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 скводких типів з файлами введення/виведення”

Виконав
Студент академічної групи КН-22
Осадчий В.К

Перевірив
Викладач кафедри кібербезпеки
Та програмного забезпечення
О.СОБІНОВ

Кропивницький-2023

Мета роботи полягає у набутті ґрунтовних вмінь і практичних навичок реалізації у Code::Blocks IDE мовою програмування C++ 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.

https://github.com/odorenskyi/

Варіант 20

— Вхідний текст - вміст вхідного текстового файлу —

Довільне речення українською мовою.

— Задача 10.1 —

У ***вихідний*** текстовий файл записати:

- авторську інформацію: ім'я й прізвище розробника модуля, установа/організація, місто, країна, рік розробки;
- кількість знаків «:» у реченні із вхідного файла;
- речення із вхідного файлу, у якому відсутні всі голосні літери.

— Задача 10.2 —

У ***вхідний*** текстовий файл дописати:

- транслітерований текст з цього файлу відповідно до Постанови Кабінету Міністрів України "Про впорядкування транслітерації українського алфавіту латиницею" від 27 січня 2010 року № 55;
- дату дозапису інформації у форматі «ДД / ММ / РР».

— Задача 10.3 —

Вхідні дані – числові значення x, y, z та натуральне число b. У ***вихідний*** текстовий файл дописати:

- результати виконання функцій із заголовкового файлу Modules***Прізвище***.h `s_calculation` з аргументами x, y, z;
- число b у двійковому коді.

- **Мова повідомлень – українська (*наприклад*, якщо у вихідний файл записується кількість символів у вхідному файлі, то модуль повинен сформувати й записати/дописати повноцінне речення: "У файлі `ВхФайл.txt` міститься 257 символів.").**
- **Вхідний файл `*.txt` створюється користувачем, у який за допомогою текстового редактора (у Windows – Блокнот) записується вхідний текст відповідно до завдання; вихідний файл створюється програмним модулем; імена вхідного й вихідного файлів є параметрами відповідного модуля.**
- **Перед читання/записом з/у файловий потік слід реалізувати перевірку його відкриття; після завершення – закрити всі відкриті файлові потоки.**
- **Оброблення текстових файлів рекомендовано реалізувати за допомогою файлових потоків `ofstream` та `ifstream <fstream>` C++.**
- **Для отримання локальної дати й часу ОС можна використати стандартні функції `time`, `ctime`, `localtime`, `asctime`, реалізовані у `ctime` / `time.h`.**

Головні умови для алгоритмів кожної функції

Алгоритм виконання задачі 10_1

1.Оголошуємо функцію **`task10_1`**, яка буде приймати рядок **`filename`** як параметр.

2.Встановлюємо локаль для коректного відображення українських символів у вікні консолі.

3.Виконуємо перевірку, чи можна відкрити файл з назвою, яку отримали в параметрах. Якщо ні, виводимо повідомлення про помилку на консоль.

4.Якщо файл вдалося відкрити, то відкриваємо його для запису.

5.Записуємо у файл інформацію про автора та його роботу, яка може бути згенерована динамічно або передана у функцію як аргумент.

6.Закриваємо файл.

7.Якщо при відкритті файлу виникла помилка, виводимо відповідне повідомлення на консоль.

8.Функція **`task10_1`** завершує роботу.

Агоритм виконання задачі 10_1_2

1.Встановлюємо кодування символів **`uk_UA.UTF-8`** за допомогою функції **`setlocale()`**.

2.Відкриваємо файл з назвою `filename` для читання за допомогою **`ifstream`.**

3.Перевіряємо, чи вдалося відкрити файл за допомогою **`is_open()`**.

4.Якщо вдалося, то створюємо змінну **`count`**, яка буде містити кількість входжень символу `':'`.

5.Зчитуємо рядок з файлу за допомогою **`getline()`** та обчислюємо кількість входжень символу `':'` у цьому рядку за допомогою **`std::count()`**.

6.Додаємо отриману кількість входжень символу `':'` до змінної **`count`**.

7.Виводимо поточну кількість входжень на консоль.

8.Повертаємо кількість входжень символу `':'` у файлі.

9.Закриваємо файл за допомогою **`close()`**.

