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”
на тему
“Реалізація програмних модулів оброблення даних складових типів з файловим введенням / виведенням”

Виконав
студент академічної
групи КІ-22-2
_____ І.М. Карась

Приймає
викладач кафедри
кібербезпеки та ПЗ
_____ О.Г. Собінов

Кропивницький – 2022

Лабораторна робота №10

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 9

— Вхідний текст - вміст вхідного текстового файлу —

Довільний текст з довільної кількості абзаців українською або англійською.

— Задача 10.1 —

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

- авторську інформацію: ім'я й прізвище розробника модуля, установа/організація, місто, країна, рік розробки;
- текст із вхідного файла з переставленими місцями абзацами (перший – останній, другий – передостанній і т.д.);
- повідомлення, до англійської чи української мови належить текст у вхідному файлі.

— Задача 10.2 —

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

- кількість ком та крапок у ньому, дату й час дозапису інформації.

— 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` створється 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`.

Аналіз та постановка виконання:

Аналіз виконання Task 10.1:

Потрібно у вихідний файл записати: авторську інформацію:

- Ім’я й прізвище.
- Організацію.
- Місто.
- Країна.
- Рік розробки.

Також потрібно переставити абзаци з вхідного файлу (перший – останній, другий – передостанній, і т. д.)

Ще потрібно додати повідомлення, якою мовою написаний вхідний файл.

Постановка Task_10.1:

Вхідні файли: файл з текстом українською, або англійською, який містить абзаци.

Вихідні файли: файл з анотацією; повідомлення, якою мовою написаний текст; зміненні місцями абзаци.

Аналіз виконання Task 10.2:

Потрібно у вхідний файл дописати: кількість ком та крапок у ньому, також додати час та дату компіляції.

Постановка Task_10.2:

Вхідні файли: файл з текстом українською, або англійською, який містить абзаци.

Вихідні файли: файл з абзацами українською, або англійською, кількість ком та крапок, дату та час компіляції.

Аналіз виконання Task 10.3:

Потрібно дописати результат функції s_calculation, та дописати перетворене натуральне число b у двійковому коді.

Постановка Task_10.2:

Вхідні файли: файл з текстом українською, або англійською, який містить змінені абзаци; x,y,z – змінні для функції s_calculation; b - натуральне число.

Вихідні файли: файл з зміненими абзацами українською, або англійською; результат функції s_calculation; двійкове число b.

Висновок:

Мета цієї 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. Для виконання даної лабораторної роботи потрібно приблизно – 4 академічні години.

Також потрібно таке обладнання:

- ✓ персональний комп'ютер з операційною системою Windows;
- ✓ вільне кросплатформове Code::Blocks IDE (www.codeblocks.org);
- ✓ текстовий редактор (OpenOffice Writer, Microsoft Word або ін.);
- ✓ – файл-шаблон тестового набору Artifact_TEST_SUITE_lab.doc;
- ✓ власний обліковий запис на GitHub https://github.com/ та Git-репозиторій https://github.com/odorenskyi/Karas-Illia-KI-22-2

При підготовці до лабораторної роботи я отримав такі завдання.

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.

Я дотримувався такого плану виконання лабораторної роботи №9:

1) Спочатку я зайшов на мій Git Repositories та отримав завдання для виконання лабораторної роботи №10.
2) Завантажив Git Repositories на мій диск.
3) Змінив вміст файлу *README.md,* вказавши: тему, мету, варіант та завдання 10.1-10.3.
4) Створив теки: prj, SoftWare, TestSuite, Report. Також завантажив все на репозиторій.
5) Здійснив аналіз задач 10.1 –10.3.
6) Задокументував артефакти аналізу та постановки задачі в звіт.
7) З лабораторної роботи №9, перемістив статичну бібліотеку *ModulesKaras* в папку prj .
8) Реалізував функції: *Task_10_1, Task_10_2, Task_10_3*.
9) Скомпілюв проєкт статичної бібліотеки.
10) З лабораторної роботи №9, перемістив заголовковий файл *ModulesKaras.h*, в ньому вписав прототипи функцій *Task_10_1, Task_10_2, Task_10_3.*
11) *У* Code::Blocks створив проєкт консольного застосунку C++, іменував його як *TestDriver*.
12) Реалізував мовою програмування C++, тестовий драйвер, підключив статичну бібліотеку та заголовковий файл в налаштуваннях компілятора.
13) Скопіював текст з консольного вікна та перемістив його в звіт.
14) Закінчив працювати з звітом.
15) Надіслав 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 №1:

