Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 7

з навчальної дисципліни

“Базові методології та технології програмування”

ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНОЇ ІНФОРМАЦІЇ

ЗАВДАННЯ ВИДАВ

доцент кафедри кібербезпеки та програмного забезпечення

Доренський О. П.

[https://github.com/odorenskyi/](https://github.com/odorenskyi/Dmytro-Parkhomenko-KB18)

ВИКОНАВ

студент академічної групи КБ-20

Карпіков О.С.

ПЕРЕВІРИВ

ст. викладач кафедри кібербезпеки   
та програмного забезпечення

Поліщук Л. І.

Кропивницький – 2021

**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.

**Завдання до лабораторної роботи:**

1. Реалізувати програмний забезпечення розв’язування задачі 7.1.
2. Реалізувати програмне забезпечення розв’язування задачі 7.2.

**Варіант № 11**

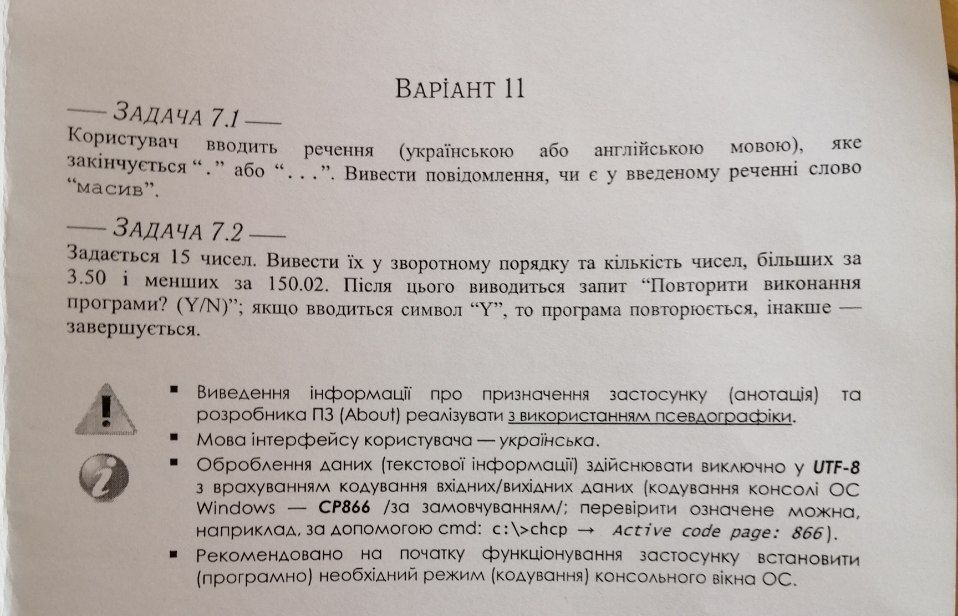


Рисунок 1 – Варіант № 11

**Завдання 1**

* 1. **Формалізація задачі**
     1. **Строга постановка задачі**

**Вихідні данні:** Повідомлення чи присутнє слово «масив» у введеному речені.

**Вхідні данні:** userStr – строка яку вводить користувач у консолі, максимальною довжиною строки є 1000 символів.

* 1. **Лістинг до завдання 1**

#include <stdio.h>

#include <wchar.h>

#include <locale.h>

#include <windows.h>

void **setLocalUtf\_8**(){

// the procedure configures the console window to work with utf-8 encoding

\_wsetlocale(LC\_ALL,L"");

SetConsoleOutputCP(CP\_UTF8);

SetConsoleCP(CP\_UTF8);

}

void **devInfo**(){

// the devInfo () procedure displays information about the application and developer of this listing

wprintf(L"╔═══════════════════════════════════════════════════════════════════════════════════════╗\n");

wprintf(L"║Користувач вводить речення (українською або англійською мовою), яке закінчується ║\n");

wprintf(L"║на \".\" або \"...\". Застосунок виводить повідомлення чи є у введеному реченні, ║\n");

wprintf(L"║слово \"масив\". ║\n");

wprintf(L"║Виконав студент групи КБ20 Карпіков Олег Сергійович ║\n");

wprintf(L"╚═══════════════════════════════════════════════════════════════════════════════════════╝\n");

}

void **pressAnyKey**(){

// A procedure that does not close the console window until any key on the keyboard is pressed.

