МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Центральноукраїнський національний технічний університет

Механіко-технологічний факультет

ЗВІТ

ПРО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 9

з навчальної дисципліни

“Базові методології та технології програмування”

Реалізація програмних модулів розгалужених та ітераційних обчислювальних процесів

ВИКОНАВ

Студент

академічної групи КН-23

Царенко Станіслав

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

Ганна ДРЄЄВА

Кропивницький – 2024

**ТЕМА:** Реалізація програмних модулів розгалужених та ітераційних обчислювальних процесів

**МЕТА:**— Набути ґрунтовних вмінь і практичних навичок реалізації технології модульного програмування, застосування операторів С/С++ арифметичних, логічних, побітових операцій, умови, циклів та вибору під час розроблення статичних бібліотек, заголовкових файлів та програмних засобів у кросплатформовому середовищі Code::Blocks

**ЗАВДАННЯ:**

1 Реалізувати функції розв’язування задач 9.1–9.3 як складових статичної бібліотеки libModulesПрізвище.а (проект ModulesПрізвище, створений під час виконання лабораторної роботи №8).

2. Реалізувати програмне забезпечення розв’язування задачі 9.4 на основі функцій статичної бібліотеки libModulesПрізвище.а

**ПОРЯДОК ВИКОНАННЯ:**

1. Завантажити власний Git-репозиторій https://github.com/odorenskyi/ student-name (в \Lab9\tasks містяться умови задач 9.1–9.4).

2. У \Lab9 заповнити файл README.md, створити теки prj, TestSuite, Software, Report; отриманий вміст теки \Lab9 завантажити до Gitрепозиторію https://github.com/odorenskyi/student-name; надалі здійснювати означену дію за позначкою .

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

4. Почергово здійснити аналіз і постановку задач 9.1, 9.2 та 9.3.

5. Виконати аналіз вимог, проектування архітектури, детальне проектування програмних модулів розв’язування задач 9.1–9.3; одержані артефакти задокументувати й включити до звіту .

6. Розробити три набори контрольних прикладів до задач 9.1–9.3 для виконання модульного тестування (Unit Testing) функцій; отримані результати задокументувати та включити до звіту .

7. В Code::Blocks IDE відкрити проект статичної бібліотеки ModulesПрізвище з \Lab8\prj, створений під час виконання лабораторної роботи № 8.

8. За отриманими під час проектування програмних модулів артефактами виконати конструювання функцій: мовою програмування С++ реалізувати функції, які за наданим інтерфейсом реалізовують розв’язування задач 9.1, 9.2 та 9.3 відповідно; проект статичної бібліотеки ModulesПрізвище, який вже містить реалізовану функцію s\_calculation, доповнюється функціями розв’язування задач 9.1, 9.2 та 9.3.

9. Скомпілювати проект статичної бібліотеки ModulesПрізвище.

10. Відкрити проект заголовкового файлу ModulesПрізвище із \Lab8\prj, створений під час виконання лабораторної роботи № 8, та доповнити його прототипами (заголовками) функцій розв’язування задач 9.1–9.3 .

11. У середовищі Code::Blocks створити в теці \prj проект консольного застосунку, іменувати його TestDriver.

12. Реалізувати тестовий драйвер для виконання модульного тестування функцій розв’язування задач 9.1, 9.2 та 9.3 за допомогою розроблених наборів контрольних прикладів . для включення функцій бібліотеки libModulesПрізвище.а до вихідного коду драйвера слід використати заголовковий файл ModulesПрізвище.h та належно налаштувати опції компілятора (Build options…); необхідно реалізувати протоколювання процесу тестування тестовим драйвером: виведення вхідних даних (аргументів функції, яка тестується), отриманий від функції результат та статус кожного тест-кейса (passed або failed).

13. За допомогою розробленого тестового драйвера (застосунку TestDriver.ехе) виконати модульне тестування функцій розв’язування задач 9.1–9.3 з бібліотеки libModulesПрізвище.а; у випадку невиконання тест-кейса(ів) слід відлагодити проект ModulesПрізвище (зазвичай з повторним виконанням детального проектування і/або конструювання), після чого процес модульного тестування повторити.

14. Результати unit-тестування задокументувати шляхом включення (копіювання) результатів роботи TestDriver.ехе до звіту ; текст з консольного вікна додатка у буфер операційної системи Windows можна скопіювати за допомогою контекстого меню, попередньо виділивши його.

15. Вихідний код проектів ModulesПрізвище та TestDriver включити до звіту як додатки.

16. Здійснити аналіз і постановку задачі 9.4.

17. Виконати аналіз вимог до ПЗ, проектування архітектури, детальне проектування програмного забезпечення розв’язування задачі 9.4; отримані артефакти задокументувати й включити до звіту .

