Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 10

з навчальної дисципліни “Базові методології та технології програмування”

РЕАЛІЗАЦІЯ ПРОГРАМНИХ МОДУЛІВ ОБРОБЛЕННЯ ДАНИХ

СКЛАДОВИХ ТИПІВ З ФАЙЛОВИМ ВВЕДЕННЯМ/ВИВЕДЕННЯМ

ВИКОНАВ

студент академічної групи \_\_\_\_\_

\_\_\_\_\_\_\_\_\_ Барамба А.А

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

\_\_\_\_\_\_\_\_\_ Павло УСІК

Кропивницький – 2022

**ЛАБОРАТОРНА РОБОТА №10**

**ТЕМА:** Реалізація програмних модулів оброблення даних складових типів з файловим введенням/виведенням.

**МЕТА:** полягає у набутті ґрунтовних вмінь і практичних навичок реалізації у Code::Blocks IDE мовою програмування С++ програмних модулів створення й оброблення даних типів масив, структура, об’єднання, множина, перелік, перетворення типів даних, використання файлових потоків та функцій стандартних бібліотек для оброблення символьної інформації.

**ВАРІАНТ 5**

**ЗАВДАННЯ:**

1. Реалізувати програмні модулі розв’язування задач 10.1–10.3

як складові статичної бібліотеки libModulesПрізвище.а (проект

ModulesПрізвище лабораторних робіт №8–9).

2. Реалізувати тестовий драйвер автоматизованої перевірки

програмних модулів розв’язування задач 10.1–10.3.

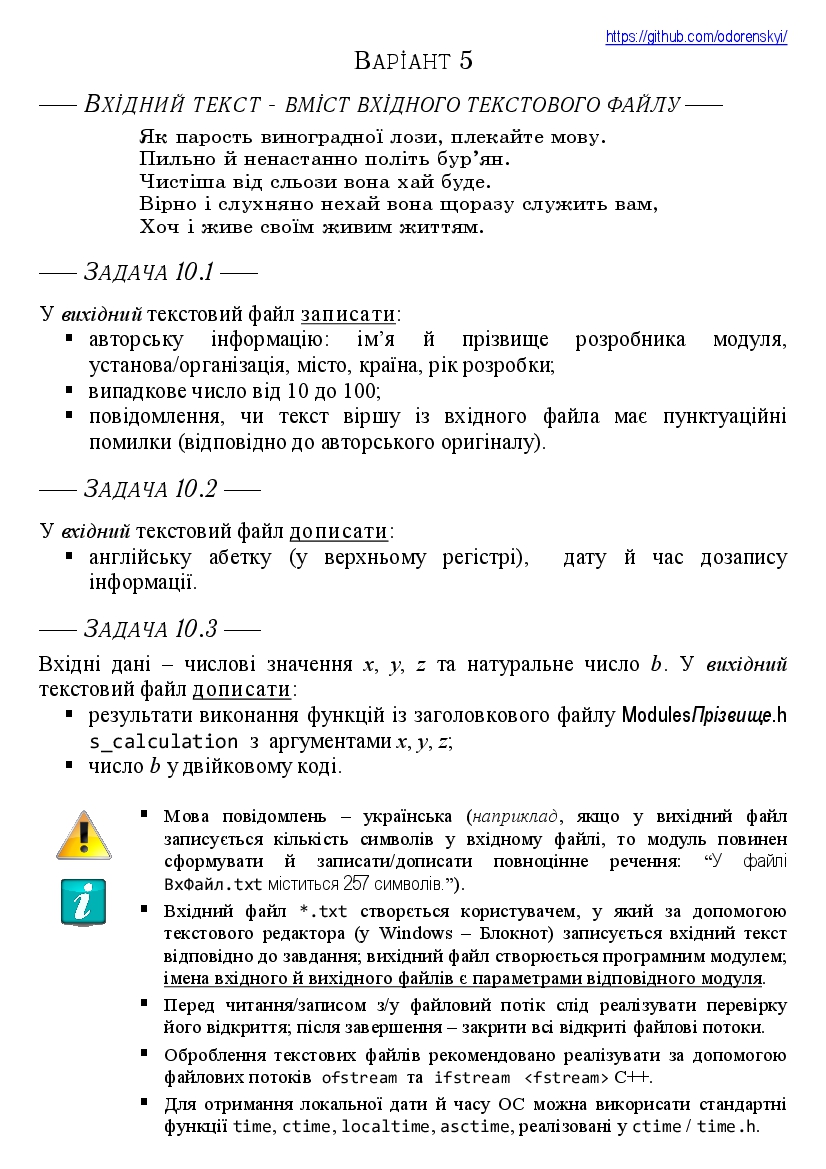
**ХІД РОБОТИ**

Рисунок 1 – Умова задачі 10.1

**Строга постановка задачі:**

* Вхідні дані: слово в форматі string записане до вхідного файлу
* Вихідні дані: записані до вихідного файлу авторська інформація, випадкове число від 10 до 100 та повідомлення, чи текст віршу із вхідного файла має пунктуаційні помилки (відповідно до авторського оригіналу)

**Проектування архітектури програмного модуля:**

authorInformation() - приймає в якості аргументу назву вихідного файлу. Виконується очищення вмісту файлу та запис авторської інформації.

randNum() - приймає в якості аргументу назву вихідного файлу. Виконується дозапис випадкового числа в діапазоні від 10 до 100 за допомогою функцій rand() та srand().

punctuation() - приймає в якості аргументу назву вихідного файлу. Для подальшої роботи створено три допоміжні рядки (signs, origPunct, inputPunct) та окремий текстовий файл із оригінальним віршом (M\_Rylsky\_poem.txt). У signs записано знаки пунктуації, origPunct та inputPunct порожні. Через вкладені цикли реалізовано запис знаків пунктуації до origPunct з M\_Rylsky\_poem.txt та до inputPunct з inputText.txt (у якому міститься вірш із завдання). Далі булевій змінній result привласнюється результат порівняння inputPunct із origPunct. У випадку якщо result = 0 (хиба, рядки не однакові) – у файл записується повідомлення про наявність пунктуаційних помилок, інакше (1, істина, рядки однакові) – записується повідомлення про відсутність пунктуаційних помилок.

