних чисел у масиві.

\* @param arr Вхідний масив чисел.

\* @param size Розмір масиву.

\* @return Кількість додатних чисел.

\*/

int count\_positive\_numbers(int arr[], int size) {

int count = 0; // Лічильник додатних чисел

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

if (arr[i] > 0) {

count++;

}

}

return count;

}

/\*\*

\* Функція обчислює суму додатних чисел у масиві.

\* @param arr Вхідний масив чисел.

\* @param size Розмір масиву.

\* @return Сума додатних чисел.

\*/

int sum\_positive\_numbers(int arr[], int size) {

int sum = 0; // Початкова сума додатних чисел

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

if (arr[i] > 0) {

sum += arr[i];

}

}

return sum;

}

/\*\*

\* Функція заповнює масив введеними користувачем числами.

\* @param arr Масив для збереження введених чисел.

\* @param size Розмір масиву.

\*/

void input\_array(int arr[], int size) {

printf("Введіть %d чисел:\n", size);

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

printf("Число №%d: ", i + 1);

while (scanf("%d", &arr[i]) != 1) { // Перевірка коректності вводу

printf("Помилка! Введіть ціле число: ");

while (getchar() != '\n'); // Очищення буфера вводу

}

}

}

/\*\*

\* Головна функція програми.

\* Отримує масив чисел від користувача, обробляє його та виводить результати.

\* @return 0 при успішному виконанні.

\*/

int main() {

// Встановлення правильного кодування для Windows-консолі

setlocale(LC\_ALL, "uk\_UA.UTF-8");

SetConsoleCP(65001); // Встановлює кодування UTF-8 для введення

SetConsoleOutputCP(65001); // Встановлює кодування UTF-8 для виводу

int numbers[SIZE]; // Оголошення масиву розміром 14 елементів

input\_array(numbers, SIZE); // Виклик функції для заповнення масиву

// Обчислення кількості та суми додатних чисел

int positive\_count = count\_positive\_numbers(numbers, SIZE);

int positive\_sum = sum\_positive\_numbers(numbers, SIZE);

// Вивід результатів

printf("\nКількість додатних чисел: %d\n", positive\_count);

printf("Сума додатних чисел: %d\n", positive\_sum);

// Очікування натискання Enter перед закриттям

printf("\nНатисніть Enter для виходу...\n");

getchar();

getchar();

return 0;

}

**Аргументи:**

1) Було реалізовано обробку масивів чисел та символьних рядків відповідно до стандарту Unicode.

2) Вивчено та застосовано кодування UTF-8 та CP866 у середовищі Code::Blocks.

3) Розроблено алгоритми оброблення масивів чисел для підрахунку додатних елементів і знаходження їх суми.

4) Реалізовано алгоритм роботи з текстовими рядками для обчислення кількості певних символів та модифікації тексту.

5) Досліджено відмінності між кодуваннями UTF-8 та CP866 при обробці текстових даних.

6) Опрацьовано способи перетворення символів та маніпуляції рядками у стандарті Unicode.

7) Навченося використовувати функції стандартної бібліотеки С (strlen, strcpy, strchr, fgets, scanf).

8) Вивчено механізм перевірки введених даних для коректної обробки масивів.

9) Застосовано динамічні структури даних у масивній обробці та текстовій інформації.

10) Реалізовано роботу з масивами фіксованої довжини в середовищі Code::Blocks.

11) Ознайомленося з особливостями обробки символьних даних у різних кодуваннях.

12) Відпрацьовано методи перевірки коректності введення користувачем.

13) Виконано тестування програмного коду для перевірки коректної роботи алгоритмів.

14) Досліджено проблеми кодування та відображення тексту в консолі Windows.

15) Вивчено можливості зміни кодової сторінки в Windows (chcp 65001 для UTF-8).

16) Використано функцію setlocale() для встановлення правильного кодування в Code::Blocks.

17) Досліджено методи видалення голосних із тексту шляхом перевірки кожного символу.

18) Реалізовано фільтрацію та перетворення текстових даних з використанням C.

19) Навченося правильно використовувати if-else та for у масивній обробці.

20) Ознайомленося з особливостями циклічної перевірки елементів масиву.

21) Виконано алгоритм знаходження певного символу у тексті (й у рядку).

22) Досліджено методи зчитування рядків та чисел із клавіатури.

23) Використано засоби перетворення числових значень у текстовому представленні.

24) Виявлено особливості обробки негативних значень у масивах чисел.

25) Реалізовано алгоритм підрахунку кількості певних символів у рядку.

26) Використано функцію fgets() для безпечного введення рядків.

27) Ознайомленося з проблемами буфера введення та способами його очищення.

28) Досліджено методи перевірки наявності розділових знаків у рядку.

29) Відпрацьовано механізм видалення небажаних символів у тексті.

30) Опрацьовано роботу з масивами цілих чисел у Code::Blocks.

31) Навченося відображати результати в консолі Windows з правильним форматуванням.

32) Реалізовано алгоритм перевірки правильності введення символів у рядку.

33) Виконано структуризацію коду для підвищення його читабельності та ефективності.

34) Вивчено основи модульного програмування в С (розбиття на функції).

35) Навченося використовувати константи (#define) для розміру масиву.

36) Реалізовано алгоритм перевірки завершальних символів у рядку (//, -, ;).

37) Вивчено методи видалення зайвих символів у тексті (strcpy, strncpy).

38) Виконано перевірку роботи програми з порожніми рядками.

39) Ознайомленося з особливостями роботи Code::Blocks із Unicode.

40) Досліджено особливості роботи кодування Windows-1251 та UTF-8 у консолі.

41) Вивчено засоби друку форматованих даних (printf, %d, %s).

42) Реалізовано збереження та обробку даних у масиві без втрати значень.

43) Досліджено оптимізацію алгоритмів для обробки великих текстів.

44) Використано функцію getchar() для затримки завершення програми.

45) Виконано тестування на реальних прикладах із тест-сьюту TS\_7\_1 та TS\_7\_2.

46) Переконанося у правильності виконання програмного коду шляхом порівняння очікуваних і фактичних результатів.

47) Ознайомленося з методами відлагодження коду та виправлення помилок.

48) Використано Code::Blocks IDE для розробки та компіляції програмного коду.

49) Досліджено підтримку мовних символів у стандартній бібліотеці С (ISO/IEC 9899:2018).

50) Досягнуто мету лабораторної роботи: набуто практичних навичок оброблення масивів та текстової інформації в кодуванні Unicode.