18. Розробити тест-сьют для виконання системного тестування ПЗ задачі 9.4, який повинен складатись з достатньої кількості тесткейсів (є допустимим використання простої структури тестового випадка: Test Case ID → Action (test steps) → Expected Result → Test Result (passed/failed/blocked); отриманий тестовий артефакт (файл з тестовим набором) зберегти у \TestSuite .

19. В Code::Blocks IDE створити у теці \prj проект консольного застосунку Прізвище\_task.

20. Мовою програмування С++ реалізувати артефакти проектування програмного забезпечення розв’язування задачі 9.4 з використанням функцій статичної бібліотеки libModulesПрізвище.а (заголовковий файл ModulesПрізвище.h), скомпілювати проект .

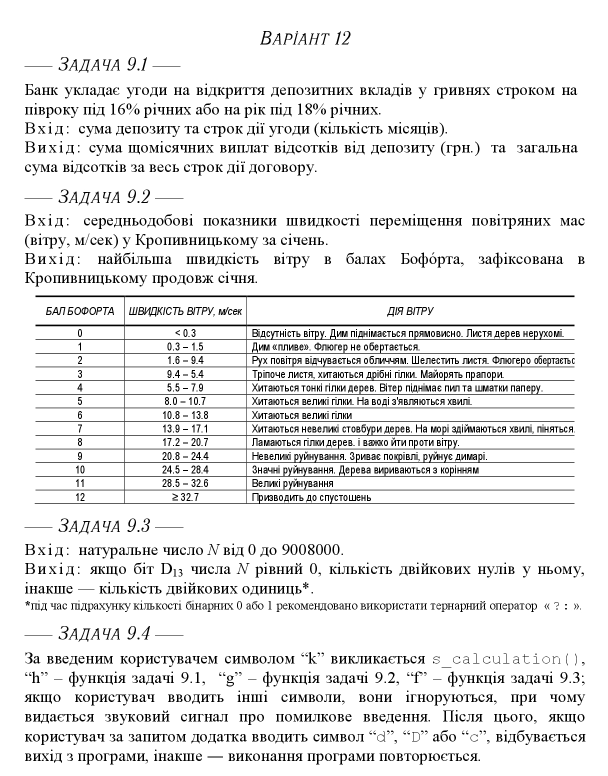
21. Виконати системне тестування створеного Прізвище\_task.ехе за допомогою тестового набору із \TestSuite, результати тестування задокументувати й включити до звіту як додаток ; у випадку невиконання тест-кейса(ів) слід виконати відлагодження проекта, після чого тестування ПЗ повторити.

22. Розроблений Прізвище\_task.ехе скопіювати у \Software, вихідний код проекта Прізвище\_task включити до звіту як додаток .

23. Проаналізувати хід виконання лабораторних завдань і самостійно одержані результати, на основі чого сформулювати обґрунтовані висновки1 з виконаної лабораторної роботи, викласти їх обсягом не менше двох 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 критику2, рекомендації тощо. 1 висновки, як результат розумової діяльності студента, повинні, зокрема, містити стислий виклад самостійно здобутих результатів в процесі виконання завдань, реалізованих власних проектних рішень, шляхи вирішення проблем, які виникли під час виконання лабораторної роботи; окремим абзацом слід конкретизовано викласти висновок про досягнення мети лабораторної роботи; структура висновків має бути логічною і охоплювати весь процес виконання лабораторної роботи. 2 критика є розглядом і оцінкою чогось з метою виявлення й усунення вад, хиб; під конструктивною слід розуміти критику, після якої стає зрозумілим, як саме виправити помилку й не допускати її в майбутньому.

24. Підготувати й зберегти у \Lab9\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.



9.1  
  
Алгоритм

1. Ввести deposit, months

2. Обчислити за формулами:



3.Вивести monthlyInterest, totalInterest

4.Кінець

Лістинг модуля

void bank(double depositAmount, int months) {

double monthlyInterest, totalInterest;

double annualRate = (months == 6) ? 0.16 : 0.18;

monthlyInterest = (depositAmount \* annualRate) / 12;

totalInterest = monthlyInterest \* months;

cout << "Сума щомісячних виплат відсотків: " << monthlyInterest << " грн." << endl;

cout << "Загальна сума відсотків за весь строк дії договору: " << totalInterest << " грн." << endl;

}

9.2

Лістинг модуля

void beaufort(double\* speed, int length) {

double max = speed[0];

cout << endl;

for (int i = 1; i < length; i++) {

if (max < speed[i])

max = speed[i];