10.Якщо файл не вдалося відкрити, виводимо відповідне повідомлення на консоль та повертаємо 0.

Агоритм виконання задачі 10_1_3

1.Встановлюємо локаль для коректного відображення українських символів у вікні консолі
2.Відкриваємо файл за заданою назвою в режимі читання.
3.Перевіряємо, чи відкриття файлу пройшло успішно. Якщо ні, виводимо повідомлення про помилку і повертаємо -1.
4.Оголошуємо змінні line і sentence, які будуть використовуватися для читання файлу.
5.Читаємо файл по рядках.
6.Проходимо по кожному символу у рядку.
7.Якщо символ є крапкою, знаком оклику або знаком запитання, то вважаємо, що закінчилася речення.
8.Записуємо речення без голосних букв у змінну sentence_without_vowels.
9.Виводимо на екран речення без голосних букв і додавання пунктуації.
10.Очищаємо змінну sentence.
11.Якщо символ не є крапкою, знаком оклику або знаком питання, то додаємо його до sentence.
12.Закриваємо файл.
13.Повертаємо 0, щоб показати, що функція виконала свою роботу успішно.

Алгоритм виконання задачі 10_2

1.Встановлюємо локаль для коректного відображення українських символів у вікні консолі.
2.Отримуємо поточний час та дату.
3.Створюємо змінну date, яка міститиме поточну дату в форматі "день\місяць\рік".
4.Відкриваємо файл з назвою filename для читання.
5.Створюємо файл з назвою "output.txt" для запису.
6.Перевіряємо, чи вдалося відкрити обидва файли для подальшої роботи з ними.

7.Якщо файли відкриті успішно, то зчитуємо по одному рядку з файлу filename.
8.Кожен зчитаний рядок транслітеруємо за допомогою функції transliterate_text() і записуємо у файл "output.txt" з новим рядком в кінці кожного зчитаного рядка.
9.Додаємо до файлу "output.txt" рядок з поточною датою.
10. Закриваємо файли filename та "output.txt".
11.Виводимо повідомлення про успішне додавання тексту до файлу "output.txt".
12. Якщо файли не вдалося відкрити, виводимо повідомлення про помилку.

Алгоритм виконання допоміжної функції transliterate_text()

1.Функція приймає рядок **`input`** в якості вхідного параметру.
2.Встановлюється локаль для коректної роботи з українською мовою.
3.Створюється порожній рядок **`output`**, в який будуть додаватись перетворені символи.
4.Створюється **`unordered_map`**, яка містить пари символів української мови та їх перекладу на англійську мову.
5.Проходиться циклом по кожному символу `c` у вхідному рядку **`input`**.
6.Перевіряється, чи містить мапа символ **`c`**. Якщо так, то до рядку **`output`** додається його переклад. В іншому випадку символ **`c`** додається до рядку **`output`** без змін.
8.Функція повертає отриманий рядок **`output`**.

Алгоритм виконання задачі 10_3

1.Оголошуємо функція з назвою **`task10_3`**, яка приймає чотири аргументи типу **`double`** і **`int`** відповідно.
2.За допомогою функції **`setlocale`** встановлюємо локаль "uk_UA.UTF-8", що дозволяє працювати з українськими символами та рядками.

3.Використовуючи об'єкт класу **ofstream** з назвою **outfile,** відкриваємо файл "output.txt" для запису, додавання даних в кінець файлу (`ios_base::app`).
4.Обчислюється результат виконання функції `s_calculation` з аргументами `x`, `y` та `z`, та зберігається у змінній `result`.
5.За допомогою методу `<<` об'єкту `outfile`, результат обчислення записуємо у відкритий раніше файл "output.txt", разом із текстовим рядком "Результат: ".
6.За допомогою методу **<<** об'єкту **outfile**, виводимо значення аргументу `b` у двійковій системі числення, разом із текстовим рядком "b у двійковій системі числення: " та символом нового рядка.
7.Файл "output.txt" закривається за допомогою методу **close()**.
8.Виводиться рядок "Дані успішно додані до файлу 'output.txt'." у стандартний вихідний потік за допомогою **cout.**
9.Функція не повертає ніякого значення, оскільки її тип повернення визначений як **void**.

