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

студентка академічної групи КН-23

Горбенко К.О.

ПЕРЕВІРИВ

викладач кафедри кібербезпеки

та програмного забезпечення

Ганна ДРЄЄВА

Кропивницький – 2024

**Лабораторна робота №8**

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

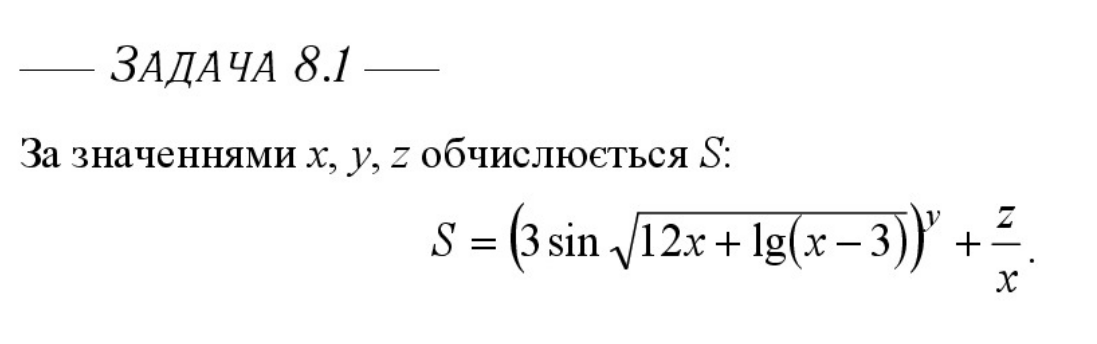
*Мета***:** Полягає у набутті ґрунтовних вмінь і практичних навичок реалізації метода функціональної декомпозиції задач, метода модульного (блочного) тестування, представлення мовою програмування С/С++ даних скалярних типів, арифметичних і логічних операцій, потокового введення й виведення інформації,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

**Завдання 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 17**

******

***Завдання 8.1***

Вхідні дані: x, y, z – Дробові дані float

Вихідні дані: S – Дробовий тип float

Якщо: Х < 2, то буде виведена помилка

Інакше: буде виконуватися обчислення S

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

***Лістинг проекту ModulesHorbenko:***

#include <cmath>

#include <iostream>

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

{

if (x < 2)

{

std::cout << "'Nan' Помилка, значення X менше 2 " << std::endl;

return 0;

}else

{

return pow(3 \* sin(sqrt(12 \* x + log10(x - 3))), y) + (z / x);

}

}

***ModulesHorbenko.h :***

#ifndef MODULESHORBENKO\_H\_INCLUDED

#define MODULESHORBENKO\_H\_INCLUDED

float s\_calculation (float , float , float );

#endif // MODULESHORBENKO\_H\_INCLUDED

***Лістинг проекту TestDriver:***

#include <iostream>

#include <iomanip>

#include <limits>

#include "ModulesHorbenko.h"

using namespace std;

void testSCalculation(float ts\_num, float x, float y, float z, float expected)

{

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

// cout << setprecision(numeric\_limits < double > ::digits10 + 2) << result << endl;

if (result == expected)

{

cout << "Test Passed\_" << ts\_num << endl;

} else

{

cout << "Test Failed\_" << ts\_num << endl;

}

cout << "=======================" << endl;

}

int main()

{

system("chcp 1251 & cls");

testSCalculation(1, 5, 2, 10, 10.929522514343262);

testSCalculation(2, 12, 5, 8, -7.1314487457275391);

testSCalculation(3, 1, 11, 10, numeric\_limits < float > ::quiet\_NaN());

testSCalculation(4, 5, 2, 0, 8.9295225143432617);

testSCalculation(5, 5, 10, 8, 56774.515625);

testSCalculation(6, 5, -2, 10, 2.1119880676269531);

testSCalculation(7, 5, 2, -10, 6.9295225143432617);

testSCalculation(8, 50, 15, 100, -2941.4345703125);

testSCalculation(9, 10, 5, 10, -241.99583435058594);

testSCalculation(10, 67, 3, 8, 0.0813121497631073);

return 0;