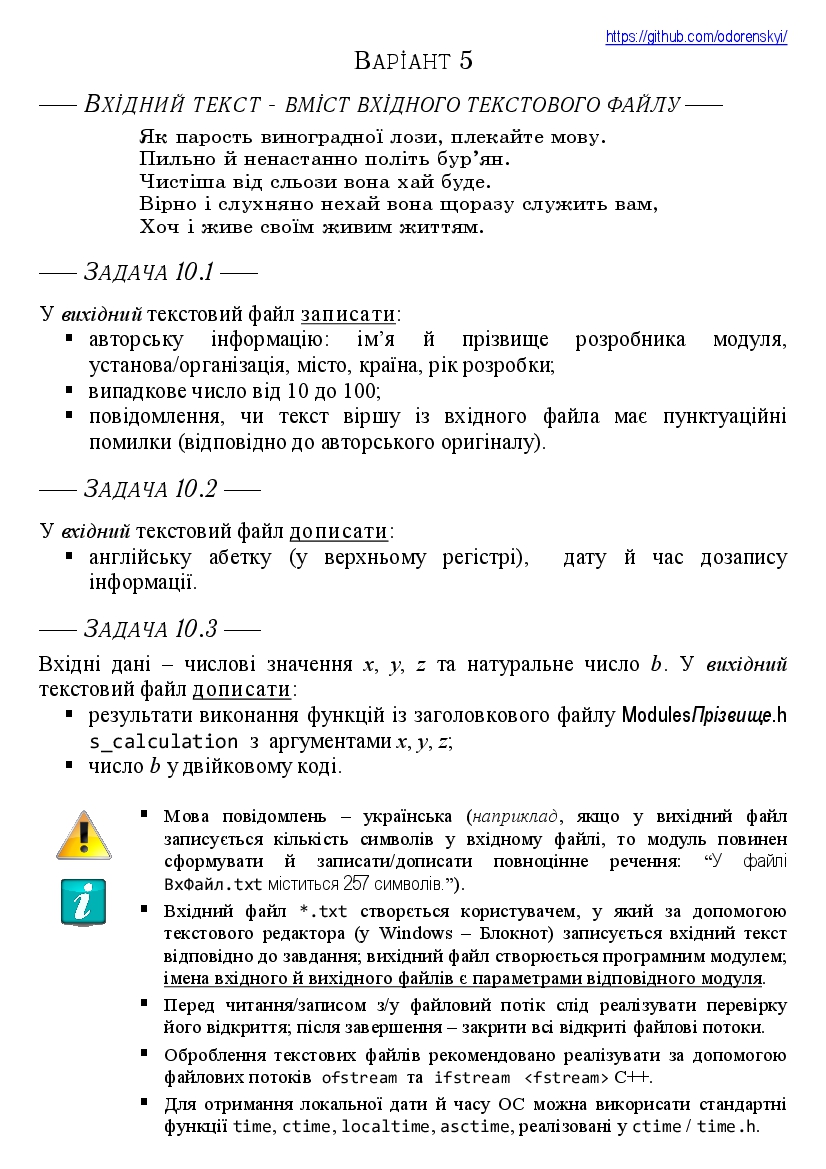


Рисунок 2 – Умова задачі 10.2

**Строга постановка задачі:**

* Вхідні дані: слово в форматі string записане до вхідного файлу
* Вихідні дані: дописана у вхідний файл англійська абетка, дата й час дозапису інформації.

**Проектування архітектури програмного модуля:**

* alphabet() - приймає в якості аргументу назву вхідного файлу. Створено допоміжну змінну letter яка зберігає значення літери “А”. Реалізовано цикл, у якому відбувається запис літер у файл шляхом збільшення letter на 1. Також для форматування було створено допоміжну змінну j = 0, яка на кожній ітерації збільшується на 1. У випадку якщо j = 4 – відбувається перехід на новий рядок, а змінна обнулюється. У випадку якщо j = 23 – додається два пробіли перед наступною літерою.
* dateTime() - приймає в якості аргументу назву вхідного файлу. В даний файл записується дата та час дозапису інформації за допомогою стандартної функції з бібліотеки <ctime>.

