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 СЛАДОВИХ ТИПІВ З ФАЙЛОВИМ ВВЕДЕННЯМ/ВИВЕДЕННЯМ

ВИКОНАЛА
студентка академічної групи

КБ-22-2 Красніцька
Єлизавета

ПЕРЕВІРИВ

викладач кафедри
кібербезпеки
та програмного забезпечення
Олександр Собінов

Кропивницький – 2023

## Лабораторна робота №10

## Реалізація програмних модулів оброблення даних сладових типів з файловим введенням/виведенням

*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.

### Завдання до лабораторної роботи

1. Реалізувати програмні модулі розв'язування задач 10.1–10.3 як складові статичної бібліотеки libModulesПрізвище.a (проект ModulesПрізвище лабораторних робіт №8–9).

2. Реалізувати тестовий драйвер автоматизованої перевірки програмних модулів розв'язування задач 10.1–10.3.

### Варіант 10

**ВХІДНИЙ ТЕКСТ - ВМІСТ ВХІДНОГО ТЕКСТОВОГО ФАЙЛУ**

Як паросте виноградної лози, плекайте мову.
Пильно й ненастанно політь бур'ян.
Чистіта від сльози вона хай буде.

Вірно і слухняно нехай вона щоразу служить вам,
Хоч і живе своїм живим життям.

**ЗАДАЧА 10.1**

У вихідний текстовий файл записати:

◦ авторську інформацію: ім’я й прізвище розробника модуля, установа/організація, місто, країна, рік розробки;

◦ випадкове число від 10 до 100;

повідомлення, чи текст віршу із вхідного файла має пунктуаційні помилки (відповідно до авторського оригіналу).

**Алгроритм виконання:**

- Ініціалізуємо генератор випадкових чисел.
- Оголошуємо змінну authorInfo, яка містить інформацію про автора.
- Генеруємо випадкове число від 10 до 100 за допомогою rand() та зберігаємо його у змінну randomNumber.
- Оголошуємо змінну inputText та відкриваємо файл input.txt за допомогою ifstream.
- Зчитуємо вміст файлу input.txt у змінну inputText за допомогою getline().
- Перевіряємо текст на наявність помилок пунктуації за допомогою функції checkPunctuation() та зберігаємо результат у змінну hasPunctuationMistakes.
- Оголошуємо змінну outputFile та відкриваємо файл output.txt за допомогою ofstream.
- Записуємо у файл output.txt інформацію про автора, випадкове число та наявність помилок пунктуації.
- Закриваємо файли input.txt та output.txt.
- Завершуємо програму зі статусом 0.

**Лістинг програми:**

```
#include <iostream>
#include <fstream>
#include <cstdlib>
#include <ctime>
#include <string>

using namespace std;


bool checkPunctuation(string text) {
   string originalText = "Як паросты виноградної лози, плекайте мову.\n"
```

```
                "Пильно й ненастанно політь бур’ян.\n"
                "Чистіта від сльози вона хай буде.\n"
                "Вірно і слухняно нехай вона щоразу служить вам,\n"
                "Хоч і живе своїм живим життям.";


    for (int i = 0; i < text.length(); i++) {
        if (ispunct(text[i]) || isspace(text[i])) {
            text.erase(i--, 1);
        }
    }


    return text == originalText;
}

int main() {
    srand(time(NULL));  // Ініціалізуємо генератор випадкових чисел
    string authorInfo = "Красніцька Єлизавета м.Кропивницький 01.05.2023";
    int randomNumber = rand() % 91 + 10;  // Генеруємо випадкове число від 10 до 100
    string inputText;
    ifstream inputFile("input.txt");


    getline(inputFile, inputText, '\0');


    bool hasPunctuationMistakes = !checkPunctuation(inputText);


    ofstream outputFile("output.txt");
    outputFile << authorInfo << endl;
    outputFile << "Випадкове число: " << randomNumber << endl;
    outputFile << "Текст має пунктуаційні помилки: " << (hasPunctuationMistakes ?
"так" : "ні") << endl;

    inputFile.close();
    outputFile.close();

    return 0;
```

}

**ЗАДАЧА 10.2**

У вхідний текстовий файл дописати:

• дату дозапису інформації у форматі «ДД.ММ.РРРР».

