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/

ВИКОНАВ
студент академічної групи КІ-23
Чепіль В.О.

ПЕРЕВІРИВ
ст. викладач кафедри кібербезпеки
та програмного забезпечення
Дрєєва Г. М.

Кропивницький – 2024

Тема: Реалізація статичних бібліотек модулів лінійних обчислювальних процесів

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. Завантажити власний Git-репозиторій https://github.com/odorenskyi/student-name (в \Lab8\tasks містяться умови задач 8.1–8.2).

2. У \Lab8 заповнити файл README.md, створити теки prj, Software, TestSuite, Report; отриманий вміст теки \Lab8 завантажити до Git-репозиторію https://github.com/odorenskyi/student-name; надалі здійснювати означену дію (git add, git commit, git push) за позначкою .

3. До звіту з лабораторної роботи (далі — звіт) включити мету роботи, номер варіанту, завдання.

4. Здійснити аналіз і постановку задачі 8.1 (див. \Lab8\tasks).

5. Виконати аналіз вимог, проектування архітектури, детальне проектування програмного модуля розв’язування задачі 8.1; одержані артефакти задокументувати й включити до звіту.

6. Розробити набір контрольних прикладів до задачі 8.1 задля виконання модульного тестування (Unit testing) модулів С++; отримані тест-сьюти належно задокументувати, зберегти у \Lab8\TestSuite та включити до звіту.

7. В Code::Blocks IDE створити проект статичної бібліотеки ModulesПрізвище, зберегти його у \Lab8\prj, розширення файлу вихідного коду (main.c) змінити на cpp. 8. На основі результатів проектування модуля, реалізувати мовою програмування С++ функцію s_calculation, яка за належним інтерфейсом реалізовує розв’язування задачі 8.1.

9. Скомпілювати проект статичної бібліотеки ModulesПрізвище (Build → Build або Ctrl+F9) ;

- в результаті компіляції з файлу ModulesПрізвище.cpp створиться libModulesПрізвище.a — файл статичної бібліотеки (за замовчуванням у теці \obj).

10. В Code::Blocks IDE створити проект заголовкового файлу ModulesПрізвище в \Lab8\prj та описати в ньому прототип функції s_calculation (скопіювати з проекта статичної бібліотеки ModulesПрізвище заголовок функції й описати як прототип), зберегти проект ;

- у \prj створиться ModulesПрізвище.h — заголовковий файл C++.

11. В Code::Blocks IDE у \prj створити проект консольного додатка C++, іменувати його TestDriver.

12. Реалізувати тестовий драйвер для виконання розроблених тестових наборів (\Lab8\TestSuite) і за його допомогою виконати модульне тестування функції s_calculation зі статичної бібліотеки libModulesПрізвище.a ;

- для підключення створеної бібліотеки слід використати директиву препроцесора #include "ModulesПрізвище.h" та налаштувати опції компілятора Build options…: Linker — шлях до файла статичної бібліотеки libModulesПрізвище.a, Compiler — шлях до заголовкового файла ModulesПрізвище.h);

- рекомендовано реалізувати протоколювання процесу тестування тестовим драйвером: виведення у консоль вхідних даних (аргументів функції, яка тестується), отриманий результат та статус тест-кейса (passed або failed).

- у випадку негативного результату тестування модуля (невиконання хоча б одного тест-кейса) виконати відлагодження проекта статичної бібліотеки ModulesПрізвище (відповідної функції), процес модульного тестування повторити.

13. Результати тестування s_calculation зі статичної бібліотеки libModulesПрізвище.a тестовим драйвером задокументувати (скопіювати з консольного вікна застосунку текст протоколу тестування) та включити до звіту;

14. Вихідний код (текст) проектів ModulesПрізвище та TestDriver включити до звіту як додатки.

15. Здійснити аналіз і постановку задачі 8.2 (див. \Lab8\tasks).

16. Виконати аналіз вимог, проектування архітектури, детальне проектування програмного забезпечення розв’язування задачі 8.2; отримані результати задокументувати й включити до звіту.