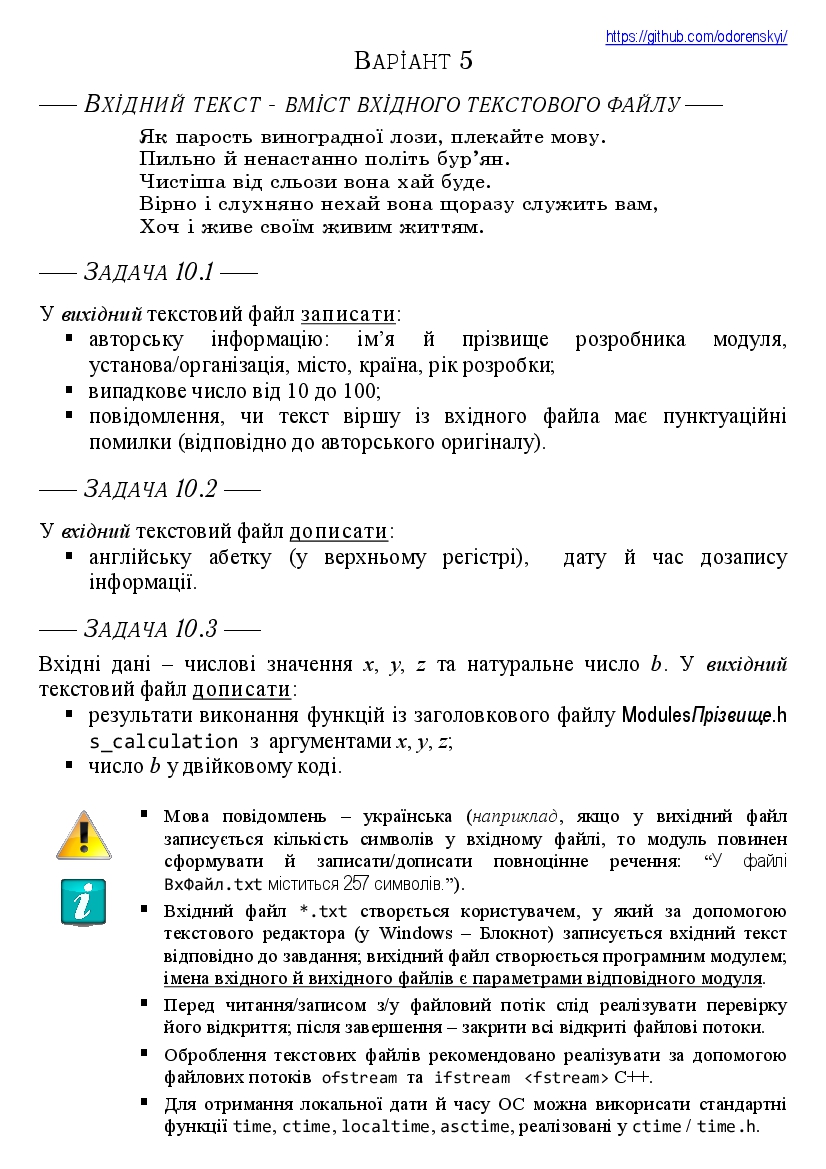


Рисунок 3 – Умова задачі 10.3

**Строга постановка задачі:**

* Вихідні та вхідні дані вказані в умові задачі.

**Проектування архітектури програмного модуля:**

* binNumber() – приймає в якості аргументу число та назву вихідного файлу. В даний файл записується число у двійковому коді за допомогою стандартної функції з бібліотеки <bitset>.

**Результат тестування за допомогою TestDriver.exe:**

=======================================================================================================================

TASK 10.1 TEST CASE #1

ЗАПИС авторської інформації

Incoming data: Шлях до файлу outputText.txt

Expected result:

Модуль розробив Барамба Андрій

Центральноукраїнський Національний Технічний Університет

Кропивницький, Україна, 2022

Actual result:

Модуль розробив Барамба Андрій

Центральноукраїнський Національний Технічний Університет

Кропивницький, Україна, 2022

Test case #1 PASSED

=======================================================================================================================

=======================================================================================================================

TASK 10.1 TEST CASE #2

ЗАПИС випадкового числа від 10 до 100

Incoming data: Шлях до файлу outputText.txt

Expected result: Випадкове число від 10 до 100

Actual result: 24

Чи "actual result" є числом від 10 до 100? (Так - Y, Ні - N)

TEST CASE #2 PASSSED

=======================================================================================================================

=======================================================================================================================

TASK 10.1 TEST CASE #3

ЗАПИС повідомлення, чи текст віршу з вхідного файла має пунктуаційні помилки

(відповідно до авторського оригіналу)

Incoming data: Шлях до файлу outputText.txt

Expected result:

Відповідно до авторського оригіналу, у вхідному файлі пунктуаційних помилок немає

Actual result:

Відповідно до авторського оригіналу, у вхідному файлі пунктуаційні помилки наявні

TEST CASE #3 FAILED

=======================================================================================================================

=======================================================================================================================

TASK 10.2 TEST CASE #1

ДОЗАПИС англійської абетки (у верхньому регістрі)

Incoming data: Шлях до файлу inputText.txt

Expected result:

Як парость виноградної лози, плекайте мову.

Пильно й нестанно політь бур’ян.

Чистіша від сльози вона хай буде.

Вірно й слухняно нехай вона щоразу служить вам,

Хоч і живе своїм життям.

A B C D

E F G H

I J K L

M N O P

Q R S T

U V W X

Y Z

Actual result:

Як парость виноградної лози, плекайте мову.

Пильно й нестанно політь бур’ян.

Чистіша від сльози вона хай буде.

Вірно й слухняно нехай вона щоразу служить вам,

Хоч і живе своїм життям.

A B C D

E F G H

I J K L

M N O P

Q R S T

U V W X

Y Z

Test case #1 PASSED

=======================================================================================================================

=======================================================================================================================

TASK 10.2 TEST CASE #2

Дата й час дозапису інформації

Incoming data: Шлях до файлу inputText.txt

Expected result: Дата й час дозапису інформації

Actual result:

Як парость виноградної лози, плекайте мову.

Пильно й нестанно політь бур’ян.

Чистіша від сльози вона хай буде.