// Displays the corresponding system message: Press any key to continue. . .

wprintf(L"\n");

system("pause");

}

int **main**(){

setLocalUtf\_8();

devInfo();

wchar\_t userStr[1000];

unsigned long userStrSize = 0;

wprintf(L"Введіть речення: ");

// read userStr in utf-8

ReadConsole(GetStdHandle(STD\_INPUT\_HANDLE), userStr, 1000,&userStrSize, NULL);

userStrSize -= 3;

// Remove "extra" spaces at the end of a sentence

while(userStr[userStrSize]==L' ')

{

userStrSize--;

}

// Convert characters to one case

for(long currentLetter = 0; currentLetter <= (long)(userStrSize); currentLetter++){

if((userStr[currentLetter] >= L'a' && userStr[currentLetter] <= 'z') || (userStr[currentLetter] >= L'а' && userStr[currentLetter] <= L'я'))

{

userStr[currentLetter] += L'А' - L'а';

}

}

// Сheck the last character of the line. If the last character does not match the condition of the task,

// then the corresponding message will be displayed: The entered sentence does not correspond to the condition of the problem !!!

if ( userStr[userStrSize] == L'.')

{

// Check the presence of the word "array" in the sentence

if (wcsstr(userStr,L"МАСИВ") || wcsstr(userStr,L"ARRAY"))

{

wprintf(L"Cлово \"масив\" присутнє у введеному речені!");

}else{

wprintf(L"Cлова \"масив\" немає у введеному речені!");

}

}else{

wprintf(L"Введене речення не відповідає умові задачі!!!");

}

pressAnyKey();

return 0;

}

Тестування цього лістингу показано у «Додатку A»

**Завдання 2**

* 1. **Формалізація задачі**
     1. **Строга постановка задачі**

**Вихідні данні:** Введені числа у зворотньому порядку;

Кількістть введених чисел у діапазоні (3,50;150,02).

**Вхідні данні:** arrayNumber – масив із 9 дійсних чисел, діапазон значень одного елементу масиву [-4 294 967 296; 4 294 967 296]

* 1. **Лістинг до завдання 2**

#include <stdio.h>

#include <wchar.h>

#include <locale.h>

#include <windows.h>

void **setLocalUtf\_8**(){

// the procedure configures the console window to work with utf-8 encoding

\_wsetlocale(LC\_ALL,L"");

SetConsoleOutputCP(CP\_UTF8);

SetConsoleCP(CP\_UTF8);

}

void **devInfo**(){

// the devInfo () procedure displays information about the application and developer of this listing

wprintf(L"╔═════════════════════════════════════════════════════════════════════════════════════════╗\n");

wprintf(L"║Користувач вводить 15 чисел. Застосунок виводить числа у зворотньому порядку та кількість║\n");

wprintf(L"║чисел більших за 3.50 і менших за 150,02. ║\n");

wprintf(L"║Виконав студент групи КБ20 Карпіков Олег Сергійович ║\n");

wprintf(L"╚═════════════════════════════════════════════════════════════════════════════════════════╝\n");

}

void **pressAnyKey**(){

// A procedure that does not close the console window until any key on the keyboard is pressed.

// Displays the corresponding system message: Press any key to continue. . .

wprintf(L"\n");

system("pause");

}

int **main**()

{

wchar\_t resultDialog;

setLocalUtf\_8();

do{

system("cls");

devInfo();

double arrayNumber[15];

int countNumber = 0;

for(char i = 0; i < 15; i++)

{

wprintf(L"Введіть %d число масиву: ", i+1);

wscanf(L"%lf", &arrayNumber[i]);

}

wprintf(L"Введені числа у зворотньому порядку: ");

for(char i = 14; i > -1; i--)

{

wprintf(L"%lf", arrayNumber[i]);

if(i > 0)

{

wprintf(L", ");

}else{

wprintf(L".\n");

}

if(arrayNumber[i]>3.50 && arrayNumber[i] < 150.02)

{

countNumber++;

}

}

wprintf(L"Кількістть чисел у діапазоні (3,50;150,02): %d\n", countNumber);

wprintf(L"Повторити виконання програми? (Y/N): ");

getwchar();

wscanf(L"%lC", &resultDialog);

}while( resultDialog == L'Y'|| resultDialog == L'y');

return 0;

}

Тестування цього лістингу показано у «Додатку Б»

**Висновок**

Під час лабораторної роботи студентам було запропоновано у набути ґрунтові вміння і практичні навички 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.