Висновок:
Мета цієї роботи полягала у набутті практичних умінь(пізнання як саме працювати з такими типами задачі, їх особливості та методи реалізації складних завданнях) і навичок застосування теоретичних положень методології модульного програмування, реалізації метода роботи з файлами введення/виведення(самі завдання були надані викладачем в репозиторії odorenskyi/Osadchyi-Vladyslav-KN22) мовою програмування C++ й використовуючи кросплатформове середовище Code::Blocks (GNU GCC Compiler).
У даній лабораторній роботі ми зосередилися на розробці програмних модулів оброблення даних складових типів з файлами введення/виведенням. Використовуючи мову програмування C++ у кросплатформовому середовищі Code::Blocks (GNU GCC Compiler), тіло нашого попереднього модуля з 8 лабораторної роботи(з іншими завданнями), та функції стандартних бібліотек(їхній список буде показано у самому коді нижче).

Сам хід нашої роботи:
Основна мета коду - це виконання різноманітних завдань, таких як розрахунок значення S за формулою(попереднє завдання минулої лабораторної роботи), запис різноманітної інформації до файлу, підрахунок кількості двокрапок у файлі, обробка текстового файлу і транслітерація тексту(використовуючи одну допоміжну створену функцію та її активацію).
1.Перевірка чи працює стара функція з попереднього коду-помилок не було знайдено
2.Аналіз всіх вимог, нюансів завданнях 10 лабораторної роботи й постанова конкретної задачі
3.Використання різноманітних інтернет ресурсів для покращення знань й знаходження певних методів для більш гарної роботи з файлами
4.Розробка кожної конкретної задачі беручи за увагу всі урахування та можливості покращити сам код
5.Підключення самого модуля до файлу TestDriver і початок активації самих функцій цього модуля
6.Переміщення самого файлу тестування у папку TestSuit
7.Початок повного тестування з різноманітними прикладами задач
8.У випадку знаходження помилок-початок аналізу помилок, можливості її виправлення та початок її корекції.Створення тест-кейсів
9.У випадку корекції всіх помилок-переносимо сам файл EXE у папку Software
10.Відправлення самого завдання у репозиторій
Від себе хочу додати, що виконання цієї роботи було дуже цікавим і повчальним.
Додатки для всього завдання 10.1(10_1,10_1_2,10_1_3)

```
#include "main.h"
#include <math.h>
#include <fstream>
#include <iostream>
#include <string>
#include <algorithm>
#include <clocale>
#include <time.h>
```

```
#include <map>
#include <unordered_map>
#include <bitset>
using namespace std;
double s_calculation(double x,double y,double z)
{   double
S=(1/2)*pow(x,2)-sqrt(fabs(pow((y+z),2)-pow(x,5)))-log(fabs(sin(z)));
    return S;

}

std::string task10_1(const string& filename)
{
    setlocale(LC_ALL, "uk_UA.UTF-8");
    ofstream file(filename, ios::app);
     if(file.is_open())
     {
      file << "Осадчий В.К" << endl
      << " ЦНТУ" << endl
      << " Кропивницький" << endl
      << " Україна" << endl
      << " 2023" << endl << endl;
      cout << "У файл була успішно записана інформація" << endl;
      file.close();
     }
     else
    {
     cout << "Помилка!Такого файлу немає" << endl;
    }
}
int task10_1_2(const string& filename)
{   setlocale(LC_ALL, "uk_UA.UTF-8");
    ifstream file(filename);
    if (file.is_open())
    {
    int count = 0;
    string line;
```

```
    while (getline(file, line))
    {
    count +=std::count(line.begin(), line.end(), ':');
    cout<<count<<endl;
    return count;
    }
    }
    else
    {
        cout<<"Помилка!Такого файлу немає"<< endl;
        return 0;
    }
}
int task10_1_3(const std::string& filename)
{
    setlocale(LC_ALL, "ukr");
    ifstream file(filename);
    if (!file.is_open()) {
        cerr << "Помилка! Неможливо відкрити файл " << filename <<
endl;
        return -1;
    }
    string line;
    string sentence;
    while (getline(file, line)) {
        for (char c : line) {
            if (c == '.' || c == '!' || c == '?') {
                // End of sentence
                string sentence_without_vowels;
                for (char s : sentence) {
                    if (s != 'a' && s != 'e' && s != 'i' && s != 'o' && s != 'u' &&
                        s != 'A' && s != 'E' && s != 'I' && s != 'O' && s != 'U') {
                        sentence_without_vowels += s;
                    }
                }
                cout << sentence_without_vowels << c << " ";
                sentence.clear();
```

```
            } else {
                sentence += c;
            }
        }
    }
    file.close();
    return 0;
}
std::string transliterate_text(const string& input) {
    setlocale(LC_ALL, "uk_UA.UTF-8");
    string output = "";
    unordered_map<char, string> translit_map {
        {'А', "A"}, {'Б', "B"}, {'В', "V"}, {'Г', "H"}, {'Ґ', "G"},
        {'Д', "D"}, {'Е', "E"}, {'Є', "Ye"}, {'Ж', "Zh"}, {'З', "Z"},
        {'И', "Y"}, {'І', "I"}, {'Ї', "Yi"}, {'Й', "Y"}, {'К', "K"},
        {'Л', "L"}, {'М', "M"}, {'Н', "N"}, {'О', "O"}, {'П', "P"},
        {'Р', "R"}, {'С', "S"}, {'Т', "T"}, {'У', "U"}, {'Ф', "F"},
        {'Х', "Kh"}, {'Ц', "Ts"}, {'Ч', "Ch"}, {'Ш', "Sh"}, {'Щ', "Shch"},
        {'Ь', ""}, {'Ю', "Yu"}, {'Я', "Ya"}, {'а', "a"}, {'б', "b"},
        {'в', "v"}, {'г', "h"}, {'ґ', "g"}, {'д', "d"}, {'е', "e"},
        {'є', "ie"}, {'ж', "zh"}, {'з', "z"}, {'и', "y"}, {'і', "i"},
        {'ї', "i"}, {'й', "i"}, {'к', "k"}, {'л', "l"}, {'м', "m"},
        {'н', "n"}, {'о', "o"}, {'п', "p"}, {'р', "r"}, {'с', "s"},
        {'т', "t"}, {'у', "u"}, {'ф', "f"}, {'х', "kh"}, {'ц', "ts"},
        {'ч', "ch"}, {'ш', "sh"}, {'щ', "shch"}, {'ь', ""}, {'ю', "iu"},
        {'я', "ia"}
    };

    for (const char& c : input) {
        if (translit_map.count(c) > 0) {
            output += translit_map[c];
            cout<<output<<endl;
        }
        else {
            output += c;
        }
    }
```

```
    return output;
}
```

