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 КІ-23

Чепіль В.О.

ПЕРЕВІРИВ

ст. викладач кафедри кібербезпеки   
та програмного забезпечення

Дрєєва Г. М.

Кропивницький – 2024

**ТЕМА: ПРОГРАМНА РЕАЛІЗАЦІЯ ОБРОБЛЕННЯ МАСИВІВ ДАНИХ ТА СИМВОЛЬНОЇ ІНФОРМАЦІЇ ЗА СТАНДАРТОМ UNICODE**

**МЕТА:** Набути 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:**

1. У веббраузері перейти за посиланням https://github.com/ та зареєструватись на вебсервісі системи контролю версій GitHub.

2. Після створення облікового запису (account) і підтвердження реєстрації через електронну пошту, у Personal settings заповнити Public profile: ♣ Name: власне ім’я й прізвище латиницею, ♣ Profile picture: офіційне фото (ділового стилю), ♣ URL: www.kntu.kr.ua, ♣ Company: Central Ukrainian National Technical University, ♣ Location: Ukraine, Kropyvnytskyi.

3. Продемонструвати викладачеві створений обліковий запис, після чого надіслати GitHub username на [dorenskyiop@kntu.kr.ua](mailto:dorenskyiop@kntu.kr.ua).

4. Отримати у викладача завдання за варіантом, задокументувати мету лабораторної роботи, завдання, номер варіанту.

5. На одному з локальних дисків ПК (D:, E: чи ін.) створити теку BMTP-LAB7-прізвище, в ній ― теки prj, Software та TestSuite.

6. Виконати аналіз умови і постановку задачі 7.1, встановлених вимог до програмного засобу, який розроблятиметься, на основі чого розробити й задокументувати тест-сьют TS\_7\_1, який повинен складатись з не менш як десяти тест-кейсів.

7. Виконати процедурно-орієнтовану алгоритмізацію задачі 7.1 з врахуванням вимоги про програмне оброблення текстової інформації у UTF-8, при чому кодування вхідних і вихідних даних залежить від джерела й приймача інформації відповідно; здобутий алгоритм розв’язування задачі 7.1 задокументувати.

8. В Code::Blocks IDE створити новий проект консольного застосунку з назвою прізвище-task\_7\_1 у /prj.

9. Алгоритм розв’язування задачі 7.1 реалізувати мовою програмування С (С18), проект прізвище-task\_7\_1 закрити.

10. З теки проекта прізвище-task\_7\_1 за допомогою "Блокнот" відкрити файл main.c.

11. Перезберегти файл main.c у UTF-8 (Файл → Зберегти як → Кодування → UTF-8 → Зберегти).

12. Закрити файл main.c. 44 Базові методології та технології програмування

13. В Code::Blocks IDE відкрити прізвище-task\_7\_1, скомпілювати і закрити проект; створений у /prj файл програми прізвищеtask\_7\_1.ехе скопіювати у /BMTP-LAB7-прізвище/Software.

14. За допомогою тестового набору TS\_7\_1 із теки /TestSuite, виконати системне тестування прізвище-task\_7\_1.ехе, який знаходиться у теці /BMTP-LAB7-прізвище/Software; отриманий тестовий артефакт — заповнений тест-сьют TS\_7\_1 — зберегти та долучити до звіту як додаток.

15. За умови, що статус всіх тест-кейсів TS\_7\_1 є passed, вихідний код (лістинг) проекту прізвище-task\_7\_1 задокументувати; у іншому випадку слід виконати відлагодження запропонованого алгоритму і/або ПЗ, проект прізвище-task\_7\_1 скомпілювати, системне тестування додатка прізвище-task\_7\_1.ехе повторити, результати повторної реалізації й тестування задокументувати.

16. Виконати аналіз умови і постановку задачі 7.2, встановлених вимог до програмного засобу, який розроблятиметься, на основі чого розробити й задокументувати тест-сьют TS\_7\_2, який повинен складатись з не менш як десяти тест-кейсів.

17. Виконати процедурно-орієнтовану алгоритмізацію задачі 7.2, отриманий алгоритм розв’язування задачі 7.2 задокументувати.

18. В Code::Blocks IDE створити новий проект консольного застосунку з назвою прізвище-task\_7\_2 у /prj.

19. Алгоритм розв’язування задачі 7.2 реалізувати мовою програмування С (С18), скомпілювати проект; створений файл програми прізвище-task\_7\_2.ехе скопіювати у /Software.

