Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 8

з навчальної дисципліни

“Базові методології та технології програмування”

РЕАЛІЗАЦІЯ СТАТИЧНИХ БІБЛІОТЕК МОДУЛІВ ЛІНІЙНИХ ОБЧИСЛЮВАЛЬНИХ ПРОЦЕСІВ

ЗАВДАННЯ ВИДАВ

доцент кафедри кібербезпеки та програмного забезпечення

Доренський О. П.

[https://github.com/odorenskyi/](https://github.com/odorenskyi/Dmytro-Parkhomenko-KB18)

ВИКОНАВ

студент академічної групи КН-23

Гребенюк Д. О.

ПЕРЕВІРИВ

ст. викладач кафедри кібербезпеки   
та програмного забезпечення

Дрєєва Г. М.

Кропивницький – 2024

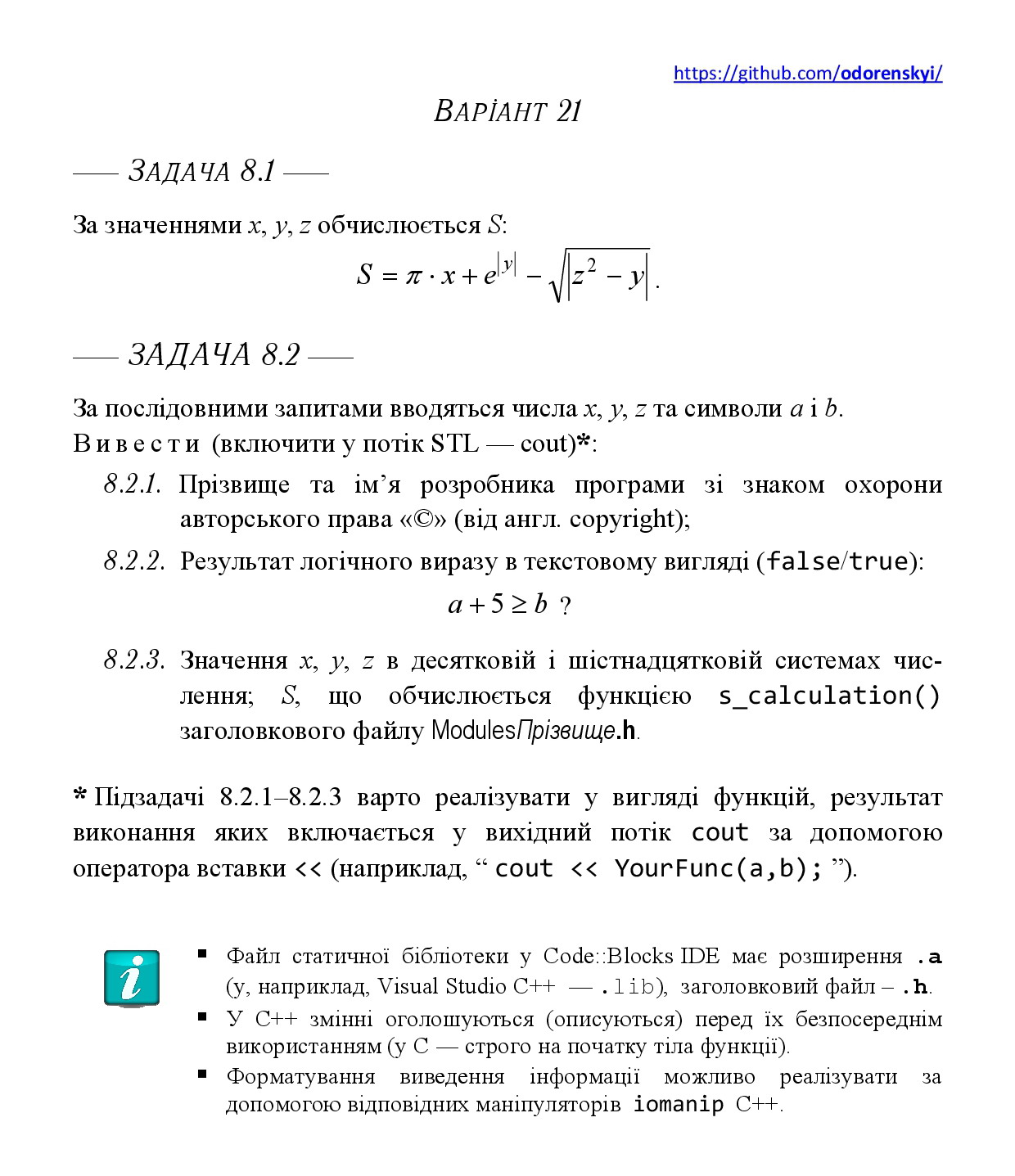
**Тема:** Реалізація статичних бібліотек модулів лінійних обчислювальних процесів

**Завдання до лабораторної роботи:**

1. Реалізувати статичну бібліотеку модулів libModulesПрізвище C / C++, яка містить функцію розв’язування задачі 8.1.