Вірно й слухняно нехай вона щоразу служить вам,

Хоч і живе своїм життям.

A B C D

E F G H

I J K L

M N O P

Q R S T

U V W X

Y Z

Sun Apr 10 13:54:43 2022

Чи "actual result" співпадають з поточним часом (можлива похибка: одна хвилина)? (Так - Y, Ні - N)

TEST CASE #2 PASSSED

=======================================================================================================================

=======================================================================================================================

TASK 10.3 TEST CASE #1

ДОЗАПИС результату функції s\_calculation

Incoming data: Шлях до файлу outputText.txt

Expected result:

102.261

Actual result:

102.261

Test case #1 PASSED

Результат s\_calculation записано в файл

Вміст файлу:

102.262

=======================================================================================================================

=======================================================================================================================

TASK 10.3 TEST CASE #3.1

ДОЗАПИС числа у двійковому коді

Incoming data: Шлях до файлу outputText.txt, число 432

Expected result:

0000000110110000

Actual result:

0000000110110000

Test case #3.1 PASSED

=======================================================================================================================

=======================================================================================================================

TASK 10.3 TEST CASE #3.2

ДОЗАПИС числа у двійковому коді

Incoming data: Шлях до файлу outputText.txt, число 543

Expected result:

0000001000011111

Actual result:

0000001000011111

Test case #3.2 PASSED

=======================================================================================================================

=======================================================================================================================

TASK 10.3 TEST CASE #3.3

ДОЗАПИС числа у двійковому коді

Incoming data: Шлях до файлу outputText.txt, число 1023

Expected result:

0000001111111111

Actual result:

0000001111111111

Test case #3.3 PASSED

=======================================================================================================================

=======================================================================================================================

Усі три тест кейса пройшли перевірку! Останнє число записано у файл у двійковому коді1

Вміст файлу:

Модуль розробив Барамба Андрій

Центральноукраїнський Національний Технічний Університет

Кропивницький, Україна, 2022

51

Відповідно до авторського оригіналу, у вхідному файлі пунктуаційні помилки наявні

102.262

0000001111111111

=======================================================================================================================

Для завершення натисніть Enter...

**ВИСНОВКИ**

Проаналізувавши завдання лабораторної роботи та користуючись поняттям модульності програмного забезпечення було прийнято рішення кожне завдання із задач 10.1, 10.2 та 10.3 реалізувати за допомогою підфункцій, що сконцентровані на вирішенні конкретної, логічної частини завдання. Метою розбиття функцій є досягнення більш детального декларування результатів тестування окремих частин програми.

Для коректного виведення результатів виконання функцій зі статичної бібліотеки libModulesBaramba.a у консольне вікно застосунку TestDriver.exe, функції було ініціалізовано типом bool. Функції, з результатом повернення bool визначають, чи є, насамперед, істинним виконання даної функції, чи ні.

Враховано коректне відкриття та закриття файлових потоків для всіх функцій. Так, на початку функціонування кожної процедури оголошується змінна типу, відповідного до мети взамодії файлових потоків (ofstream - для запису / ifstream - для читання). Додатково, в аргументах даних змінних, окрім вказання назви файлу, вказано тип відкриття файлу (ios::out - для запису / ios::app - для дозапису). В режимі ios::out працюють лише ті процедури, що мають виконати перезапис всього файлу (тобто функції, що вперше взаємодіють з файлом).

Якщо файл відсутній і до нього спробуватищось записати, зазначивши при відкритті назву цього файлу – створиться новий файл із вказаною назвою. З цієї причини перевірку на відкриття було реалізовано лише у функції redfile(), оскільки тільки при зчитуванні буде неможливо відкрити неіснуючий файл.

Оперуючи вимогою щодо локалізації програмного забезпечення українською мовою, усі операції де потрібно вносити дані з файлу у ПЗ та/або їх порівнювати було реалізовано за допомогою змінних типу string.

Запис авторської інформації відбуваєтьсяза допомогою звичайного запису у файл через оператор <<.

Запис випадкового числа відбувається за допомогою двох функцій: rand() та srand(). Оскільки функція rand() повертає псевдовипадкові числа (при багаторазовому запуску програми із функцією rand() програма буде виводити однакові числа), була використана функція srand(), яка встановлює вихідне число для послідовності, що генерується функцією rand(). Але якщо аргумент в srand() не змінювати, то все одно будуть виводитися однакові числа. Виходячи з цього, було прийнято рішення в аргумент srand() передавати функцію time(0) із стандартної бібліотеки <ctime>. Ця функція повертає кількість секунд, які пройшли з 00:00:00 1 січня 1970, тобто число постійно різне, а отже результат rand() постійно різний.

Для задання граничного значення вибору числа після функції rand() потрібно дописати % \*граничне значення\*. Таким чином, функція буде обирати числа в діапазоні від 0 до заданого граничного значення (не включаючи його). Щоб функція обирала число починаючи від іншого значення, потрібно написати \*початкове значення\* + rand(). Таким чином функція буде обирати починаючи від заданого значення. Але якщо записано початкове і граничне значення, то початкове збільшує граничне (наприклад, якщо написано 5 + rand() % 10, то буде обиратися число від 5 до 15). З цієї причини було записано 10 + rand() % 91. Таким чином, функція буде обирати число від 10 до 100 включно.

Функція readfile() зчитує переданий файл, попередньо перевіряючи, чи він відкривається.