20. За допомогою TS\_7\_2 із /TestSuite, виконати системне тестування /BMTP-LAB2-прізвище/Software/прізвище-task\_7\_2.ехе; заповнений TS\_7\_2 долучити до звіту як додаток.

21. За умови, що статус всіх тест-кейсів TS\_7\_2 є passed, вихідний код (лістинг) проекту прізвище-task\_7\_2 задокументувати; у іншому випадку слід виконати відлагодження запропонованого алгоритму і/або ПЗ, проект прізвище-task\_7\_2 скомпілювати, системне тестування додатка прізвище-task\_7\_2.ехе повторити, результати повторної реалізації й тестування задокументувати.

22. Одержані результати виконання завдань лабораторної роботи (/BMTP-LAB7-прізвище) зберегти на флеш-накопичувач, у хмарному сховищі даних тощо. 45

23. Проаналізувати хід виконання завдань і одержані результати, сформулювати обґрунтовані висновки15 та викласти їх обсягом не менше двох сторінок машинного (комп’ютерного) тексту; також слід зазначити особисті враження від виконання лабораторної роботи, викласти вмотивовані пропозиції, обґрунтовані зауваження, конструктивну критику16, рекомендації тощо.

24. Інсталювати Git (Git Bash) https://git-scm.com/book/uk/v2/ВступІнсталяція-Git, використовуючи “Pro Git” <https://gitscm.com/book/uk/v2>.

25. На github.com/student-name створити Git-репозиторій (New → Create a new repository) та за допомогою Git (Git Bash) завантажити у нього теку BMTP-LAB7-прізвище (git init, git add, git commit, git branch, git remote, git push; https://techglimpse.com/git-push-github-token-based-passwordless/), а також звіт, оформлений згідно з ДСТУ 3008:2015 зі структурою й змістом, визначеними цим порядком виконання роботи.

26. Додати викладача і одногрупників до спільної роботи над проєктом BMTP-LAB7-прізвище: Settings → Access → Collaborators → Add people → Add NAME to REPOSITORY (NAME викладача — на дошці в аудиторії, у якій проходить заняття).

27. У комітах Git-репозиторію BMTP-LAB7-прізвище (https://github.com/ . . . /commits/main/) відреагувати і/або відповісти на коментарі співучасників ІТ-проєкту ( Commits), захистити результати й здобути зарахування роботи викладачем.

**Варіант 18**

**Аналіз задачі 7.1:**

Вх.: Речення, що закінчується на « | »

Вих.: Повідомлення що оповіщає чи є слово «Кропивницький» в

реченні.

Процес: Обробка у UTF-8.

**Лістинг 7.1:**

#include <stdio.h>

int main(void)

{

char list[100];

int reckon = 0;

printf("%c%c%c%c%c%c%c%c %c%c%c%c%c%c%c:",-115,-96,-81,-88,-24,105,-30,-22, -32,-91,-25,-91,-83,-83,-17);

fgets(list, sizeof(list), stdin);

for (int i = 0; i < 100; i++) {

if (list[i] == -118 && list[i+1] == -32 && list[i+2] == -82 && list[i+3] == -81 && list[i+4] == -88 && list[i+5] == -94 && list[i+6] == -83

&& list[i+7] == -88 && list[i+8] == -26 && list[i+9] == -20 && list[i+10] == -86 && list[i+11] == -88 && list[i+12] == -87) {

reckon = reckon + 1;

}

if (list[i] == '|') {

break;

}

}

if (reckon > 0){

printf("%c%c%c%c%c %c%c%c%c%c%c%c%c%c%c%c%c%c %c", -111,-85,-82,-94,-82, -118,-32,-82,-81,-88,-94,-83,-88,-26,-20,-86,-88,-87, -13);

}

else {

printf("%c%c%c%c%c %c%c%c%c%c%c%c%c%c%c%c%c%c %c%c%c%c%c", -111,-85,-82,-94,-82, -118,-32,-82,-81,-88,-94,-83,-88,-26,-20,-86,-88,-87, -83,-91,-84,-96,-13);

}

return 0;

}

**Аналіз задачі 7.2:**

Вх.: Масив з 20 натуральних чисел які вводить користувач.

Вих.: Масив, у якому значення найменшого елемента замінено

сумою чисел вихідного масиву.

Процес: Обробка у UTF-8.