Основна проблема у реалізаії такої задачі у тому , що OS Windows не вміє «адекватно» працювати із багатобайтними кодуваннями. Ця операційна система може працювати «виключно» у кодуванні UTF-16. Працювати у UTF-8 неможливо в основному тому, що Windows не допускає UTF-8 в якості системної кодової сторінки ANSI, навіть якщо вона має кодову сторінку ANSI для UTF-8, кодову сторінку 65001. Здається, є кілька причин для цього:

* Коли Unicode був новим, Microsoft вирішила, що UCS-2 буде кращим способом підтримки Unicode. У той час Unicode був 16-бітовим.
* У Windows є одна кодова сторінка ANSI для кожного підтримуваного мови, на відміну від Unix подібних систем, де мова і кодування можуть бути встановлені незалежно.
* Кодова сторінка 65001 (UTF-8 це CP 65001) не працює всюди. Зокрема, він не працює з деякими функціями для оброблення багатобайтних символів у Windows, де багатобайтові символи вимагають один або два байти, тоді як UTF-8 вимагає від одного до чотирьох байтів.

Так неодмінно можна сказати що у 2019 році

Розробники Windows ось що говорять з цього приводу :

« Оскільки Windows працює в UTF-16 ( WCHAR), вам може знадобитися перетворити дані UTF-8 у UTF-16 (або навпаки), щоб взаємодіяти з API Windows.»

Для переведення введених данних з UTF-16 в UTF-8 Microsoft рекомендує використовувати такі дві функції реалізовані у бібліотеці windows.h: [MultiByteToWideChar](https://docs.microsoft.com/en-us/windows/desktop/api/stringapiset/nf-stringapiset-multibytetowidechar) та [WideCharToMultiByte](https://docs.microsoft.com/en-us/windows/desktop/api/stringapiset/nf-stringapiset-widechartomultibyte).

[MultiByteToWideChar](https://docs.microsoft.com/en-us/windows/desktop/api/stringapiset/nf-stringapiset-multibytetowidechar) і [WideCharToMultiByte](https://docs.microsoft.com/en-us/windows/desktop/api/stringapiset/nf-stringapiset-widechartomultibyte) дозволяє конвертувати між UTF-8 і UTF-16 ( WCHAR) (і інших кодових сторінок). Це особливо корисно, коли застарілий API Win32 може розуміти лише WCHAR. Ці функції дозволяють перетворити вхід UTF-8 WCHAR на перехід в API -W, а потім перетворити будь-які результати назад, якщо це необхідно. При використанні цих функцій із CodePageвстановленим значенням CP\_UTF8 використовується dwFlags або 0 , або MB\_ERR\_INVALID\_CHARS, інакше  ERROR\_INVALID\_FLAGS.

Але під час реалізації програмного застосунку до задачі 7.1 з’явилася така проблема: функції для «читання» розширених символів з потоку вводу нормально «зчитували» лише «однобайтні» символи ( наприклад функція wscanf(), \_getws() реалізовані у бібліотеці wchar.h ). Багатобайтні символи просто інтерпретувались як нуль символи («\0»).

Для вирішення цієї проблеми було застосовано функцію реалізовану у бібліотеці «windows.h» ReadConsole().

**Реалізація функції ReadConsole():**

BOOL WINAPI ReadConsole(

\_In\_     HANDLE  hConsoleInput,

\_Out\_    LPVOID  lpBuffer,

\_In\_     DWORD   nNumberOfCharsToRead,

\_Out\_    LPDWORD lpNumberOfCharsRead,

\_In\_opt\_ LPVOID  pInputControl

);

**Параметри функції:**

* hConsoleInput [in]  
  Дескриптор вхідного буфера консолі. Дескриптор повинен мати право доступу **GENERIC\_READ** . Для отримання додаткової інформації див. [Захист консольного буфера та права доступу](https://docs.microsoft.com/en-us/windows/console/console-buffer-security-and-access-rights) .
* lpBuffer [out]  
  Вказівник на буфер, який приймає дані, прочитані з вхідного буфера консолі.
* nNumberOfCharsToRead [in]  
  Кількість символів, які потрібно прочитати. Розмір буфера, на який вказує параметр lpBuffer, повинен бути не менше nNumberOfCharsToRead \* sizeof(TCHAR)байтів.
* lpNumberOfCharsRead [out]  
  Вказівник на змінну, яка отримує кількість фактично прочитаних символів.
* pInputControl [in, optional]  
  Вказівник на структуру [**CONSOLE\_READCONSOLE\_CONTROL,**](https://docs.microsoft.com/en-us/windows/console/console-readconsole-control) яка визначає керуючий символ для сигналізації закінчення операції зчитування. Цей параметр може мати значення **NULL** .