Перевірка пунктуаційних помилок у функції punctuation() реалізована наступним чином. Створено рядок signs, який містить у собі знаки пунктуації та два допоміжні рядки inputPunct та origPunct, у які будуть зберігатися знаки пунктуації із вхідного файла та файла з оригінальним віршом відповідно. Через вкладені цикли реалізовано внесення знаків пунктуації до inputPunct та origPunct. Далі вони порівнюються (за допомогою функції switch), і у випадку рівності до вихідного файлу записується повідомлення про відсутність пунктуаційних помилок, інакше – про наявність.

Дозапис англійської абетки за допомогою функції alphabet() у вхідний файл було реалізовано наступним чином. Оскільки запис “ \*літера\* + 1 ” буде дорівнювати наступній літері запис алфавіту було реалізовано через цикл. Також, для форматування було прийняте рішення за допомогою додаткової змінної через кожні 4 літери переносити абетку на новий рядок, а останні дві літери написати посередині. Дата та час дозапису інформації було реалізовано через стандартні функції бібліотеки <ctime>.

Запис числа у двійковому коді було реалізовано за допомогою стандартної функції bitset бібліотеки <bitset>. У функції вказується кількість бітів, а за аргумент приймається число.

Було прийняте рішення не створювати окрему функцію для запису результату функції s\_calculation, а записати у файл під час тестування.

Під час тестування використовувалось автоматизоване тестування, але у двох випадках було використане напівавтоматизоване. Перший випадок – запис випадкового числа. У цьому випадку неможливо точно протестувати, яке число запишеться у файл, а перевірка на те, чи є записані символи числом від 10 до 100 ускладнить код та зробить його громіздким. Тому було прийняте рішення використати конструкцію switch. Тестувальнику виводиться запит, чи є дане число числом від 10 до 100. Якщо тестувальник вводить Y, тест кейс вважається виконаним, якщо N – тест кейс не виконано. При спробі ввести інші значення програма видає звуковий сигнал. Аналогічно було зроблено із перевіркою правильності запису часу, але у запиті була введена похибка на одну хвилину. Це було зроблено через те, що при запуску програми може бути 59 секунд, а коли тестувальник подивиться на годинник – буде наступна хвилина, тому було прийняте рішення ввести похибку.

**Додаток А**

**(TestSuite до завдання 10.1)**

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TS\_10\_1 |
| Date: 09/04/2022 | Назва проекта/ПЗ /  **Name of project** | TestDriver.exe |
|  | Рівень тестування /  **Level of testing** | Модульний |
|  | Автор тест-сьюта /  **Test Suite Author** | Барамба Андрій |
|  | Виконавець /  **Implementer** | Барамба Андрій |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Preliminary Steps** | **Action (test steps)** | **Expected Result** | **Test Result** |
| TC-01 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за запис авторської інформації | Вміст файлу:  Модуль розробив Барамба Андрій  Центральноукраїнський Національний Технічний Університет  Кропивницький, Україна, 2022 | Passed |
| TC-02 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за запис випадкового числа від 10 до 100 | Вміст файлу:  Повинно міститися випадкове число від 10 до 100 | Passed |
| TC-03 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за запис повідомлення про наявність/відсутність пунктуаційних помилок у вірші із *вхідного* файлу (порівняно з оригіналом) | Вміст файлу:  Відповідно до авторського оригіналу, у вхідному файлі пунктуаційних помилок немає | Passed |

**Додаток Б**

**(TestSuite до завдання 10.2)**

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TS\_10\_2 |
| Date: 09/04/2022 | Назва проекта/ПЗ /  **Name of project** | TestDriver.exe |
|  | Рівень тестування /  **Level of testing** | Модульний |
|  | Автор тест-сьюта /  **Test Suite Author** | Барамба Андрій |
|  | Виконавець /  **Implementer** | Барамба Андрій |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Preliminary Steps** | **Action (test steps)** | **Expected Result** | **Test Result** |
| TC-01 | Вхідний файл | Виклик функції, що відповідає за дозапис англійської абетки (у верхньому регістрі) | Вміст файлу:  Як парость виноградної лози, плекайте мову.  Пильно й нестанно політь бур’ян.  Чистіша від сльози вона хай буде.  Вірно й слухняно нехай вона щоразу служить вам,  Хоч і живе своїм життям.  A B C D  E F G H  I J K L  M N O P  Q R S T  U V W X  Y Z | Passed |
| TC-02 | Вихідний файл | Виклик функції, що відповідає за дозапис дати й часу дозапису інформації | Вміст файлу:  Як парость виноградної лози, плекайте мову.  Пильно й нестанно політь бур’ян.  Чистіша від сльози вона хай буде.  Вірно й слухняно нехай вона щоразу служить вам,  Хоч і живе своїм життям.  A B C D  E F G H  I J K L  M N O P  Q R S T  U V W X  Y Z  \*дата й час дозапису\* | Passed |