2. Реалізувати програмне забезпечення розв’язування задачі 8.2 — консольний застосунок.

**Варіант 21**



**Задача 8.1**

Вхідні дані: x, y, z – цілочисельний тип int

Вихідні дані: S – дробовий тип float

Прототип функції: float s\_calculation(int, int, int);

Для реалізації функції s\_calculation буде використовуватися бібліотека cmath. Для використання числа Пі знадобиться константа M\_PI, а для e – M\_E. Для піднесення e до степеня |y| буде використана функція pow(). Для знаходження модуля числа – abs(). Для знаходження квадратного кореня – sqrt().

*Результат виконання тестового драйвера:*

Тест-кейс №1

Вхідні дані:

x: 5

y: -27

z: -21

Результат: 5.32048e+11

Статус тест-кейса: passed

Тест-кейс №2

Вхідні дані:

x: -20

y: 28

z: -24

Результат: 1.44626e+12

Статус тест-кейса: passed

Тест-кейс №3

Вхідні дані:

x: 27

y: 21

z: 15

Результат: 1.31882e+09

Статус тест-кейса: passed

Тест-кейс №4

Вхідні дані:

x: 15

y: 4

z: 14

Результат: 87.8656

Статус тест-кейса: passed

Тест-кейс №5

Вхідні дані:

x: 27

y: -7

z: -25

Результат: 1156.32

Статус тест-кейса: passed

Тест-кейс №6

Вхідні дані:

x: 16

y: 30

z: 16

Результат: 1.06865e+13

Статус тест-кейса: passed

Тест-кейс №7

Вхідні дані:

x: 28

y: -28

z: 16

Результат: 1.44626e+12

Статус тест-кейса: passed

Тест-кейс №8

Вхідні дані:

x: 10

y: -17

z: -18

Результат: 2.4155e+07

Статус тест-кейса: passed

Тест-кейс №9

Вхідні дані:

x: 20

y: -17

z: 30

Результат: 2.4155e+07

Статус тест-кейса: passed

Тест-кейс №10

Вхідні дані:

x: -25

y: 19

z: 23

Результат: 1.78482e+08

Статус тест-кейса: passed

*Лістинг проекту ModulesHrebeniuk:*

#include <cmath>

float s\_calculation(int x, int y, int z) {

float S = M\_PI \* x + pow(M\_E, abs(y)) - sqrt(abs(pow(z, 2) - y));

return S;

}

*Лістинг проекту TestDriver:*

#include <iostream>

#include "ModulesHrebeniuk.h"

using namespace std;

void test\_tc(int test\_case, int x, int y, int z, float expected) {

cout << "Тест-кейс №" << test\_case << endl;

cout << "Вхідні дані:\nx: " << x << "\ny: " << y << "\nz: " << z << endl;

float result = s\_calculation(x, y, z);

cout << "Результат: " << result << endl;

if (result == expected) {

cout << "Статус тест-кейса: passed" << endl;

} else {

cout << "Статус тест-кейса: failed" << endl;

}

cout << endl;

}

int main() {

system("chcp 1251 & cls");

test\_tc(1, 5, -27, -21, 532048248832);

test\_tc(2, -20, 28, -24, 1446257098752);

test\_tc(3, 27, 21, 15, 1318815744);

test\_tc(4, 15, 4, 14, 87.865631103515625);

test\_tc(5, 27, -7, -25, 1156.3165283203125);

test\_tc(6, 16, 30, 16, 10686474223616);

test\_tc(7, 28, -28, 16, 1446257098752);

test\_tc(8, 10, -17, -18, 24154966);

test\_tc(9, 20, -17, 30, 24154986);

test\_tc(10, -25, 19, 23, 178482192);

return 0;

}

**Задача 8.2**

Вхідні дані: x, y, z – цілочисельний тип int; a і b – символьний тип char

Вихідні дані: результат логічного виразу у вигляді true/false; значення x, y, z в десятковій і шістнадцятковій системах числення; S, що обчисл. функціює s\_calculation().