Цей параметр за замовчуванням вимагає введення Unicode. Для режиму ANSI встановіть для цього параметра значення **NULL** .

**Повертаєме значення**

Якщо функція успішна, повертається значення відміне від нуля.

Якщо функція виходить з ладу, повернене значення дорівнює нулю.

ReadConsole читає введення з клавіатури з буфера введення консолі. Він поводиться як функція [ReadFile](https://docs.microsoft.com/en-us/windows/win32/api/fileapi/nf-fileapi-readfile) , за винятком того, що він може читати як в Unicode (широкосимвольний), так і в режимі ANSI. Щоб мати програми, які підтримують єдиний набір джерел, сумісних з обома режимами, використовують ReadConsole, а не ReadFile . Хоча ReadConsole можна використовувати лише з дескриптором вхідного буфера консолі, ReadFile можна використовувати з іншими дескрипторами (наприклад, файлами або конвеєрами ). ReadConsole не працює, якщо використовується зі стандартним дескриптором, який був перенаправлений як щось інше, ніж консольний дескриптор.

Завдяки тому, що тепер данні введені користувачем можливо було «нормально» прочитати подальша робота із завданнями полягала у реалізації базових «методів» обробки масивів таких як сортування , пошук та заміна певних елементів, тощо.

|  |  |
| --- | --- |
| Назва тестового набору  Test Suite Description | TS\_7\_1 |
| Назва проекта / ПЗ  Name of Project / Software | KARPIKOV-TASK\_7\_1.exe |
| Рівень тестування  Level of Testing | системний / System Testing |
| Автор тест-сьюта  Test Suite Author | Karpikov Oleg |
| Виконавець  Implementer | Karpikov Oleg |

**Додаток А**

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест-кейса / Test Case ID | Дії (кроки) /  Action  (Test Steps) | Очікуваний  результат /  Expected Result | Результат тестування (пройшов/не вдалося/ заблокований) /  Test Result (passed/failed/ blocked) |
| 01 | 1.Запустити застосунок | Мова інтерфейсу користувача – українська. | Passed |
| 02 | 1.Запустити застосунок | У консольному вікні присутня інформація про призначення застосунку та його розробника. Ця інформація оформлена за допомогою псевдографіки. | Passed |
| 03 | 1.Запустити застосунок  2.Ввести таку строку: array. | У вікні застосунку:  Cлово "масив" присутнє у введеному речені!  Press any key to continue . . . | Passed |
| 04 | 1.Запустити застосунок  2.Ввести таку строку: масив. | У вікні застосунку:  Cлово "масив" присутнє у введеному речені!  Press any key to continue . . . | Passed |
| 05 | 1.Запустити застосунок  2.Ввести таку строку: масив…! | У вікні застосунку:  Введене речення не відповідає умові задачі!!!  Press any key to continue . . . | Passed |
| 06 | 1.Запустити застосунок  2.Ввести таку строку: array …! | У вікні застосунку:  Введене речення не відповідає умові задачі!!!  Press any key to continue . . . | Passed |
| 07 | 1.Запустити застосунок  2.Ввести таку строку: Домашня дорама та array. | У вікні застосунку:  Cлово "масив" присутнє у введеному речені!  Press any key to continue . . . | Passed |
| 08 | 1.Запустити застосунок  2.Ввести таку строку: deadly? | У вікні застосунку:  Введене речення не відповідає умові задачі!!!  Press any key to continue . . . | Passed |
| 09 | 1.Запустити застосунок  2.Ввести таку строку: deadly масив!. | У вікні застосунку:  Cлово "масив" присутнє у введеному речені!  Press any key to continue . . . | Passed |
| 10 | 1.Запустити застосунок  2.Ввести таку строку: Домашня дорама та масив. | У вікні застосунку:  Cлово "масив" присутнє у введеному речені!  Press any key to continue . . . | Passed |

|  |  |
| --- | --- |
| Назва тестового набору  Test Suite Description | TS\_7\_2 |
| Назва проекта / ПЗ  Name of Project / Software | KARPIKOV-TASK\_7\_2.exe |
| Рівень тестування  Level of Testing | системний / System Testing |
| Автор тест-сьюта  Test Suite Author | Karpikov Oleg |
| Виконавець  Implementer | Karpikov Oleg |