**Додаток В**

**(TestSuite до завдання 10.3)**

|  |  |  |
| --- | --- | --- |
| Artifact: Test Suite | Назва тестового набору /  **Test Suite Description** | TS\_10\_3 |
| Date: 09/04/2022 | Назва проекта/ПЗ /  **Name of project** | TestDriver.exe |
|  | Рівень тестування /  **Level of testing** | Модульний |
|  | Автор тест-сьюта /  **Test Suite Author** | Барамба Андрій |
|  | Виконавець /  **Implementer** | Барамба Андрій |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Preliminary Steps** | **Action (test steps)** | **Expected Result** | **Test Result** |
| TC-01 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за дозапис результату виконання функції s\_calculation | Вміст файлу:  Модуль розробив Барамба Андрій  Центральноукраїнський Національний Технічний Університет  Кропивницький, Україна, 2022  98  Відповідно до авторського оригіналу, у вхідному файлі пунктуаційні помилки наявні  102.262 | Passed |
| TC-02 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за дозапис числа *b* у двійковому коді  *b* =432 | Вміст файлу:  0000000110110000 | Passed |
| TC-03 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за дозапис числа *b* у двійковому коді  *b* =543 | Вміст файлу:  0000001000011111 | Passed |
| TC-04 | Вихідний файл: outputText.txt  Вміст: порожній | Виклик функції, що відповідає за дозапис числа *b* у двійковому коді  *b* =1023 | Вміст файлу:  Модуль розробив Барамба Андрій  Центральноукраїнський Національний Технічний Університет  Кропивницький, Україна, 2022  98  Відповідно до авторського оригіналу, у вхідному файлі пунктуаційні помилки наявні  102.262  0000001111111111 | Passed |

**Додаток Г**

(Вихідний код ModulesBaramba)

**… / ModulesBaramba / main.cpp**

#include <iostream>

#include <fstream>

#include <ctime>

#include <algorithm>

#include <bitset>

#include "cmath"

using namespace std;

double s\_calculation(float x, float z)

{

const double pi = 3.14159;

return (sqrt(1 + x) - 3 \* cos(x)) / (log(pow(x, 2)) + 3 \* sin(pi \* x)) + pow(sqrt(pow(z, 3)) + 2, 2);

}

string BeaufortScale(int BeaufortScore) {

switch (BeaufortScore){

case 0:

return "Wind speed less than 0.0003 km/s";

case 1:

return "Wind speed varies from 0.0003 to 0.0015 km/s";

case 2:

return "Wind speed varies from 0.0016 to 0,0094 km/s";

case 3:

return "Wind speed varies from 0.0094 to 0,0054 km/s";

case 4:

return "Wind speed varies from 0.0055 to 0,0079 km/s";

case 5:

return "Wind speed varies from 0.008 to 0,0107 km/s";

case 6:

return "Wind speed varies from 0.0108 to 0,0138 km/s";

case 7:

return "Wind speed varies from 0.0139 to 0,0171 km/s";

case 8:

return "Wind speed varies from 0.0172 to 0,0207 km/s";

case 9:

return "Wind speed varies from 0.0208 to 0,0224 km/s";

case 10:

return "Wind speed varies from 0.0245 to 0,0284 km/s";

case 11:

return "Wind speed varies from 0.0285 to 0,0326 km/s";

case 12:

return "Wind speed more than 0.0327 km/s";

default:

return "Invalid value";

}

}

string helmetSize(int headSize){

if(headSize < 475 || headSize > 624){

return "Your size is not in the table";

}

headSize = round(headSize / 10);

int minimalSize = 47;

char helmetSizes[8][255] = {"Children's helmet, size S",

"Children's helmet, size M",

"Children's helmet, size L",

"Adult's helmet, size XS",

"Adult's helmet, size S",

"Adult's helmet, size M",

"Adult's helmet, size L",

"Adult's helmet, size XL",

};

int i = 0;

for(; minimalSize < 63; minimalSize++){

if(minimalSize != headSize){

if(minimalSize % 2 == 0) i++;

continue;

} else {

return helmetSizes[i];

}

}

}

string binOnesZeros(int number){

int binZeros = 0;

int iter\_num = ~number;

while(iter\_num){

iter\_num &= iter\_num - 1;

binZeros ++;

}

if(number < 0 || number > 8999){

return "Wrong number! Your number: " + to\_string(number);

}

switch(number >> 9 & 1){

case 0:

return "Number of binary 0 = " + to\_string(binZeros);

case 1:

return "Number of binary 1 = " + to\_string(32 - binZeros);

}

}

bool authorInformation(string path)

{

ofstream fout;

fout.open(path);

fout << "Модуль розробив Барамба Андрій\n" <<

"Центральноукраїнський Національний Технічний Університет\n" <<

"Кропивницький, Україна, 2022\n";

fout.close();

return true;

}

bool randNum(string path)

{

ofstream fout;

srand(time(0));

fout.open(path, ios\_base::app);

fout << 10 + rand() % 91;

fout.close();

return true;

}

string readfile(string path)

{

ifstream file;

string text = "";

string line;

file.open(path);

if(!file.is\_open()) { return path + " is not open"; };

while(getline(file, line)){

text += line + '\n';

}

file.close();

return text;

}

bool punctuation(string path)

{

string origPoem = readfile("M\_Rylsky\_poem.txt");

string inputPoem = readfile("inputText.txt");

string signs = "!?,.:;-";

string origPunct;

string inputPunct;

for(int i = 0; i < origPoem.length(); i++){

for(int j = 0; j < signs.length(); j++){

if(origPoem[i] == signs[j]){

origPunct.append(1, origPoem[i]);

break;

}

}

}

cout << endl << endl;

for(int i = 0; i < inputPoem.length(); i++){

for(int j = 0; j < signs.length(); j++){

if(inputPoem[i] == signs[j]){

inputPunct.append(1, inputPoem[i]);

break;

}

}

}

bool result = (inputPunct == origPunct);

ofstream fout;

fout.open(path, ios\_base::app);

switch(result){

case 0:

fout << "\nВідповідно до авторського оригіналу, у вхідному файлі пунктуаційні помилки наявні\n";

fout.close();

case 1:

fout << "\nВідповідно до авторського оригіналу, у вхідному файлі пунктуаційних помилок немає\n";

fout.close();

}

return true;

}

bool alphabet(string path)