Додаток результат виконання TestDriver (з консольного вікна):

```
<<<< Test_10_1 >>>>

Test case #01 PASSED.

Test case #02 PASSED.

<<<< Test_10_2 >>>>

Test case #01 PASSED.

Test case #02 PASSED.

<<<< Test_10_3 >>>>

Test case #01 PASSED.

Test case #02 PASSED.
```

Додаток №2:

Вихідний код проєкту ModulesKaras:

```
#include <cmath>
#include <iostream>
#include <bitset>

#include <fstream>
#include <locale.h>
#include <Windows.h>
#include <ctime>
#include <cstring>

using namespace std;

float s_calculation(int x, int y, int z){
    return (pow((2 * z + 1), x)) - (sqrt(abs(y - 0.5 * z))) + z + 
3.141592653589793;
}

int Task_9_1(int posadOklad, int zarobPlata, int year){
    int nadbavka, number1;
    if (year >= 3 && year < 5){
        nadbavka = (posadOklad*10)/100;
        number1 = zarobPlata+posadOklad+nadbavka;
    }
    if (year >= 5 && year < 7){
        nadbavka = (posadOklad*15)/100;
        number1 = zarobPlata+posadOklad+nadbavka;
    }
    if (year >= 7 && year < 15){
        nadbavka = (posadOklad*20)/100;
        number1 = zarobPlata+posadOklad+nadbavka;
    }
    if (year >= 15){
        nadbavka = (posadOklad*25)/100;
```

```
            number1 = zarobPlata+posadOklad+nadbavka;
        }
        return number1;
    }

    int Task_9_2_1(int one, int two, int three, int four, int five, int six){
        int celsia;
        celsia = (one+two+three+four+five+six)/6;
        return celsia;
    }

    int Task_9_2_2(int celsia){
        int farangeit;
        farangeit = (32 + 1.8 * celsia);
        return farangeit;
    }

    int Task_9_3(int N){
        bitset<32>b_number(N);
        if (N >= 0 && N < 10008000){
            if (b_number[10] == 0){
                return 32-b_number.count();
            }
            else if (b_number[10] == 1){
                return b_number.count();
            }
        }
        else{
            return 0;
        }
    }

    char Task_10_1(char in_path[], char out_path[]){

        string annotationsUkr = {"|---------------------------------------------
-------------------------------------|\n|Карась          Ілля          Миколайович,
|\n|Центральноукраїнського національного технічного університету,|\n|Кропивницький,
Україна, Рік розробки - 28.05.2023                    |\n|-------------------------
-------------------------------------------------------|\n"};
        string annotationsEng = {"|---------------------------------------------
-------------------------------------|\n|Karas         Illia         Mikolayovitch,
|\n|Central          Ukrainian          National          Technical          University,
|\n|Kropyvnytskyi, Ukraine, Year of construction  - 28.05.2023                |\n|---
-------------------------------------------------------------------------------
|\n"};
        ofstream fout;
        ifstream fin;

        fout.open(out_path, ofstream::app);
        fin.open(in_path);

        if(!fin.is_open()){
            cout << "Помилка відкривання файлу №1" << endl;
        }
        else{
            char ch;
            string str1, str2, str3, str4;