Перший підпункт цієї задачі я планую відокремити в заголовковий файл, щоб в подальшому використовувати власний копірайт в майбутніх лабораторних роботах без необхідності реалізовувати однаковий програмний код. При реалізації другого підпункту знадобиться використання маніпулятора boolalpha, щоб у потік cout включити саме результат у вигляді false/true. У третьому підпункті використаю маніпулятор hex для виведення значення x, y, z в шістнадцятковій системі числення.

*Лістинг проекту Hrebeniuk\_task:*

#include <iostream>

#include "Copyright.h"

#include "ModulesHrebeniuk.h"

using namespace std;

bool getBoolResult(char a, char b)

{

cout << boolalpha;

return (a + 5 >= b);

}

int getHexadecimal(int number)

{

cout << hex;

return number;

}

int main()

{

cout << printCopyright() << endl;

int x;

cout << "Уведіть x: ";

cin >> x;

int y;

cout << "Уведіть y: ";

cin >> y;

int z;

cout << "Уведіть z: ";

cin >> z;

char a;

cout << "Уведіть a: ";

cin >> a;

char b;

cout << "Уведіть b: ";

cin >> b;

cout << "\n" << "Результат логічного виразу: " << getBoolResult(a, b) << endl;

cout << "\nx (в десятковій): " << x << "\ny (в десятковій): " << y << "\nz (в десятковій): " << z << endl;

cout << "\nx (в шістнадцятковій): " << getHexadecimal(x) << "\ny (в шістнадцятковій): " << getHexadecimal(y) << "\nz (в шістнадцятковій): " << getHexadecimal(z) << endl;

cout << "\nS = " << s\_calculation(x, y, z) << endl;

return 0;

}

**Висновок**

Виконуючи цю лабораторну роботу я набув знань та практичних навичок застосування методології модульного програмування, реалізації метода функціональної декомпозиції задач, метода модульного тестування та роботи з мовою програмування C++.

Лабораторна робота №8 унікальна тим, що потрібно використовувати Git-репозиторій з фіксацією багатьох пунктів, що виконуюються згідно з методичних рекомендацій. Саме завдяки цьому я набув додаткових знань у роботі з Git та GitHub, умінню видаляти помилкові коміти та коректно складати опис кожного зробленого коміту.

Завдання 8.1 починалось з аналізу та постановки задачі. Потрібно було виконати аналіз вимог, спроектувати архітектуру та програмний модуль розв’язання цієї задачі. Одержані результати задокументувати й включити до звіту. Після цього було розроблено тест-сьют модульного тестування модуля ModulesHrebeniuk, а саме функції s\_calculation(). При реалізації функції мовою програмування C++ я навчився працювати зі статичними бібліотеками, їх створенням та подальшим підключенням до інших проектів. Найбільше сподобався новий підхід – тестовий драйвер. Саме в цьому проекті була змогу на практиці підключити бібліотеку та попрацювати з директивами препроцесора. Для роботи консолі з кирилицею використовувалось кодування UTF-8, з яким я навчився працювати в лабораторній роботі №7.

Завдання 8.2 теж почалось з аналізу та постановки задачі, аналізу вимог, проектування архітектури. Результати були задокументовані й включені до звіту. Було розроблено тест-сьют для виконання вже системного тестування ПЗ. При реалізації задачі мовою програмування була можливість набути навичок з використанням потокового введення й виведення інформації. Був розроблений заголовковий файл Copyright.h, що буде використаний в подальших лабораторних роботах з метою повідомлення користувача про розробника ПЗ. Також, як і в попередньому завданні потрібно було викликати функцію s\_calculation з модулю ModulesHrebeniuk. Були здійснені ті самі кроки підключення бібліотеки, що й в завдання 8.1 при розробці тестового драйвера. При виконанні тестування, використовуючи тест-сьют для завдання 8.2, помилок не знайдено, ПЗ працює коректно.

Завдяки цій лабораторній роботі я покращив свої знання та практичні навички у використанні Git, роботи з GitHub, проектуванні та розробці модульних проектів, розробці тестових драйверів та дізнався більше про роботу лінкера, препроцесора. Мені сподобалось виконувати цю роботу, я знайшов для себе багато нової інформації та виконувати завдання було цікаво. Завдяки покроковому опису роботи в методичних рекомендаціях виникало мало запитань щодо того, як саме зробити певне завдання.