{

ofstream fout;

fout.open(path, ios\_base::app);

char letter = 'A';

int j = 0;

fout << "\n";

for(int i = 0; i < 26; i++){

fout << letter << " ";

letter++;

j++;

if(j == 4){ fout << "\n"; j = 0; };

if(i == 23) { fout << " "; };

}

fout << "\n";

fout.close();

return true;

}

bool dateTime(string path)

{

ofstream fout;

fout.open(path, ios\_base::app);

time\_t curtime;

time(&curtime);

fout << "\n\n" << ctime(&curtime);

fout.close();

return true;

}

bool binNumber(int number, string path)

{

ofstream fout;

fout.open(path, ios\_base::app);

fout << "\n" << bitset<16> (number) << "\n";

fout.close();

return true;

}

**… / prj / ModulesBaramba.h**

#ifndef MODULESBARAMBA\_H\_INCLUDED

#define MODULESBARAMBA\_H\_INCLUDED

#include <iostream>

#include <cmath>

double s\_calculation(float x, float z);

std::string BeaufortScale(int BeaufortScore);

std::string helmetSize(int headSize);

std::string binOnesZeros(int number);

bool authorInformation(std::string path);

bool randNum(std::string path);

std::string readfile(std::string path);

bool punctuation(std::string path);

bool alphabet(std::string path);

bool dateTime(std::string path);

bool binNumber(int number, std::string path);

#endif // MODULESBARAMBA\_H\_INCLUDED

**Додаток Д**

(вихідний код TestDriver)

#include <iostream>

#include <cstring>

#include <conio.h>

#include <fstream>

#include "ModulesBaramba.h"

using namespace std;

int main()

{

system("chcp 65001 & cls");

fstream file;

string outputPath = "outputText.txt";

string inputPath = "inputText.txt";

file.open(inputPath, ios\_base::out);

system("chcp 65001 & cls");

file << "Як парость виноградної лози, плекайте мову.\n"

<< "Пильно й нестанно політь бур’ян.\n"

<< "Чистіша від сльози вона хай буде.\n"

<< "Вірно й слухняно нехай вона щоразу служить вам,\n"

<< "Хоч і живе своїм життям.";

file.close();

string TC\_1\_1\_expectedResult = "Модуль розробив Барамба Андрій\nЦентральноукраїнський Національний Технічний Університет\nКропивницький, Україна, 2022\n";

cout << authorInformation(outputPath) << "\b";

string TC\_1\_1\_actualResult = readfile(outputPath);

int TC\_1\_1\_result = 0;

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.1 TEST CASE #1" << endl

<< "\t\t\t\t\t ЗАПИС авторської інформації\n\n"

<< "Incoming data: Шлях до файлу outputText.txt\n\n"

<< "Expected result:\n\n" << TC\_1\_1\_expectedResult << "\n"

<< "Actual result:\n\n" << TC\_1\_1\_actualResult << "\n";

if(TC\_1\_1\_actualResult == TC\_1\_1\_expectedResult){

cout << "\tTest case #1 PASSED" << endl;

TC\_1\_1\_result++;

} else cout << "\tTest case #1 FAILED" << endl;

file.open(outputPath, ios\_base::out);

file << "";

file.close();

cout << "=======================================================================================================================" << endl << endl;

cout << randNum(outputPath) << "\b";

string TC\_1\_2\_actualResult = readfile(outputPath);

int TC\_1\_2\_result = 0;

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.1 TEST CASE #2" << endl

<< "\t\t\t\t\t ЗАПИС випадкового числа від 10 до 100\n\n"

<< "Incoming data: Шлях до файлу outputText.txt\n\n"

<< "Expected result: Випадкове число від 10 до 100\n\n"

<< "Actual result: " << TC\_1\_2\_actualResult << "\n";

cout << "Чи \"actual result\" є числом від 10 до 100? (Так - Y, Ні - N)\n" << endl;

while(true){

switch(getch()){

case 'Y': cout << "\tTEST CASE #2 PASSSED" << endl; break;

case 'N': cout << "\tTEST CASE #2 FAILED" << endl; break;

default: cout << "\a"; continue;

}

break;

}

TC\_1\_2\_result++;

file.open(outputPath, ios\_base::out);

file << "";

file.close();

cout << "=======================================================================================================================" << endl << endl;

string TC\_1\_3\_expectedResult = "\nВідповідно до авторського оригіналу, у вхідному файлі пунктуаційних помилок немає\n";

cout << punctuation(outputPath) << "\b";

string TC\_1\_3\_actualResult = readfile(outputPath);

int TC\_1\_3\_result = 0;

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.1 TEST CASE #3" << endl

<< "\t\t ЗАПИС повідомлення, чи текст віршу з вхідного файла має пунктуаційні помилки\n"

<< "\t\t\t\t\t(відповідно до авторського оригіналу)\n\n"

<< "Incoming data: Шлях до файлу outputText.txt\n\n"

<< "Expected result:\n" << TC\_1\_3\_expectedResult << "\n\n"

<< "Actual result:\n" << TC\_1\_3\_actualResult << "\n";

if(TC\_1\_3\_actualResult == TC\_1\_3\_expectedResult){

cout << "\tTEST CASE #3 PASSED" << endl;

TC\_1\_3\_result++;

} else cout << "\tTEST CASE #3 FAILED" << endl;

file.open(outputPath, ios\_base::out);

file << "";

file.close();

cout << "=======================================================================================================================" << endl << endl;

if(TC\_1\_1\_result == TC\_1\_2\_result == TC\_1\_3\_result == 1){

cout << authorInformation(outputPath) << "\b"

<< randNum(outputPath) << "\b"

<< punctuation(outputPath) << "\b";

}

string TC\_2\_1\_expectedResult = "Як парость виноградної лози, плекайте мову.\n"

"Пильно й нестанно політь бур’ян.\n"

"Чистіша від сльози вона хай буде.\n"

"Вірно й слухняно нехай вона щоразу служить вам,\n"

"Хоч і живе своїм життям."