17. Розробити тест-сьют для виконання системного тестування ПЗ розв’язування задачі 8.2, який повинен складатись не менш як з п’яти тест-кейсів; файл тестового набору зберегти у \TestSuite .

18. В Code::Blocks IDE створити проект консольного додатка Прізвище_task у теці \prj.

19. Мовою програмування C/C++ реалізувати результати проектування програмного забезпечення розв’язування задачі 8.2, скомпілювати проект (Build → Build або Ctrl+F9) .

20. Відповідно до вимог міжнародного стандарту ISO/IEC 12207 здійснити системне тестування ПЗ Прізвище_task.exe за допомогоютест-сьюту із \TestSuite; результати тестування задокументувати ;

- у випадку негативного результату тестування Прізвище_task.exe виконати відлагодження проекта, системне тестування повторити.

21. Отриманий тестовий артефакт (тестовий набір у теці Lab8\TestSuite) включити до звіту як додаток.

22. Консольний додаток Прізвище_task.exe скопіювати у \Software .

23. Проаналізувати хід виконання лабораторних завдань і самостійно одержані результати, на основі чого сформулювати обґрунтовані висновки з виконаної лабораторної роботи, викласти їх обсягом не менше 2 сторінок машинного (комп’ютерного) тексту та включити до звіту;

- у висновках (підсумках) варто також зазначити особисті враження від процесу виконання завдань лабораторної роботи, аргументовано викласти вмотивовані пропозиції, обґрунтовані зауваження, конструктивну критику, рекомендації тощо.

24. Підготувати й зберегти у \Lab8\Report звіт про виконання лабораторної роботи, оформлений згідно з ДСТУ 3008:2015 “Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання”, та зі змістом, визначеним цим порядком виконання лабораторної роботи .

25. Представити до захисту звіт з виконаної лабораторної роботи і проект у Git-репозиторії https://github.com/odorenskyi/student_name.

Варіант 19

Лістинг ModulesChepil.h:

```
#ifndef MODULESCHEPIL_H_INCLUDED
#define MODULESCHEPIL_H_INCLUDED

void s_calculation(int, int, int);

#endif // MODULESCHEPIL_H_INCLUDED
```

Лістинг libModulesChepil.a:

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

using namespace std;

void s_calculation(int x, int y, int z)
{
   system("chcp 65001 > nul");
   const double PI = 3.141592653589793238;
   float S;
   S = pow(2 * z + 1, x) - sqrt(abs(y - (1 / 2) * z)) + z + PI;
   cout << "S: " << setprecision(1) << fixed << S << endl;
}
```

Лістинг TestDriver:

```
#include <iostream>
#include "ModulesChepil.h"

using namespace std;

int main()
{
   system("chcp 65001 & cls");
   cout << "TC_01: ";
   s_calculation(5, 10, 2);
   cout << "TC_02: ";
   s_calculation(5, 12, 7);
   cout << "TC_03: ";
   s_calculation(2, 5, 10);
   cout << "TC_04: ";
   s_calculation(10, 10, 10);
   cout << "TC_05: ";
   s_calculation(0, 0, 0);
   cout << "TC_06: ";
   s_calculation(-10, 5, -12);
   cout << "TC_07: ";
```

```
        s_calculation(0, 0, -23);
        cout << "TC_08: ";
        s_calculation(-2, 1, 87);
        cout << "TC_09: ";
        s_calculation(11, -9, 0);
        cout << "TC_10: ";
        s_calculation(7, 129, 5);
        return 0;
    }
```

Результати тестування s_calculation:

TC_01: S: 3127.0

TC_02: S: 759381.7

TC_03: S: 451.9

TC_04: S: 16679880884224.0

TC_05: S: 4.1

TC_06: S: -11.1

TC_07: S: -18.9

TC_08: S: 89.1

TC_09: S: 1.1

TC_10: S: 19487168.0

Лістинг Chepil_task:

```
#include <iostream>
#include "ModulesChepil.h"

void task_8_1();
void task_8_2(int a, int b);

using namespace std;

int main()
{
   int x, y, z, a, b;
   system("chcp 65001 & cls");
   cout << "Введіть значення x:";
   cin >> x;
```