**Контрольні запитання і завдання**

**1.** *У чому полягають мета й задачі процесів проектування програмного забезпечення відповідно до міжнародного стандарту ISO/IEC 12207:2008?*

- Процес проектування архітектури

Визначення структури та організації системи

- Процес детального проектування

Розробка детальних специфікацій компонентів системи

- Процес конструювання

Реалізація компонентів системи на основі детальних специфікацій

- Процес комплексування ПЗ

Об'єднання різних компонентів та підсистем для створення готового програмного продукту

**2.** *У чому полягає сутність метода функціональної декомпозиції і під час якого процесу життєвого циклу ПЗ він застосовується?*

Метод функціональної декомпозиції полягає у розбитті системи на менші функціональні блоки (модулі) для подальшої деталізації та реалізації. Він застосовується під час процесу аналізу та проектування системи в життєвому циклі програмного забезпечення.

**3.** *Перелічіть та поясніть стадії утворення ПЗ: процес трансляції вихідного коду .срр у виконуваний .ехе?*

1) Препроцесування - обробка директив препроцесора

2) Компіляція - перетворення вихідного коду на машинний код

3) Лінкування - об'єднання об'єктних файлів та бібліотек виконуваних програм

4) Створення виконуваного файлу - формування виконуваного .ехе файлу

**4.** *Які символи С/С++ називають ключовими (escape-послідовності) та яке їх призначення кожного з них?*

|  |  |  |
| --- | --- | --- |
| **ESC-послідовність** | **Код** | **Дія** |
| \a | 0x7 | Звуковий сигнал |
| \n | 0xA | Перехід на новий рядок |
| \f | 0xC | Перегортання сторінки |
| \r | 0xD | Повернення курсора |
| \t | 0x9 | Горизонтальна табуляція |
| \v | 0xB | Вертикальна табуляція |
| \b | 0x8 | Повернення курсора на одну позицію назад |
| \\ | 0x5C | Зворотна коса риса |
| \’ | 0x27 | Апостроф |
| \” | 0x22 | Подвійні лапки |
| \? | 0x77 | Знак питання |

**5.** *Яке призначення директиви препроцесора #include С/С++?*

Директива препроцесора #include у мові програмування С/С++ використовується для включення вмісту файлів заголовків у вихідний код програми. Це дозволяє використовувати функції, константи та інші елементи, які описані у включених файлах.

**6.** *Що міститься у заголовкому файлі С++ та яка директива препроцесора його використовує?*

У заголовковому файлі С++ зазвичай містяться прототипи функцій, декларації змінних та інших об'єктів. Для включення заголовкових файлів в програму використовується директива препроцесора #include.

**7.** *Яким є синтаксис запису числових, символьних, рядкових літералів (літеральних констант) у С/С++?*

int a = 10; // запис числових літералів

char a = 'a'; // запис символьних літералів

const char\* a = "Hello, world!"; // запис рядкових літералів

**8.** *Сформулюйте правило побудови ідентифікаторів (в тому числі щодо його довжини, регістра) С/С++.*

Ідентифікатор може складатися з літер, цифр та символу підкреслення і починатися з літери або символу підкреслення. Довжина ідентифікатора обмежена певним компілятором. Регістр літер у ідентифікаторі важливий, тобто abc і ABC будуть різними ідентифікаторами.

**9.** *Перелічіть вбудовані числові й символьні типи даних С/С++, їх розмірність (в байтах), діапазон значень.*

|  |  |  |
| --- | --- | --- |
| **Тип даних** | **Байт** | **Діапазон значень** |
| char | 1 | 0 / 255 |
| short int | 2 | -32 768 / 32 767 |
| unsigned short int | 2 | 0 / 65 535 |
| int | 4 | -2 147 483 648 / 2 147 483 647 |
| unsigned int | 4 | 0 / 4 294 967 295 |
| long int | 4 | -2 147 483 648 / 2 147 483 647 |
| unsigned long int | 4 | 0 / 4 294 967 295 |
| float | 4 | -2 147 483 648.0 / 2 147 483 647.0 |
| long float | 8 | -9 223 372 036 854 775 808.0 / 9 223 372 036 854 775 807.0 |
| double | 8 | -9 223 372 036 854 775 808.0 / 9 223 372 036 854 775 807.0 |