            int num1, num2, num3, k = 0;
            while(fin.get(ch)){
                if(ch >= 'a' && ch <= 'я') k++;
                if(ch != '@') str1 += ch;
                else {
```

```
                num1 = fin.tellg();
                break;
            }
        }
        fin.seekg(num1, ios_base::beg);
        while(fin.get(ch)){
            if(ch != '@') str2 += ch;
            else {
                num2 = fin.tellg();
                break;
            }
        }
        fin.seekg(num2, ios_base::beg);
        while(fin.get(ch)){
            if(ch != '@') str3 += ch;
            else {
                num3 = fin.tellg();
                break;
            }
        }
        if(k >= 1) num3 -= 2;
        fin.seekg(num3, ios_base::beg);
        while(fin.get(ch)){
            if(ch != '@') str4 += ch;
            else break;
        }
        if(k > 2) {
            fout << annotationsUkr;
            fout << "Текст написаний Українською мовою\n\n";
            fout << '\t';
            fout << str4;
            fout << '\n';
            fout << '\t';
            fout << str3;
            fout << str2;
            fout << '\n';
            fout << str1;
        }
        else {
            fout << annotationsEng;
            fout << "The text is written in English\n\n";
            fout << '\t';
            fout << str4;
            fout << '\n';
            fout << '\t';
            fout << str3;
            fout << str2;
            fout << '\n';
            fout << str1;;
        }
    }
    fin.close();
    fout.close();
}

char Task_10_2(char in_path[]){

    ifstream fin;
    ofstream fout;

    time_t rawtime;
    time(&rawtime);
```

```
    fin.open(in_path);
    fout.open(in_path, ofstream::app);
    if(!fin.is_open()){
        cout << "Помилка відкривання файлу №2 " << endl;
    }
    else{
        int koma, krapka, k;
        char ch;
        while(fin.get(ch)){
            if(ch >= 'a' && ch <= 'я') k++;
            if(ch == ',') koma++;
            if(ch == '.') krapka++;
        }
        if (k >= 2) {
            fout << '\n';
            fout << "В данному тексті ком: = ";
            fout << koma;
            fout << '\n';
            fout << "В данному тексті крапок: = ";
            fout << krapka;
            fout << '\n';
            fout << ctime(&rawtime);
        }
        else {
            fout << '\n';
            fout << "In this text com: = ";
            fout << koma;
            fout << '\n';
            fout << "In this text dots: = ";
            fout << krapka;
            fout << '\n';
            fout << ctime(&rawtime);
        }
    }
    fin.close();
    fout.close();
}

void Task_10_3(int x, int y, int z, int b, char out_path[]){

    ofstream fout;
    int bin = 0, k = 1;
    while (b){
        bin += (b % 2) * k;
        k *= 10;
        b /= 2;
    }
    fout.open(out_path, ofstream::app);
    fout << endl;
    fout << "S: = ";
    fout << s_calculation(x,y,z);
    fout << endl;
    fout << "Число b, в двійкомову коді: = ";
    fout << bin;
    fout.close();
}
```