}

***Результат:***

Test\_1 Passed

Result: 10.9295

==================================

Test\_2 Passed

Result: -7.13145

==================================

'Nan' Помилка, значення X менше 2

Test\_3 Failed

==================================

Test\_4 Passed

Result: 8.92952

==================================

'Nan' Помилка, значення X менше 2

Test\_5 Failed

==================================

'Nan' Помилка, значення X менше 2

Test\_6 Failed

==================================

Test\_7 Passed

Result: 6.92952

==================================

Test\_8 Passed

Result: -2941.43

==================================

Test\_9 Passed

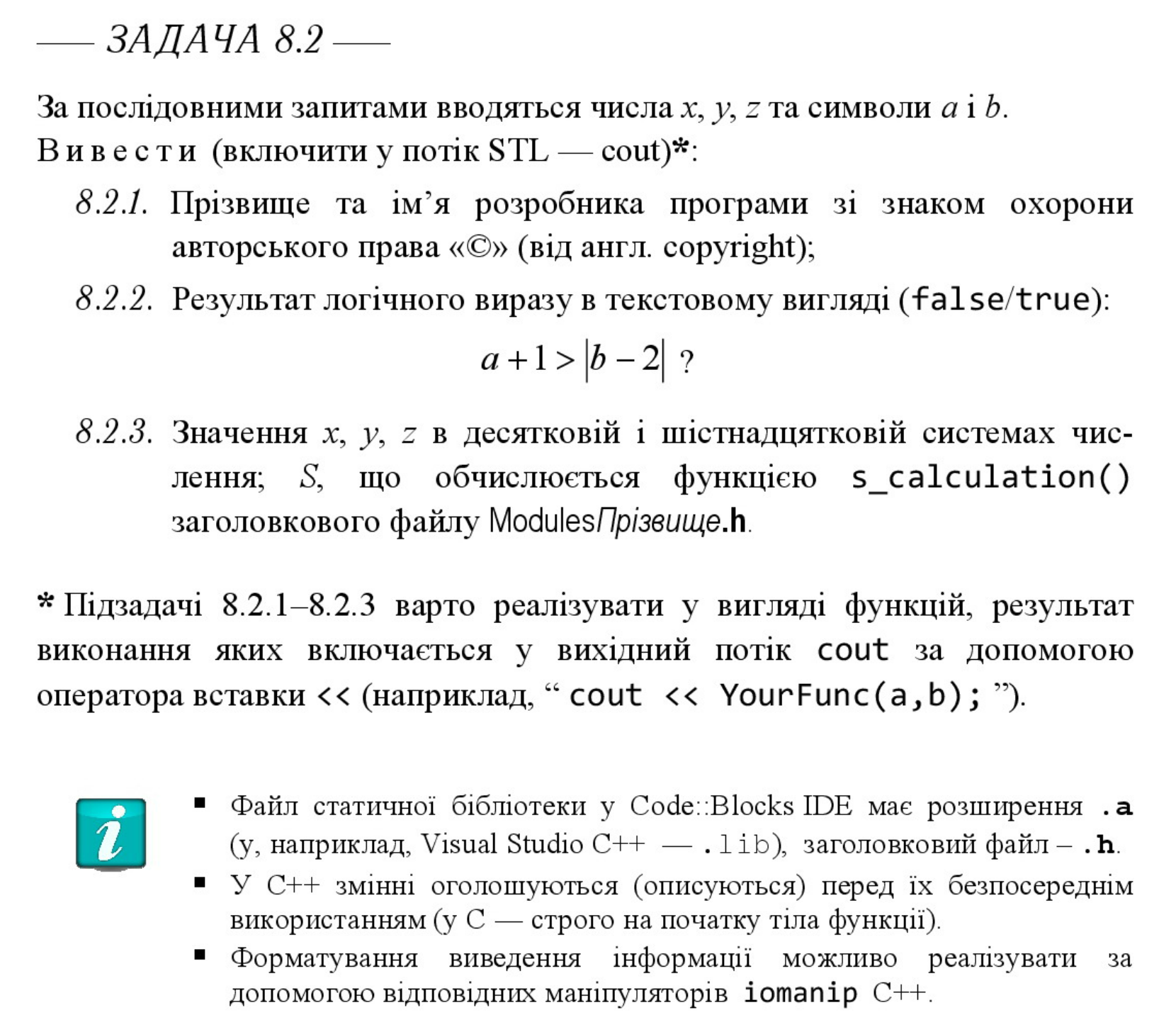
Result: -241.996

==================================

Test\_10 Passed

Result: 0.0813121

==================================



***Завдання 8.2***

Вхідні дані:

x, y, z – Дробові дані float

a і b – символьний тип char

Вихідні дані:

Результат – true або false;

x, y, z – Які виводяться в десятковій і шістнадцятковій системах числення;

S – Який обчислюється функцією s\_calculation

***Лістинг проекту Horbenko\_task:***

#include <iostream>

#include "ModulesHorbenko.h"

using namespace std;

const char\* printCopyright()