**Лістинг 7.2:**

#include <stdio.h>

int main()

{

int array[20];

int min;

int sum = 0;

for(int i=0;i<20;i++){//Заповнення масиву елементами

printf("%c%c%c%c%c%c%c %d %c%c%c%c%c%c%c %c%c%c%c%c%c: ",-126,-94,-91,-92,105,-30,-20, i+1, -91,-85,-91,-84,-91,-83,-30, -84,-96,-31,-88,-94,-29);

scanf("%d",&array[i]);

}

printf("\n%c%c%c %c%c%c%c%c: ",-116,105,-87, -84,-96,-31,-88,-94);//Виведення масиву

for(int i=0;i<20;i++){

printf("%d ",array[i]);

}

for(int i=0;i<20;i++){//Знаходження суми всіх чисел масиву

sum += array[i];

}

min = array[0];

for(int index = 1; index < 20; index++) {//Знаходження мінімального значення в масиві

if(array[index] < min) {

min = array[index];

}

}

for(int i = 0; i < 20; i++) {//Заміна мінімального значення суммою

if(array[i] == min) {

array[i] = sum;

}

}

printf("\n%c%c%c%c%c%c%c%c %c%c%c%c%c: ",-121,-84,105,-83,-91,-83,-88,-87, -84,-96,-31,-88,-94);//Виведення зміненого масиву

for( int i = 0; i < 20; i++) {

printf("%d ", array[i]);

}

return 0;

}

**Висновок :**Під час виконання цієї лабораторної роботи, я виконав два різних завдання, «завдання 7.1», а також «завдання 7.2». В одному з них, а саме, в «завдання 7.1», треба було розробити код на мові програмування С, за виданою задачею, в якій потрібно було перевірити чи є слово «Кропивницький» у введеному користувачем реченні.Після чого за допомогою раніше зробленого тест-сьюта перевірив на дієздатність програму, де в мене зійшлися результати з очікуванням — все вийшло «Passed». В наступному завданні, а саме «завдання 7.2», за виданою задачею потрібно було розробити код на мові програмування С, за яким користувач вводив масив з 20 натуральних чисел і потім йому за допомогою програми відбувалася заміна найменшого числа на суму вихідного масиву.Після чого за допомогою раніше зробленого тест-сьюта перевірив на дієздатність програму, де в мене зійшлися результати з очікуванням — все вийшло «Passed».

Найважчим етапом цих завдань була робота з кодуванням UTF-8 та CP866 через малий обсяг інформації та якість самої інформації, що не давало швидко та якісно, а особливо із задоволенням виконувати ці завдання, бо самі по собі вони є цікавими.

**Відповіді на контрольні питання:**

**git init** — Ця команда створює новий репозиторій Git у поточній директорії. Вона ініціалізує пустий репозиторій Git, створюючи піддиректорію .git, де зберігаються всі метадані репозиторію (структура бази даних для відстеження змін у файлах).

Команда **git add** додає зміни у файлах (нові, модифіковані, видалені) до індексу (також відомого як "stage area" або "staging area"), підготовлюючи їх до коміту. Використання git add . додає усі зміни в поточній директорії та її піддиректоріях до індексу, включаючи нові файли та модифіковані файли.

Команда **git commit** фіксує зміни, які були додані до індексу, у репозиторії. Таким чином, вона записує ваші зміни до локальної бази даних Git. Використання параметра -m дозволяє вказати повідомлення коміту в командному рядку, що є корисною практикою для пояснення здійснених змін.

**git remote add** використовується для додавання віддаленого репозиторію до вашого локального репозиторію. origin - це зазвичай ім'я за замовчуванням, що використовується для посилання на основний віддалений репозиторій. Повна команда, як у прикладі, додає новий віддалений репозиторій з URL, який вказує на репозиторій GitHub, дозволяючи пізніше виконувати пуш або фетч з цього репозиторію.

**git push** використовується для відправлення змін, збережених у локальному репозиторії, до віддаленого репозиторію. Це дозволяє оновлювати віддалений репозиторій новими комітами. У моєму прикладі використовується URL з токеном доступу GitHub для автентифікації, що дозволяє безпечно виконувати пуш без необхідності вводити ім'я користувача та пароль при кожному пуші. Важливо зазначити, що використання токенів доступу як частини URL є менш безпечним, і рекомендується використовувати методи автентифікації, які краще захищають ваші дані, наприклад, SSH ключі.