```
    cout << "Введіть значення y:";
    cin >> y;
    cout << "Введіть значення z:";
    cin >> z;
    cout << "Введіть значення a:";
    cin >> a;
    cout << "Введіть значення b:";
    cin >> b;
    cout << endl;
    task_8_1();
    task_8_2(a,b);
    s_calculation(x, y, z);
    cout << "x у десятковій формі: " << x << endl;
    cout << "y у десятковій формі: " << y << endl;
    cout << "z у десятковій формі: " << z << endl;
    cout << "x у шістнадцятковій формі: " << hex << x << endl;
    cout << "y у шістнадцятковій формі: " << hex << y << endl;
    cout << "z у шістнадцятковій формі: " << hex << z << endl;
    return 0;
}
void task_8_1(){
    cout << " Чепіль Вадим ©" << endl;
}

void task_8_2(int a, int b){
    bool result;
    cout << boolalpha;
    result = a + 1 >= b;
    cout << result << endl;
}
```

Висновок: під час виконання завдання 8.1 було розроблено статичну бібліотеку, яка включає функцію s_calculation, а також створений відповідний заголовковий файл із прототипом цієї функції.

Після чого, я проаналізував вимоги до програмного забезпечення:

Вх: x,y,z - змінні типу int

Вих: S - змінна типу float, у якій було обраховано зазначену формулу

Далі було створено TestSuite_lab_8_1 для модульного тестування, були протестовані 10 тест кейсів через консольний застосунок TestDriver.

У завданні 8.2 я створив консольний застосунок, який містить вирішення задачі 8.2(8.2.1-8.2.3), було проаналізовано вимоги для програмного забезпечення:

Вх: x,y, z, a, b – п’ять змінних типу int

Вих:

8.2.1. – Чепіль Вадим©

8.2.2. - Результат даного виразу (true or false)

8.2.3. - Значення x,y,z в десятковій та шістнадцятковій системах числення. S, за s_calculation()

Вимоги: 8.2.1.-8.2.3. реалізувати функціями, вести у потік cout

Архітектура:

Модуль s_calculation.

Прототипи функцій для кожної окремої підзадачі розташовані на початку програми.

Далі було створено TestSuite_lab_8_2 для системного тестування, були протестовані 5 тест кейсів.

Контрольні запитання:

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

Мета процесів проектування програмного забезпечення за ISO/IEC 12207:2008 полягає в систематичному підході до розробки, впровадження, тестування та обслуговування програмного забезпечення. Основні задачі

включають визначення вимог до ПЗ, архітектурне та детальне проектування, кодування, тестування та інтеграцію компонентів, а також забезпечення якості та управління конфігураціями.

2. Обґрунтовано поясніть, чим функція мови програмування C/C++ відрізняється від модуля.

Функція відрізняється від модуля тим, що є блоком коду, який виконує певну операцію і може повертати значення, тоді як модуль (або файл джерела) може містити велику кількість функцій, глобальних змінних та інших типів даних, організуючи код у логічні секції.

3. Яка відмінність функції main від решти функцій C/C++?

З функції main починається виконання будь-якої програми.

4. Яке призначення маніпуляторів і яким чином вони використовуються під час реалізації ПЗ мовою програмування C++?

Маніпулятори в C++ — це спеціальні функції або об'єкти, які використовуються для управління форматуванням виводу або вводу. Вони використовуються у потоках вводу-виводу для зміни параметрів потоку.

5. Як і для чого використовується заголовковий файл в процесі препроцесинга програми?

Заголовковий файл у процесі препроцесинга програми використовується для включення декларацій функцій, макросів, констант та інших визначень, що використовуються у багатьох частинах програми. Він дозволяє організувати код та спрощує його повторне використання та розподіл.

6. Що під час виконання лабораторної роботи Вами використано зі стандартного заголовкового файлу iostream та задля реалізації яких функцій?

system() для налаштування виводу у форматі UTF-8.

7. Що розуміють під стандартним простором імен у C++ і якою директивою він визначається?

Стандартний простір імен у C++ — це std, він включає стандартні класи та функції мови. Визначається директивою using namespace std;, яка дозволяє використовувати елементи стандартної бібліотеки без необхідності їх явного префіксування.