Вихідний код проєкту TestDriver:

```
#include <iostream>
#include <windows.h>
#include <cstring>
#include <fstream>
#include <ctime>
#include "ModulesKaras.h"

using namespace std;

void testdriver_1(void);
void testdriver_2(void);
void testdriver_3(void);
string ReadFile(string path_file);
string clearText(string out_path);

int main()
{
    SetConsoleCP(1251);
    SetConsoleOutputCP(1251);
    testdriver_1();
    testdriver_2();
    testdriver_3();

    return 0;
}

void testdriver_1(void){

    char   in_path[]   =   "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text.txt";
    char in_path_ENG[] = "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text-ENG.txt";
    char  out_path[]  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\out-text.txt";
    string out_path2 = "C:\\Users\\user\\Desktop\\GitHub\\ Karas-Illia-KI-22-
2\\lab10\\TextFile\\out-text.txt";
    string expected_test_10_1_1 = "C:\\Users\\user\\Desktop\\GitHub\\ Karas-
Illia-KI-22-2\\lab10\\TextFile\\Expected Task_10_1\\expected_test_10_1_1.txt";
    string expected_test_10_1_2 = "C:\\Users\\user\\Desktop\\GitHub\\ Karas-
Illia-KI-22-2\\lab10\\TextFile\\Expected Task_10_1\\expected_test_10_1_2.txt";
    string      reader,      result_task_10_1_1,      Rexpected_test_10_1_1,
Rexpected_test_10_1_2;

    clearText(out_path2);
    Task_10_1(in_path, out_path);
    Rexpected_test_10_1_1 = ReadFile(expected_test_10_1_1);
    reader = ReadFile(out_path2);
    cout << "<<<< Test_10_1 >>>>" << endl;
    if(reader == Rexpected_test_10_1_1) cout << "\nTest case #01 PASSED.\n";
    else cout << "\nTest case #01 FAILED.\n";

    clearText(out_path2);
    Task_10_1(in_path_ENG, out_path);
    Rexpected_test_10_1_2 = ReadFile(expected_test_10_1_2);
    reader = ReadFile(out_path2);
    if(reader == Rexpected_test_10_1_2) cout << "\nTest case #02 PASSED.\n";
    else cout << "\nTest case #02 FAILED.\n";
}

void testdriver_2(void){

    time_t rawtime;
```

```
          time(&rawtime);
          char  in_path[]  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text.txt";
          string  in_path2  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text.txt";
          char in_path_ENG[] = "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text-ENG.txt";
          string in_path_ENG2 = "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text-ENG.txt";
          string  expected_path_10_2_1  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-
Illia-KI-22-2\\lab10\\TextFile\\Expected Task_10_2\\expected_test_10_2_1.txt";
          string  expected_path_10_2_2  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-
Illia-KI-22-2\\lab10\\TextFile\\Expected Task_10_2\\expected_test_10_2_2.txt";
          string reader, Rexpected_test_10_2_1, Rexpected_test_10_2_2;
          Task_10_2(in_path);
          Rexpected_test_10_2_1 = ReadFile(expected_path_10_2_1);
          Rexpected_test_10_2_1 += ctime(&rawtime);
          reader = ReadFile(in_path2);
          cout << "\n<<<< Test_10_2 >>>>" << endl;
          if(reader == Rexpected_test_10_2_1) cout << "\nTest case #01 PASSED.\n";
          else cout << "\nTest case #01 FAILED.\n";

          Task_10_2(in_path_ENG);
          Rexpected_test_10_2_2 = ReadFile(expected_path_10_2_2);
          Rexpected_test_10_2_2 += ctime(&rawtime);
          reader = ReadFile(in_path_ENG2);
          if(reader == Rexpected_test_10_2_2) cout << "\nTest case #02 PASSED.\n";
          else cout << "\nTest case #02 FAILED.\n";
      }

      void testdriver_3(void){

          char  in_path[]  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\in-text-UKR.txt";
          char  out_path[]  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\out-text.txt";
          string  out_path2  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-Illia-KI-22-
2\\lab10\\TextFile\\out-text.txt";
          string  expected_path_10_3_1  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-
Illia-KI-22-2\\lab10\\TextFile\\Expected Task_10_3\\expected_test_10_3_1.txt";
          string  expected_path_10_3_2  =  "C:\\Users\\user\\Desktop\\GitHub\\Karas-
Illia-KI-22-2\\lab10\\TextFile\\Expected Task_10_3\\expected_test_10_3_2.txt";
          string reader, Rexpected_test_10_3_1, Rexpected_test_10_3_2;
          int number1[2] = {1, 3};
          int number2[2] = {2, 2};
          int number3[2] = {3, 1};
          int b[2] = {36, 63};
          Task_10_3(number1[0], number2[0], number3[0], b[0], out_path);
          Rexpected_test_10_3_1 = ReadFile(expected_path_10_3_1);
          reader = ReadFile(out_path2);
          cout << "\n<<<< Test_10_3 >>>>" << endl;
          if(reader == Rexpected_test_10_3_1) cout << "\nTest case #01 PASSED.\n";
          else cout << "\nTest case #01 FAILED.\n";

          clearText(out_path2);
          Task_10_1(in_path, out_path);
          Task_10_3(number1[1], number2[1], number3[1], b[1], out_path);
          Rexpected_test_10_3_2 = ReadFile(expected_path_10_3_2);
          reader = ReadFile(out_path2);
          if(reader == Rexpected_test_10_3_2) cout << "\nTest case #02 PASSED.\n";
          else cout << "\nTest case #02 FAILED.\n";
      }
```

```
string ReadFile(string path_file){
    ifstream file(path_file);
    if(!file.is_open()) return "ERROR";
    string text = "";
    string line;
    while (getline(file, line)){
        text += line + '\n';
    }
    file.close();
    return text;
}

string clearText(string out_path){
    ofstream fout;
    fout.open(out_path);
    fout << "";
    fout.close();
    return out_path;
}
```