}

cout << "Максимум: " << max << endl;

if (max < 0.3)

cout << "Бал Бофорта 0!" << endl;

else if (0.3 < max && max < 1.5)

cout << "Бал Бофорта 1!" << endl;

else if (1.6 < max && max < 3.4)

cout << "Бал Бофорта 2!" << endl;

else if (3.4 < max && max < 5.4)

cout << "Бал Бофорта 3!" << endl;

else if (5.5 < max && max < 7.9)

cout << "Бал Бофорта 4!" << endl;

else if (8 < max && max < 10.7)

cout << "Бал Бофорта 5!" << endl;

else if (10.8 < max && max < 13.8)

cout << "Бал Бофорта 6!" << endl;

else if (13.9 < max && max < 17.1)

cout << "Бал Бофорта 7!" << endl;

else if (17.2 < max && max < 20.7)

cout << "Бал Бофорта 8!" << endl;

else if (20.8 < max && max < 24.4)

cout << "Бал Бофорта 9!" << endl;

else if (24.5 < max && max < 28.4)

cout << "Бал Бофорта 10!" << endl;

else if (28.5 < max && max < 32.6)

cout << "Бал Бофорта 11!" << endl;

else if (max >= 32.7)

cout << "Бал Бофорта 12!" << endl;

}

9.3

Лістинг модуля

unsigned int bits(unsigned int num) {

int zeros = 0;

int ones = 0;

int mask = 1;

for (int i = 0; i < 16; ++i) {

if (num & mask) {

ones++;

} else {

zeros++;

}

mask <<= 1;

}

return (num & 0x8000) ? ones : zeros;

}

Лістинг тестового драйвера

#include <iostream>

#include"ModulesTsarenko.h"

using namespace std;

int main()

{

system("chcp 1251 & cls");

bank(20000, 5);

double speed[] = {1.1, 2.3, 0.9, 2.1, 0.8, 3.0, 0.6, 1.6, 0.5, 0.7, 1.8, 0.2, 2.7, 0.3, 2.5, 0.1, 1.7, 2.2, 0.4, 1.3, 1.4, 2.9, 0.0, 1.5, 2.8, 0.2, 2.4, 0.8, 1.9, 2.6, 0.1};

int length = sizeof(speed) / sizeof(speed[0]);

beaufort(speed, length);

cout << bits(1);

}

**Контрольні запитання і завдання**

1. Яким оператором С/С++ можливо повноцінно замінити

тернарний оператор?

Тернарний оператор:

result = (condition) ? value1 : value2;

Еквівалентна конструкція з if-else:

if (condition) {

result = value1;

} else {

result = value2;

}

2. Що в програмуванні розуміють під пріоритетом виконання

операцій та асоціативністю?

Пріоритет виконання операторів визначає порядок, в якому вони виконуються. Асоціативність оператора визначається напрямком обробки операндів при однаковому пріоритеті. Наприклад, вирази a + b + c та a - b - c мають ліву асоціативність, тобто оператори виконуються зліва направо.

3. Яку область видимості мають змінні, оголошені в тілі циклу

або умови (вибору)?

Змінні, оголошені в тілі циклу або умови (вибору), мають область видимості, обмежену цим блоком коду. Поза цим блоком вони недоступні. Експериментально можна перевірити цю особливість.

4. Якою є асоціативність операцій арифметичних, логічних,

логічних порозрядних, інкреметна, декремента, тернарної та

порівняння в мові програмування С/С++?

Асоціативність та пріоритет операторів в С/С++ залежить від їх типу та призначення.

5. Перелічіть випадки, за яких доцільно використовувати тернарний оператор С/С++, й наведіть приклад його запису.

Тернарний оператор доцільно використовувати, коли потрібно обрати один із двох значень залежно від умови.

Приклад використання тернарного оператора:

int x = 5;

int y = (x > 0) ? 10 : -10;

Якщо x > 0, то y = 10, інакше y = -10

6. Яке значення міститиме змінна cnt після виконання наступної

інструкції: cnt--; ?

Після виконання інструкції cnt--; значення змінної cnt буде зменшене на 1.

7. Чим константна змінна, оголошена за допомогою кваліфікатора

типів const, відрізняється від змінної? Сформулюйте правило,

коли змінну варто оголошувати саме константною

Константна змінна, оголошена з допомогою ключового слова const, не може бути змінена після ініціалізації. Це дозволяє захистити значення від неочікуваних змін. Змінну варто оголошувати константною, коли її значення не повинно змінюватись у програмі.

8. Яких типів можуть бути операнди логічних операторів С/С++?

Операнди логічних операторів в С/С++ можуть бути типу bool.

9. Яке значення міститиме змінна cnt при: bool cnt = !!0; ?

При виконанні bool cnt = !!0;, змінна cnt буде мати значення false, оскільки подвійна занегація від нуля дає значення false у логічному контексті.

10. Сформулюйте правило запису виразу ініціалізації у циклах з

параметром (for) С++.

Правило запису виразу ініціалізації у циклах з параметром (for) у С++ полягає в тому, що можна вказати початкову ініціалізацію, умову продовження циклу та вираз крокування, що виконується після кожної ітерації циклу.

Приклад:

for (int i = 0; i < 10; i++) {

Тіло циклу

}

У цьому прикладі int i = 0 - початкова ініціалізація, i < 10 - умова продовження циклу, i++ - вираз крокування.