**Алгоритм виконання:**

- Створити об'єкт ofstream з ім'ям outfile і відкрити файл "input.txt" для дозапису.
- Перевірити, чи вдалося відкрити файл outfile.
- Отримати поточний час у секундах від початку епохи (01.01.1970 00:00:00 UTC) з допомогою функції time.
- Перетворити поточний час у структуру часу з допомогою функції localtime.
- Створити буфер char з ім'ям date розміром 11, який буде містити дату у форматі "DD.MM.YYYY".
- Отримати дату у форматі "DD.MM.YYYY" з поточного часу з допомогою функції strftime та зберегти її у буфері date.
- Дописати до файлу outfile рядок, що містить дату дозапису у форматі "Дата дозапису: DD.MM.YYYY".
- Закрити файл outfile.
- Повернути значення 0.

Отже, після виконання цього коду в файлі "input.txt" буде додано рядок з датою дозапису у форматі "Дата дозапису: DD.MM.YYYY". Якщо відкриття файлу завершиться неуспішно, програма виведе повідомлення про помилку та поверне значення 1.

**Лістинг програми:**

```
#include <iostream>
#include <fstream>
#include <ctime>

int main() {
   std::ofstream outfile("input.txt", std::ios_base::app); // відкрити файл для дозапису
   if (!outfile) { // перевірити, чи вдалося відкрити файл
      std::cerr << "Помилка: не вдалося відкрити файл для дозапису!\n";
      return 1;
   }

   std::time_t now = std::time(nullptr); // отримати поточний час
   std::tm* now_tm = std::localtime(&now); // перетворити у структуру часу
   char date[11]; // буфер для дати у форматі "DD.MM.YYYY"
```

```
   std::strftime(date, sizeof(date), "%d.%m.%Y", now_tm); // отримати дату у форматі "DD.MM.YYYY"

   outfile << "\nДата дозапису: " << date << "\n"; // дописати дату до файлу
   outfile.close(); // закрити файл
   return 0;
}
```

**ЗАДАЧА 10.3**

Вхідні дані - числові значення x, y, з та натуральне число в. У вихідний текстовий файл дописати:

- результати виконання функцій із заголовкового файлу Modules.h s_calculation 3 аргументами x, У, =;

число в у двійковому коді

**Алгоритм виконання:**

- Відкрити консоль.
- Запустити програму.
- Ввести числа x, y, z та натуральне число v.
- Програма відкриє файл output.txt в режимі допису (якщо файл не існує, він буде створений).
- Якщо не вдалося відкрити файл, вивести повідомлення про помилку та завершити роботу програми.
- Обчислити результат функції s_calculation з аргументами x, y, z та записати результат у файл output.txt.
- Перетворити число v у двійкову систему числення, зберігши кожен розряд у масиві binaryV.
- Записати число v у двійковому коді у файл output.txt.
- Закрити файл output.txt.
- Завершити роботу програми.

**Лістинг програми:**

```
#include <iostream>
#include <fstream>
#include "Krasnitska.h"

using namespace std;

int main() {
   double x, y, z;
   int v;

   cout << "Введіть числа x, y, z та натуральне число v:" << endl;
```

```
    cin >> x >> y >> z >> v;


    ofstream outFile;
    outFile.open("output.txt", ios_base::app);


    if (!outFile) {
        cerr << "Не вдалося відкрити файл output.txt" << endl;
        return 1;
    }


    outFile << "Результати функції s_calculation з аргументами " << x << ", " << y << ", "
<< z << ": " << s_calculation(x, y, z) << endl;


    int binaryV[32];
    int i = 0;
    while (v > 0) {
        binaryV[i] = v % 2;
        v /= 2;
        i++;
    }
    outFile << "Число " << v << " у двійковому коді: ";
    for (int j = i - 1; j >= 0; j--) {
        outFile << binaryV[j];
    }
    outFile << endl;

    // Закриття файлу
    outFile.close();

    return 0;
}
```

**Висновок:**

Мета цієї роботи полягає у набутті грунтовних 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. Для виконання даної лабораторної роботи потрібно приблизно - 4 академічні
години.
Також потрібно таке обладнення:
~ персональний комп'ютер з операційною системою Windows; v вільне кросплатформове Code:: Blocks IDE (www.codeblocks.org); v текстовий редактор (OpenOffice Writer, Microsoft Word або ін.);
v - файл-шаблон тестового набору Artifact_TEST_SUITE_lab.doc;
v власний обліковий запис на GitHub
При підготовці до лабораторної роботи я отримала такі завдання.

- Реалізувати програмні модулі розв'язування задач 10.1-10.3 як складові статичної бібліотеки libModules.a (проект Modules лабораторних робіт №8-9).
- Реалізувати
тестовий драйвер автоматизованої
перевірки
програмних модулів розв'язування задач 10.1-10.3.

