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

ВИКОНАВ

студент академічної групи \_\_\_\_\_\_\_\_\_\_\_\_\_\_ М.Л. Іванов

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

\_\_\_\_\_\_\_\_\_ Павло УСІК

Кропивницький – 2022

**Мета** полягає у набутті ґрунтовних вмінь і практичних навичок застосування теоретичних положень методології модульного програмування, реалізації метода функціональної декомпозиції задач, метода модульного (блочного) тестування, представлення мовою програмування С++ даних скалярних типів, арифметичних і логічних операцій, потокового введення й виведення інформації, розроблення програмних модулів та засобів у кросплатформовому середовищі Code::Blocks (GNU GCC Compiler).

**Завдання 1**

Створити модуль для обчислення виразу

**Завдання 2**

За полсідовними запитами вводяться числа x, y, z та символи a i b. a)Вивести прізвище та ім'я розробника програми із знаком охорони авторського права b)Результат логічного виразу а <= b - 32 (0/1) c)Вивести значення x, y, z в десятковій та 16-ковій системах ичслення та S, що обчислюється за функцією s\_calculation() заголовкового файлу ModulesIvanov.h

**Варіант № 18**

**ХІД РОБОТИ**

**Умова задачі 8.1**

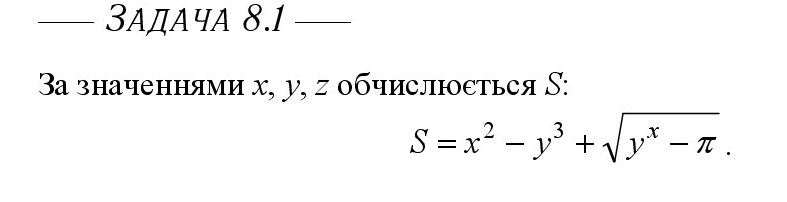


Рисунок 1.1 – Умова задачі 8.1

### **Строга постановка задачі:**

Вхідні дані: x, y, z - дійсні знакові числа;

Вихідні дані: S - дійсне число (результат розрахунків);

### **Проектування програмного модуля:**

Модуль ModulesIvanov, що складається з заголовкового файлу (містить оголошення прототипу функції) та файлу вихідного коду (в которому пристуня реалізація функції c\_calculation).

Функція c\_calculation приймає два аргументи в якості вхідних даних, та повертає значення, розраховане за виразом, зазначеним в умові.

Тестовий драйвер TestDriver - функція, що містить в собі:

* Значення вхідних змінних
* Значення результату
* Перевірка на істинність

Для вихідного коду, див. ДОДАТОК Г.

**Результати модульного тестуваня**

TC\_01:|x = 5 y = 3 | - result: 13.487363 - passed

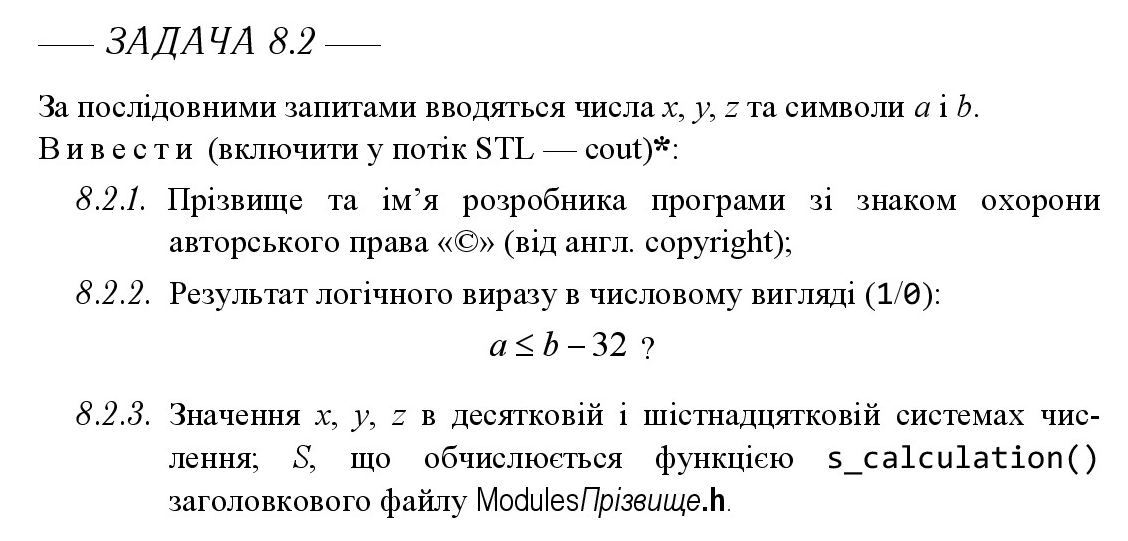
TC\_20:|x = 6 y = 2 | - result: 35.801180 - passed

TC\_30:|x = 3 y = 5 | - result: -104.961050 - passed

TC\_40:|x = 7 y = 2 | - result: 52.174006 - passed

TC\_50:|x = 6 y = 3 | - result: 35.941760 - passed

**Умова задачі 8.2**



### **Строга постановка задачі**:

Вхідні дані:

* x, y, z, a, b – раціональні числа;

Вихідні дані:

* Ім’я розробника з символом copyright;
* Результат логічного виразу: a <= b - 32;
* x, y, z в десятковій та шістнадцятковій системі числення;
* Результат функції c\_calculation з модуля ModulesIvanov.cpp

ВИСНОВКИ

Набуто ґрунтовних вмінь і практичних навичок застосування теоретичних положень методології модульного програмування, реалізації метода функціональної декомпозиції задач, метода модульного (блочного) тестування, представлення мовою програмування С++ даних скалярних типів, арифметичних і логічних операцій, потокового введення й виведення інформації, розроблення програмних модулів та засобів у кросплатформовому середовищі Code::Blocks (GNU GCC Compiler).

При виконанні завдань 8.1 та 8.2 був задіяний метод висхідного проектування програмного забезпечення, що полягав у початковій реалізації менш значимих (або допоміжних) функцій нижнього рівня. Такою функцією виявилась c\_calculation, з модуля ModulesIvanov, розробленим на етапі задачі 8.1. До даного застосовувались всі правила побудови файлів вихідного коду та заголовкового. Надалі, модуль ModulesIvanov використовувався в main функціях обох задач.

Заголовковий файл .h формату складається з препроцессорних макропідстановок та оголошених прототипів функцій. #ifndef - макрос, що перевіряє визначеність файлу. Якщо файл не визначено - виконуються #define макропідстановка, що задає файлу визначеності. Внаслідок цього, виконується наступний набір операторів, а в кінці вихідного файлу, перевірка на визначеність файлу припиняється макросом #endif. Інакше кажучи - дані макроси дозволяють, або забороняють компіляцію виділеної ними частини коду.

Файл вихідних кодів формату .cpp містить ініціалізацію функцій, раніше оголошених в .h файлі. Потребує підключення директиви .h файлу через макропідстановку #include для належної реалізації функцій.

Також, необхідно виділити увагу бібліотеці <cmath>, що використовувалась для операцій всередині c\_calculation. Базова математична стала ПІ та математичні функції pow і sqrt були реалізовані завдяки їй.

Завдання 8.2 потребує використання відлагодженого модуля ModulesStriuk. Проект 8.2 користується як процедурною, так і модульною парадигмою програмування (дана потреба викликана умовою задачі). Завдання націлене на детальний огляд роботи потокового виведення за допомогою cout (з простору імен std), функціонування маніпуляторів форматованого виведення та запису виклику функції безпосередньо в потік виведення.

Важливою частиною лабораторної роботи було використання системи контролю версій файлів та спільнох роботи Git. Ідея системи контролю версій полягає у зручному зберіганні різних версій проекту в одному місці. Замість зберігання копій одного й того ж проекту, реєструються зміни в проекті, що дозволяє за потреби відкатити проект до потрібної версії (в даному контексті “версія” описує стан проект після певної послідовності змін). У випадку цієї лабораторної роботи проект зберігається на сайті Github. Для відсилання змін у проекті на сайт подрібно за допомогою застосунку git-scm клонувати

Реалізація тестового модуля визначена в main.cpp проекту TestDriver. Основною метою створення даного модуля була автоматизація процесу Unit Testing, яка дозволяє не перезапускаючи застосунок продовжувати модульне тестування

Модульне тестування, відоме як Unit-тестування, використовується для автоматизованого тестування модулів шляхом порівняння значення, що повертається функцією з модуля, яка приймає набір еталонних вхідних даних (аргументів), з еталонним результатом для відповідного набору. Важливою ідеєю юніт-тестування є така: якщо хочаб один з тестів провалився, після відлагодження модуля потрібно проводити повторне проведення тестів, що йшли перед провальним тестом, до моменту, коли усі тести проходять успішно.

Працювати з GitHub значно легше, ніж із Гугл Диск. Скарг на документацію та методичні вказівки не виникало, саме завдяки ним лабораторна робота була виконана

ДОДАТОК А

Test Suite 8.1

|  |  |
| --- | --- |
| Назва тестового набору  Test Suite Description | TestSuite |
| Назва проекта / ПЗ  Name of Project / Software | ModulesIvanov.exe |
| Рівень тестування  Level of Testing | системний / System Testing |
| Автор тест-сьюта  Test Suite Author | Ivanov Max |
| Виконавець  Implementer | Ivanov Max |

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест-кейса / Test Case ID | Дії (кроки) /  Action  (Test Steps) | Очікуваний  результат /  Expected Result | Результат тестування (пройшов/не вдалося/ заблокований) /  Test Result (passed/failed/ blocked) |
| 8.1.A | Вхідні дані:  X = 5  Y = 3  Z =1 | S = 13.4874 | PASSED |
| 8.1.B | Вхідні дані:  X = 6  Y = 2  Z =7 | S =35.8012 | PASSED |
| 8.1.C | Вхідні дані:  X = 3  Y = 5  Z = 1 | S = -104.961 | PASSED |
| 8.1.D | Вхідні дані:  X = 7  Y = 2  Z = 1 | S = 52.174 | PASSED |
| 8.1.E | Вхідні дані:  X = 6  Y = 3  Z = 3 | S = 35.9418 | PASSED |