**10.** *Яким є синтаксис оголошення змінних у С/С++? Чи можливо одночасно оголосити й ініціалізувати змінну за допомогою константного літерала або виразу?*

Синтаксис: тип\_даних ім'я\_змінної;

тип\_даних ім'я\_змінної = значення;

**11.** *Що розуміють під інтерфейсом функції (модуля) і який синтаксис його запису мовою програмування С/С++?*

Під інтерфейсом функції (модуля) розуміють набір зовнішніх функцій, їх параметрів та тип даних, що вони повертають. У мові програмування С++ це відображається у використанні прототипів функцій.

Синтаксис прототипу функції: тип\_даних ім'я\_функції(параметри);

**12.** *За допомогою якої функції заголовкового файлу clocale/locale.h здійснюють локалізацію кодування символів?*

Локалізацію кодування символів здійснюється за допомогою функції setlocale().

**13.** *Сформулюйте порядок (алгоритм) створення статичної бібліотеки у Code::Blocks IDE та використання функцій із неї під час модульного програмування.*

1) Створити проект статичної бібліотеки

2) Якщо мова бібліотеки C++, то змінити розширення файлу вихідного коду на .cpp

3) Скомпілювати проект статичної бібліотеки, в результаті чого створюється файл статичної бібліотеки назва\_проекту.a

4) Створити проект заголовкового файлу та описати прототипи функцій, що були розроблені у статичній бібліотеці

5) Щоб використати функцію з бібліотеки, потрібно налаштувати опції компілятора “Build options”. Linker ― шлях до файла статичної бібліотеки назва\_бібліотеки,.а Compiler ― шлях до заголовкового файла назва\_бібліотеки.h

**14.** *У якому заголовковому файлі стандартної бібліотеки визначені математичні функції мови програмування С, а у якому – С++?*

Математичні функції мови програмування С визначені у заголовковому файлі math.h, а у мові С++ вони доступні як у cmath (стандартна бібліотека С++) так і у math.h.

**15.** *Яким чином у С++ реалізовується потокове виведення даних з та без форматування?*

У C++ потокове виведення даних з форматуванням здійснюється за допомогою оператора <<, а без форматування - за допомогою функції write() або put().

**16.** *За допомогою якої функції С та якого об’єкта С++ в просторі імен std можливо реалізувати консольне введення інформації?*

Консольне введення інформації у мові програмування С можна реалізувати за допомогою функції scanf(), а у мові С++ - за допомогою об'єкта cin з простору імен std.

**17.** *Яке призначення маніпуляторів boolalpha, hex, fixed, endl?*

Маніпулятори boolalpha, hex, fixed, endl використовуються для форматування потоків у виведенні даних. boolalpha відображає значення bool у вигляді "true" або "false", hex встановлює шістнадцятковий формат виведення, fixed встановлює фіксований формат виведення для чисел з плаваючою точкою, а endl додає символ нового рядка і змушує потік виведення буферизувати дані.

**18.** *Поясніть сутність і призначення кожного технічного процесу (стадії) конструювання ПЗ: препроцесинг (preprocessing), компіляція (compiling), зв’язування (linking).*

- Препроцесинг

В цій стадії препроцесор обробляє інструкції, що починаються з символу #, такі як директиви #include для включення заголовкових файлів, #define для визначення макросів тощо.

- Компіляція

У цьому етапі компілятор перетворює вихідний код програми у машинний код або у код асемблера, який вже можна виконувати комп'ютером.

- Зв’язування

На останньому етапі компіляції, компонувальник об'єднує окремі об'єктні файли у виконуваний файл або бібліотеку.

**19.** *Яким чином реалізовується тестування модуля (Unit testing)?*

Тестування модуля (Unit testing) полягає у відокремленому тестуванні окремих модулів. Для цього пишуться тестові кейси, які перевіряють коректність роботи кожного модуля незалежно від інших частин програми. Потім, на основі тест-сьютів виконуються тестування тестовим драйвером, що було розроблено саме під це ПЗ.

**20.** *В межах якого процесу життєвого циклу ПЗ розробляється набір контрольних прикладів (тест-кейсів)?*

Набір контрольних прикладів (тест-кейсів) розробляється під час процесу тестування (testing) у життєвому циклі програмного забезпечення.