Назва тестового набору Test Suite Description	TestSuit 10.1
Name of Project / Software	Osadchyi-Vladyslav-KN22/lab10/TestSuit/TestSuit 10.1
Рівень тестування Level of Testing	Модульне тестування
Автор тест-сьюта Test Suite Author	Осадчий Владислав Костянтинович
Виконавець Implementer	Осадчий Владислав Костянтинович

Action	Expected Result	Test Result(passed/failed/blocked)
task10_1(назва файлу)	Осадчий В.К ЦНТУ Кропивницький Україна 2023	passed
task10_1(Enter)	Створення файлу з назвою “-” та записання самої іфнормації від функції(як було вище)	passed

Назва тестового набору Test Suite Description	TestSuit 10.1.2
Name of Project / Software	Osadchyi-Vladyslav-KN22/lab10/TestSuit/TestSuit 10.1.2

Рівень тестування Level of Testing	Модульне тестування
Автор тест-сьюта Test Suite Author	Осадчий Владислав Костянтинович
Виконавець Implementer	Осадчий Владислав Костянтинович

input file	Action	Expected Result	Test Result(passed/failed/blocked)
-	task10_1_2(Назва неіснуючогор файлу))	Помилка!Такого файлу немає	passed
Порожній	task10_1_2(input file))	0	passed
:	task10_1_2(input file))	1	passed
У саду росли дерева:сливи, груші, яблуні.Біля ставка росли квіти:конвалії, нарциси.	task10_1_2(input file))	2	passed

Назва тестового набору Test Suite Description	TestSuit 10.1.3
Name of Project / Software	Osadchyi-Vladyslav-KN22/lab10/TestSuit/TestSuit 10.1.3
Рівень тестування Level of Testing	Модульне тестування
Автор тест-сьюта Test Suite Author	Осадчий Владислав Костянтинович