ДОДАТОК Б

Test Suite 8.2

|  |  |
| --- | --- |
| Назва тестового набору  Test Suite Description | TestSuite |
| Назва проекта / ПЗ  Name of Project / Software | ModulesIvanov.exe |
| Рівень тестування  Level of Testing | системний / System Testing |
| Автор тест-сьюта  Test Suite Author | Ivanov Max |
| Виконавець  Implementer | Ivanov Max |

|  |  |  |  |
| --- | --- | --- | --- |
| Ід-р тест-кейса / Test Case ID | Дії (кроки) /  Action  (Test Steps) | Очікуваний  результат /  Expected Result | Результат тестування (пройшов/не вдалося/ заблокований) /  Test Result (passed/failed/ blocked) |
| 8.2.A | Вхідні дані:  X = 5  Y = 3  Z =1  A = 25  B = 3 | A <= B – 32 is false  X10 = 5 X16 = 05  Y10 = 3 Y16 = 03  Z10 = 1 Z16 = 01  S = 13.4874 | PASSED |
| 8.2.B | Вхідні дані:  X = 6  Y = 2  Z =7  A = 51  B = 32 | A <= B – 32 is false  X10 = 6 X16 = 06  Y10 = 2 Y16 = 02  Z10 = 7 Z16 = 07  S = 35.8012 | PASSED |
| 8.2.C | Вхідні дані:  X = 3  Y = 5  Z = 1  A = 3  B = 54 | A <= B – 32 is true  X10 = 3 X16 = 03  Y10 = 5 Y16 = 05  Z10 = 1 Z16 = 01  S = -104.961 | PASSED |
| 8.2.D | Вхідні дані:  X = 7  Y = 2  Z = 1  A = 29  B = 33 | A <= B – 32 is false  X10 = 7 X16 = 07  Y10 = 2 Y16 = 02  Z10 = 1 Z16 = 01  S = 52.174 | PASSED |
| 8.2.E | Вхідні дані:  X = 6  Y = 3  Z = 3  A = 100  B = 132 | A <= B – 32 is true  X10 = 6 X16 = 06  Y10 = 3 Y16 = 03  Z10 = 3 Z16 = 03  S = 35.9418 | PASSED |

ДОДАТОК В

Лістинг ModulesIvanov.exe

double c\_calculation(double x, double y)

{

double S;

S = pow(x, 2) - pow(y, 3) + sqrt(pow(y, x) - M\_PI);

return S;

}

ДОДАТОК Г

Лістинг TestDriver.exe

#include <iostream>

#include <iomanip>

#include <cmath>

#include "ModulesIvanov.h"

using namespace std;

int main()

{

double input[5][2] = {5, 3,

6, 2,

3, 5,

7, 2,

6, 3};

double result[5] = {13.487363,

35.801180,

-104.961050,

52.174006,

35.941760};

for (int i = 0; i < 5; i++) {

cout << resetiosflags(ios::fixed) << "TC\_" << setfill('0') << setw(2) << i+1 << ":|" << setfill(' ');

cout << "x = " << setw(5) << left << input[i][0];

cout << "y = " << setw(5) << left << input[i][1];

cout << fixed << "| - result: " << setw(10) << c\_calculation(input[i][0], input[i][1]);

cout << left << " - " << ((round(c\_calculation(input[i][0], input[i][1])\*1000000)/1000000.0 == result[i]) ? "passed" : "failed") << endl;

}

return 0;

}

ДОДАТОК Г

Лістинг Ivanov-task

#include <iostream>

#include "ModulesIvanov.h"

#include <locale>

#include <locale.h>

#include <windows.h>

using namespace std;

void Greetings()

{

cout << "|====================|" << endl

<< "|Max Ivanov ";

wcout << L"\x00A9 |";

cout << endl

<< "|====================|" << endl;

}

void Inequality(char a, char b)

{

if (a+1 >= b)

cout << "a+1 >= b is true" << endl;

else

cout << "a+1 >= b is false" << endl;

}

double Change(int value)

{

int k = 1, b = 0;

while (value)

{

b += value%16\*k;

value/=16;

k\*=10;

}

return b;

}

void ChangeOut(int val)

{

cout << val << "x10: " << val << " " << val << "x16: 0" << Change(val) << endl;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int x, y, z;

char a, b;

cout << "Уведіть x: ";

cin >> x;

cout << "Уведіть y: ";

cin >> y;

cout << "Уведіть z: ";

cin >> z;

cout << "Уведіть a: ";

cin >> a;

cout << "Уведіть b: ";

cin >> b;

Greetings();

Inequality(a,b);

ChangeOut(x);

ChangeOut(y);

ChangeOut(z);

cout << c\_calculation(x,y);

return 0;

}