Я дотримуваласьтакого плану виконання лабораторної роботи No9:

1) Спочатку я зайшла на мій Git Repositories та отримав завдання для виконання лабораторної роботи No10.
2) Завантажила Git Repositories на мій диск.
3) Змінила вміст файлу README md, вказавши: тему, мету, варіант та завдання 10.1-10.3.
4) Створила теки: prj, SoftWare, TestSuite, Report. Також завантажив все на репозиторій.
5) Здійснила аналіз задач 10.1 - 10.3.
6) Задокументувала артефакти аналізу та постановки задачі в звіт.
7) З лабораторної роботи No9, перемістила статичну бібліотеку ModulesKaras в папку prj.
8) Реалізувала функції: Task_10_1, Task_10_2, Task_10_3.
9) Скомпілювала проєкт статичної бібліотеки.
10) З лабораторної роботи No9, перемістила заголовковий файл ModulesKaras.h, в ньому вписала прототипи функцій Task_10_1, Task_10_2, Task_10_3.
11) В Code::Blocks створила проєкт консольного застосунку C++, іменував його як TestDriver.
12) Реалізувала мовою програмування C+, тестовий драйвер, підключила статичну бібліотеку та заголовковий файл в налаштуваннях компілятора.
13) Скопіювала текст з консольного вікна та перемістила його в ЗВІТ.
14) Закінчила працювати з звітом.
15) Надіслала всі файли на Git Repositorie.

Данна лабораторна робота розширила мої знання. Ми перший раз почали працювати на пряму з файлами. Всі додатки знаходяться в кінці звіту. Вона мені сподобалась, також вона була чудова!

## Додаток А - Test Suite

Artifact: Test Suite

Date: 5/7/2023

| | |
|---|---|
| Назва тестового набору<br>Test Suite Description | TestSuite |
| Рівень тестування<br>Level of Testing | автоматизований / Unit Testing |
| Автор тест-сьюта<br>Test Suite Author | Красніцька Єлизавета |
| Виконавець<br>Implementer | Красніцька Єлизавета |

| **10.1** | | | |
|---|---|---|---|
| Preliminary Steps | Action (test steps) | Expected Result | Result |
| Вихідний файл output.txt:<br><br>Вхідний файл input.txt:<br>*Як паросты виноградної лози,*<br>*плекайте мову.*<br>*Пильно й ненастанно політь*<br>*бур'ян.*<br>*Чистіша від сльози вона хай буде.*<br>*Вірно і слухняно нехай вона*<br>*щоразу служить вам,*<br>*Хоч і живе своїм живим життям.* | Викликаємо function_10_1() | Вихідний файл output.txt:<br>Красніцька Єлизавета<br>*м.Кропивницький 01.05.2023*<br>*(число від 10 до 100)*<br>*Пунктуаційних помилок немає*<br><br>Вхідний файл input.txt:<br>*Як паросты виноградної лози,*<br>*плекайте мову.*<br>*Пильно й ненастанно політь*<br>*бур'ян.*<br>*Чистіша від сльози вона хай буде.*<br>*Вірно і слухняно нехай вона*<br>*щоразу служить вам,*<br>*Хоч і живе своїм живим життям.* | |

| **10.2** | | | |
|---|---|---|---|
| Preliminary Steps | Action (test steps) | Expected Result | Result |
| Вихідний файл input.txt:<br>*Як паросты виноградної лози,*<br>*плекайте мову.*<br>*Пильно й ненастанно політь*<br>*бур'ян.*<br>*Чистіша від сльози вона хай буде.*<br>*Вірно і слухняно нехай вона*<br>*щоразу служить вам,*<br>*Хоч і живе своїм живим життям.* | Викликаємо function_10_2() | Вихідний файл input.txt:<br>*Як паросты виноградної лози,*<br>*плекайте мову.*<br>*Пильно й ненастанно політь*<br>*бур'ян.*<br>*Чистіша від сльози вона хай буде.*<br>*Вірно і слухняно нехай вона*<br>*щоразу служить вам,*<br>*Хоч і живе своїм живим життям.*<br><br>*01.05.2023* | |

| 10.3 | | | |
|---|---|---|---|
| Preliminary Steps | Action (test steps) | Expected Result | Result |
| Вхідні дані x,y,z,v | Викликаємо function_10_3(1, 2, 5, 6) | Вихідні дані:<br>*-1262.22*<br>*00000000000000000000000000000110* | |