Виконавець Implementer	Осадчий Владислав Костянтинович

input file	Action	Expected Result	Test Result(passed/failed/blocked)
-	task10_1_3(Назва неіснуючогр файлу)	Помилка! Неможливо відкрити файл	passed
Порожній	task10_1_3(input file)	Нічого	passed
Виконав домашнє завдання.	task10_1_3(input file)	Вкнв дмшн звднн	passed
У саду росли дерева:сливи, груші, яблуні.Біля ставка росли квіти:конвалії, нарциси.Біля мене були тварини:кацапи, курки та ведмеді.	task10_1_3(input file)	У сд рсл дрв:слв, грш, йблн.Бл ствк рсл квт:кнвл, нрцс.Бл мн бли тврн:кцп, крк т вдмд.	passed

Додатки для завдання 10_2

```
void task10_2(const string& filename) {
    setlocale(LC_ALL, "uk_UA.UTF-8");
    time_t now = time(0);
    tm* ltm = localtime(&now);
    string date = to_string(ltm->tm_mday) + "\\" + to_string(ltm->tm_mon +
1) + "\\" + to_string(ltm->tm_year + 1900);
    ifstream input_file(filename);
    ofstream output_file("output.txt");
    if (input_file.is_open() && output_file.is_open()) {
        string line;
        while (getline(input_file, line)) {
```

```
            string transliterated_line = transliterate_text(line);
            output_file << transliterated_line << endl;
        }
        output_file << date << endl;
        input_file.close();
        output_file.close();

        cout << "Текст успішно доданий до файлу з назвою 'output.txt'."
<<endl;
    }
    else {
    cout << "Помилка!Такого файлу немає" <<endl;
    }
}
```

Назва тестового набору Test Suite Description	TestSuit 10.2
Name of Project / Software	Osadchyi-Vladyslav-KN22/lab10/TestSuit/TestSuit 10.2
Рівень тестування Level of Testing	Модульне тестування
Автор тест-сьюта Test Suite Author	Осадчий Владислав Костянтинович
Виконавець Implementer	Осадчий Владислав Костянтинович

input file	Action	Expected Result	Test Result(passed/failed/blocked)
-	task 10_2(Назва неіснуючогор файлу))	Помилка!Такого файлу немає	passed
Порожній	task10_2(input file))	Нічого	passed

Я зробив домашнє завдання.	task10_2(input file))	Ya zrobyv domashnie zavdannia.	passed
У саду росли яблука та груши.	task10_2(input file))	U sadu rosly yabluka ta hrushy.	passed

Додатки для завдання 10_3

```
double task10_3(double x, double y, double z, int b) {
    setlocale(LC_ALL, "uk_UA.UTF-8");
    ofstream outfile("output.txt", ios_base::app);
    double result = s_calculation(x, y, z);
    outfile << "Результат: " << result << endl;
    outfile << "b у двійковій системі числення: " << bitset<32>(b) <<
endl;
    outfile.close();
    cout << "Дані успішно додані до файлу 'output.txt'." << endl;
}
```

Назва тестового набору Test Suite Description	TestSuit 10.3
Name of Project / Software	Osadchyi-Vladyslav-KN22/lab10/TestSuit/TestSuit 10.3
Рівень тестування Level of Testing	Модульне тестування
Автор тест-сьюта Test Suite Author	Осадчий Владислав Костянтинович
Виконавець Implementer	Осадчий Владислав Костянтинович

Action	Expected Result	Test Result(passed/failed/blocked)
task10_3(0,0,0,0)	Результат: inf b у двійковій системі числення: 00000000000000000000000000000000	passed
task10_3(1,1,1,1)	Результат: -1.55945 b у двійковій системі числення: 00000000000000000000000000000001	passed
task10_3(0,0,0,0)	Результат: -53.949 b у двійковій системі числення: 00000000000000000000000000001101	passed
task10_3(5,6,7,13)	Результат: -53.949 b у двійковій системі числення: 00000000000000000000000000001101	passed
task10_3(100,140,500,8000)	Результат: -99997.2 b у двійковій системі числення: 00000000000000000001111101000000	passed
task10_3(-100,0,9,-123)	Результат: -99999.1 b у двійковій системі числення: 11111111111111111111111000000101	passed