8. Наведіть приклади одночасного оголошення й ініціалізації початковим значенням змінної (об’єкта).

int a = 5;

double b = 3.17;

char c = 'A';

9. Перелічіть символи, використання яких у ідентифікаторах не допускаються відповідно до синтаксису мови C/C++.

Не допускається використання пробілів, ком, спеціальних символів в ідентифікаторах. Також не можна використовувати цифру на початку ідентифікатора.

10. Яким чином у C++ можливо вивести десятковий літерал у шістнадцятковій системі числення?

Наприклад: cout << hex << 255;

hex - маніпулятор

11. Що необхідно виконати для підключення й використання функцій нестандартної (наприклад, власної) бібліотеки?

Створіть персональну бібліотеку, виконайте її компіляцію, після цього розробіть заголовковий файл, що містить прототипи функцій, і також здійсніть його компіляцію. Надалі під'єднайте створену бібліотеку через Build options… задаючи шлях в compiler and linker.

12. Чим відрізняється заголовковий файл від об’єктного?

Заголовковий файл (.h) містить декларації функцій, макровизначення та визначення типів, що дозволяє їх використовувати у багатьох файлах програми без повторного опису. Об’єктний файл (.obj або .o) - це скомпільований кодовий файл, який генерується компілятором з вихідного коду, і який потім може бути зв'язаний з іншими об'єктними файлами для створення виконуваної програми.

13. Яким чином мовою програмування C/C++ можна реалізувати консольне виведення нестандартних символів (наприклад,)?

Можна використовувати управляючі послідовності (якщо символ є частиною ASCII) або використовувати спеціальні функції та бібліотеки для роботи з Unicode, наприклад, wprintf з wide-символами.

14. В чому полягає відмінність між записом символьного і рядкового константного літерала в C/C++?

Символьний літерал записується у одинарних лапках ('a'), тоді як рядковий константний літерал - у подвійних ("abc"). Символьний літерал представляє один символ, тоді як рядковий літерал - послідовність символів, що завершується нуль-символом.

15. Яким є синтаксис запису прототипа функції у C/C++? Де він записується у програмі та яке його призначення?

тип_повернення назва_функції(тип_параметра1, тип_параметра2, ...);

Прототип зазвичай розміщується на початку файлу або у заголовковому файлі для інформування компілятора про існування функції, що дозволяє викликати її перед її власним описом.

16. Який оператор C/C++ призначений для повернення функцією значення - результату і який синтаксис його запису?

Оператор return у C/C++ використовується для повернення значення функцією. Синтаксис: return вираз;

17. Перелічіть ключові символи (ESC-послідовності) мови C, їх призначення, синтаксис запису та спосіб використання у C++.

Ключові символи (ESC-послідовності) в C включають:

\n (новий рядок),

\t (горизонтальний таб),

\\ (зворотня коса риска),

\" (двійні лапки),

\' (одинарні лапки).

Вони використовуються для представлення спеціальних символів у рядках.

18. Перелічіть відомі Вам функції заголовкового файла cmath та їх призначення? В чому полягає їх відмінність від функцій бібліотеки math.h?

Функції заголовкового файла cmath включають sqrt() (квадратний корінь), pow() (степінь), sin() (синус), cos() (косинус) та інші. Відмінність від math.h полягає в тому, що cmath адаптований для C++, включає перевантаження функцій та може використовувати простори імен.

19. Яке призначення тестових драйверів?

Тестові драйвери призначені для автоматизації тестування коду, імітуючи взаємодію з користувачем або іншими частинами програми, дозволяючи перевірити окремі компоненти у ізоляції.

20. Яким чином здійснюється тестування модулів (функцій бібліотек) і чим означений процес відрізняється від тестування програмного засобу (exe-файла)?

Тестування модулів зазвичай включає в себе виклик функцій з різними вхідними даними та перевірку результатів, відокремлюючи їх від інших частин програми. Процес відрізняється від тестування програмного засобу, оскільки останнє передбачає перевірку інтеграції компонентів і взаємодії з користувачем або системою в цілому.