**Додаток Б**

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест-кейса / Test Case ID | Дії (кроки) /  Action  (Test Steps) | Очікуваний  результат /  Expected Result | Результат тестування (пройшов/не вдалося/ заблокований) /  Test Result (passed/failed/ blocked) |
| 01 | 1.Запустити застосунок | Мова інтерфейсу користувача – українська. | Passed |
| 02 | 1.Запустити застосунок | У консольному вікні присутня інформація про призначення застосунку та його розробника. Ця інформація оформлена за допомогою псевдографіки. | Passed |
| 03 | 1.Запустити застосунок | У вікні застосунку:  Введіть 1 число масиву: | Passed |
| 04 | 1.Запустити застосунок  2.Ввести : 1 | У вікні застосунку:  Введіть 2 число масиву: | Passed |
| 05 | 1.Запустити застосунок  2.Ввести : 0 | У вікні застосунку:  Введіть 2 число масиву: | Passed |
| 06 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 0 | У вікні застосунку:  Введіть 3 число масиву: | Passed |
| 07 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 5  4. Ввести : 6  5. Ввести : 2  6. Ввести : 69  7. Ввести : 6  8. Ввести : 6  9. Ввести : 47  10. Ввести : 63  11.Ввести : 0  12. Ввести : 15  13. Ввести : 22  14. Ввести : 56  15. Ввести : 1  16. Ввести : 2 | У вікні застосунку:  Введені числа у зворотньому порядку: 2,000000, 1,000000, 56,000000, 22,000000, 15,000000, 0,000000, 63,000000, 47,000000, 6,000000  , 6,000000, 69,000000, 2,000000, 6,000000, 5,000000, 1,000000.  Кількістть чисел у діапазоні (3,50;150,02): 10  Повторити виконання програми? (Y/N): | Passed |
| 08 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 5  4. Ввести : 6  5. Ввести : 2  6. Ввести : 69  7. Ввести : 6  8. Ввести : 6  9. Ввести : 47  10. Ввести : 63  11.Ввести : 0  12. Ввести : 15  13. Ввести : 22  14. Ввести : 56  15. Ввести : 1  16. Ввести : 2  17. Ввести : Y | У вікні застосунку:  Введіть 1 число масиву: | Passed |
| 09 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 5  4. Ввести : 6  5. Ввести : 2  6. Ввести : 69  7. Ввести : 6  8. Ввести : 6  9. Ввести : 47  10. Ввести : 63  11.Ввести : 0  12. Ввести : 15  13. Ввести : 22  14. Ввести : 56  15. Ввести : 1  16. Ввести : 2  17. Ввести : N | Застосунок закриється. | Passed |
| 10 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 5  4. Ввести : 6  5. Ввести : 2  6. Ввести : 69  7. Ввести : 6  8. Ввести : 6  9. Ввести : 47  10. Ввести : 63  11.Ввести : 0  12. Ввести : 15  13. Ввести : 22  14. Ввести : 56  15. Ввести : 1  16. Ввести : 2  17. Ввести : y | У вікні застосунку:  Введіть 1 число масиву: | Passed |
| 11 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 5  4. Ввести : 6  5. Ввести : 2  6. Ввести : 69  7. Ввести : 6  8. Ввести : 6  9. Ввести : 47  10. Ввести : 63  11.Ввести : 0  12. Ввести : 15  13. Ввести : 22  14. Ввести : 56  15. Ввести : 1  16. Ввести : 2  17. Ввести : n | Застосунок закриється. | Passed |
| 12 | 1. Запустити застосунок  2 .Ввести : 1  3. Ввести : 5  4. Ввести : 6  5. Ввести : 2  6. Ввести : 69  7. Ввести : 6  8. Ввести : 6  9. Ввести : 47  10. Ввести : 63  11.Ввести : 0  12. Ввести : 15  13. Ввести : 22  14. Ввести : 56  15. Ввести : 1  16. Ввести : 2  17. Ввести : 12 | Застосунок закриється. | Passed |