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

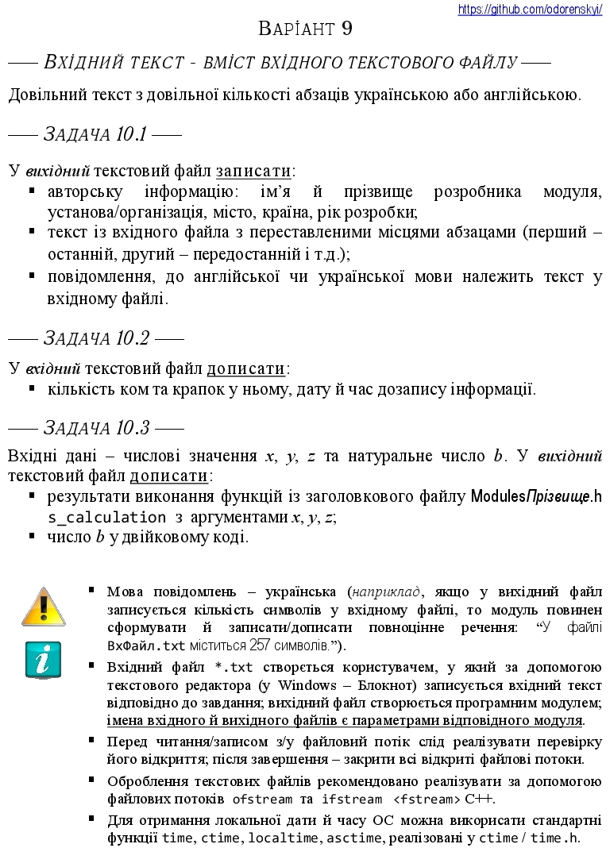
кібербезпеки та ПЗ

\_\_\_\_\_\_\_\_\_  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 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.

****

***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 зміненні 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 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;
* вільне кросплатформове Code::Blocks IDE ([www.codeblocks.org](http://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.а (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 С++, іменував його як *TestDriver*.
12. Реалізував мовою програмування С++, 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 вiдкривання файлу №1" << endl;*

*}*

*else{*

*char ch;*

*string str1, str2, str3, str4;*

*int num1, num2, num3, k = 0;*

*while(fin.get(ch)){*

*if(ch >= 'а' && 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 вiдкривання файлу №2 " << endl;*

*}*

*else{*

*int koma, krapka, k;*

*char ch;*

*while(fin.get(ch)){*

*if(ch >= 'а' && 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;

}