{

system("chcp 1251 & cls");

return " -----------------------------------------------------\n\

| Kate Horbenko, CNTU, github.com/Horbenko-Kate |\n\

| Катерина Горбенко, ЦНТУ, github.com/Horbenko-Kate |\n\

--------------- © All rights reserved ---------------\n";

}

bool getBoolResult(char a, char b)

{

cout << boolalpha;

return (a + 1 > abs(b - 2));

}

int getHexadecimal(int number)

{

cout << hex;

return number;

}

int main()

{

cout << printCopyright() << endl;

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;

cout << "\nРезультат 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;

}

***Результат:***

***Висновок:***

Отже провівши лабораторну роботу, я здобула знання та практичні навички, необхідні для реалізації методології модульного програмування. Крім того, я оволодів методом функціональної декомпозиції задач, методом модульного тестування та вміннями роботи з мовою програмування C++. Ці навички дозволять мені ефективно розробляти програмні модулі та засоби у кросплатформовому середовищі Code::Blocks з використанням GNU GCC Compiler.

В цій лабораторній роботі я використовувала Git-репозиторій. Завдяки методичці, я тепер зрозуміла, коли саме варто робити коміти, і навчилася фіксувати кожен етап моєї роботи крок за кроком. Додавання комітів, включення нових файлів та клонування репозиторіїв тепер стали для мене зрозумілішими та простішими.

В ході виконання завдання 8.2 був розроблений модуль, який містить функцію для обчислення значення виразу S за введеними параметрами x, y та z. Ця функція називається s\_calculation – вона перевіряє, чи значення x більше 2. Якщо x менше 2, вона виводить повідомлення про помилку і повертає 0. В іншому випадку вона обчислює значення виразу S за заданими формулами і повертає результат.

Для організації інтерфейсу функції був створений заголовочний файл ModulesHorbenko.h, який містить декларацію функції s\_calculation.

Після цього було розроблено програму-тестувальник, яка використовує функцію s\_calculation для перевірки її коректності на різних вхідних даних. У цій програмі використовуються набори тестових даних для різних значень x, y та z, і порівнюється отриманий результат з очікуваним.

В ході виконання завдання 8.2 було розроблено програму, яка відповідає поставленим вимогам. Програма приймає на вхід числа x, y, z типу float і символи a та b типу char. Після цього вона виконує наступні дії:

Виводить ім'я та прізвище розробника програми зі знаком охорони авторського права "©".

Обчислює результат логічного виразу a + 1 > |b - 2| і виводить його у текстовому вигляді (false або true).

Виводить значення x, y, z у десятковій і шістнадцятковій системах числення.

Обчислює значення S за допомогою функції s\_calculation() з попередньо розробленого заголовкового файлу ModulesHorbenko.h і виводить його.  
  
Обидва завдання були успішно вирішені, що дозволило поглибити знання з використання мови програмування C++, роботи з функціями, операторами введення/виведення та роботи з рядками та числовими типами даних.

***Контрольні запитання і завдання:***

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

- Планування архітектурних рішень

Визначення організаційної структури та компонентів системи

- Деталізація проекту

Уточнення характеристик та визначення специфікацій компонентів системи

- Реалізація розробки

Створення компонентів системи відповідно до детальних специфікацій

- Інтеграція програмного забезпечення

Збирання та з'єднання різних компонентів та модулів для утворення 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 мовах програмування С/С++ використовується для вставки вмісту файлів заголовків у вихідний код програми. Це дозволяє використовувати функції, константи та інші елементи, описані у включених файлах, у програмі.

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

У заголовковому файлі мови C++ зазвичай містяться прототипи функцій, декларації змінних та інших об'єктів. Для того щоб включити ці заголовкові файли у 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 містити літери, цифри та символ підкреслення і повинен починатися з літери або символу підкреслення. Його довжина може бути обмежена певним компілятором. Регістр літер у ідентифікаторі важливий, що означає, що ідентифікатори "abc" і "ABC" розглядаються як різні.

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 або модуля розуміється набір зовнішніх функцій, їх параметрів та типів даних, які 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 і endl використовуються для форматування виведення даних у потоках. boolalpha перетворює значення типу bool у "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 перевірці роботи окремих модулів програми. Для цього створюються тестові кейси, які перевіряють коректність функціонування кожного модуля незалежно від інших частин програми. Після цього застосовуються тест-сьюти для виконання тестування за допомогою спеціально розробленого тестового драйвера, який виконує перевірку усіх тестових кейсів.

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.