"\nA B C D \nE F G H \nI J K L \nM N O P \nQ R S T \nU V W X \n Y Z \n";

cout << alphabet(inputPath) << "\b";

string TC\_2\_1\_actualResult = readfile(inputPath);

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.2 TEST CASE #1" << endl

<< "\t\t\t\t ДОЗАПИС англійської абетки (у верхньому регістрі)\n\n"

<< "Incoming data: Шлях до файлу inputText.txt\n\n"

<< "Expected result:\n\n" << TC\_2\_1\_expectedResult << "\n"

<< "Actual result:\n\n" << TC\_2\_1\_actualResult << "\n";

if(TC\_2\_1\_actualResult == TC\_2\_1\_expectedResult) cout << "\tTest case #1 PASSED" << endl;

else cout << "\tTest case #1 FAILED" << endl;

cout << "=======================================================================================================================" << endl << endl;

cout << dateTime(inputPath) << "\b";

string TC\_2\_2\_actualResult = readfile(inputPath);

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.2 TEST CASE #2" << endl

<< "\t\t\t\t\t Дата й час дозапису інформації\n\n"

<< "Incoming data: Шлях до файлу inputText.txt\n\n"

<< "Expected result: Дата й час дозапису інформації\n\n"

<< "Actual result: \n\n" << TC\_2\_2\_actualResult << "\n";

cout << "Чи \"actual result\" співпадають з поточним часом (можлива похибка: одна хвилина)? (Так - Y, Ні - N)\n" << endl;

while(true){

switch(getch()){

case 'Y': cout << "\tTEST CASE #2 PASSSED" << endl; break;

case 'N': cout << "\tTEST CASE #2 FAILED" << endl; break;

default: cout << "\a"; continue;

}

break;

}

cout << "=======================================================================================================================" << endl << endl;

double TC\_3\_1\_expectedResult = 102.261;

double TC\_3\_1\_actualResult = floor(s\_calculation(3.0, 4.0) \* 1000) / 1000;

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.3 TEST CASE #1" << endl

<< "\t\t\t\t ДОЗАПИС результату функції s\_calculation\n\n"

<< "Incoming data: Шлях до файлу outputText.txt\n\n"

<< "Expected result:\n\n" << TC\_3\_1\_expectedResult << "\n\n"

<< "Actual result:\n\n" << TC\_3\_1\_actualResult << "\n\n";

if(TC\_3\_1\_actualResult == TC\_3\_1\_expectedResult){

cout << "\tTest case #1 PASSED" << endl

<< "\n\tРезультат s\_calculation записано в файл\n\n";

file.open(outputPath, ios\_base::app);

file << s\_calculation(3.0, 4.0);

file.close();

} else cout << "\tTest case #1 FAILED" << endl;

string outoutFileContents = readfile(outputPath);

cout << "Вміст файлу:" << endl << endl << outoutFileContents;

cout << "=======================================================================================================================" << endl << endl;

file.open(outputPath, ios\_base::out);

file << "";

file.close();

int numbers[3] = {432, 543, 1023};

char TC\_3\_2\_expectedResult[3][255] = {"\n0000000110110000\n",

"\n0000001000011111\n",

"\n0000001111111111\n"};

unsigned short int TC\_3\_2\_result = 0;

for(int i = 0; i < 3; i++){

cout << "=======================================================================================================================" << endl

<< "\t\t\t\t\t\tTASK 10.3 TEST CASE #3." << i + 1 << endl

<< "\t\t\t\t\t ДОЗАПИС числа у двійковому коді\n\n"

<< "Incoming data: Шлях до файлу outputText.txt, число " << numbers[i] << "\n\n"

<< "Expected result:\n" << TC\_3\_2\_expectedResult[i] << "\n";

cout << binNumber(numbers[i], outputPath) << "\b";

string TC\_3\_2\_actualResult = readfile(outputPath);

cout << "Actual result:\n" << TC\_3\_2\_actualResult << "\n\n";

if(TC\_3\_2\_actualResult == TC\_3\_2\_expectedResult[i]){

cout << "\tTest case #3." << i + 1 << " PASSED" << endl;

TC\_3\_2\_result++;

} else cout << "\tTest case #3." << i + 1 << " FAILED" << endl;

file.open(outputPath, ios\_base::out);

file << "";

file.close();

cout << "=======================================================================================================================" << endl << endl;

}

cout << "=======================================================================================================================" << endl;

if(TC\_3\_2\_result == 3){

cout << "\nУсі три тест кейса пройшли перевірку! Останнє число записано у файл у двійковому коді";

cout << authorInformation(outputPath) << "\b"

<< randNum(outputPath) << "\b"

<< punctuation(outputPath) << "\b";

file.open(outputPath, ios\_base::app);

file << s\_calculation(3.0, 4.0);

file.close();

cout << binNumber(1023, outputPath) << "\b";

}

outoutFileContents = readfile(outputPath);

cout << "Вміст файлу:" << endl << endl << outoutFileContents << endl;

cout << "=======================================================================================================================" << endl << endl;

cout << "Для завершення натисніть Enter..." << endl;

cin.get();

return 0;

}