МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Львівський національний університет імені Івана Франка

Факультет електроніки та комп'ютерних технологій

Кафедра оптоелектроніки та інформаційних технологій

Допустити до захисту

Завідувач кафедри

_____проф.

«___» _____20__р.

Кваліфікаційна робота

Бакалавр

"Порівняльна характеристика мов веб програмування C++ і C#"

Виконав:

Студент IV курсу групи ФЕП-41с

Спеціальності 121 – Інженерія програмна забеспечення

_____ Медицький Павло

Науковий керівник:

_____ доц. Злобін Г.Г.

«___» _____20__р.

Рецензент:

_____ _____.

Львів 2023

АНОТАЦІЯ

Дипломна робота на тему "Порівняльна характеристика мов веб програмування C++ і C#" присвячена аналізу та порівнянню двох популярних мов програмування - C++ та C# - з точки зору веб-розробки. Основною метою дослідження є виявлення переваг та недоліків кожної мови з метою зробити обґрунтований вибір при розробці веб-проектів.

У роботі проводиться аналіз різних аспектів обох мов, зокрема їх швидкодії та продуктивності, надійності та безпеки, підтримки інтегрованих середовищ розробки, масштабованості та розширюваності, а також наявності стандартних бібліотек та програмний каркасів.

Для досягнення поставленої мети використовується метод дослідження, що базується на аналізі наукової літератури, офіційних документацій, веб-ресурсів та практичних прикладів. Також будуть проведені експерименти та тестування з метою порівняння продуктивності та швидкодії між C++ та C#.

В результаті дослідження очікується отримати об'єктивну порівняльну характеристику мов веб програмування C++ та C#, яка допоможе розробникам, проектним командам та організаціям зробити обґрунтований вибір при розробці веб-проектів. Отримані результати та висновки будуть представлені у відповідній структурованій формі і супроводжуватимуться підтверджуючими доказами.

Ключові слова: веб-розробка, C++, C#, порівняльна характеристика, швидкодія, продуктивність, надійність, безпека, масштабованість, розширюваність, стандартні бібліотеки, програмний каркаси.

Abstract

The diploma thesis on the topic "Comparative Characteristics of C++ and C# Web Programming Languages" is dedicated to the analysis and comparison of two popular programming languages - C++ and C# - in the context of web development. The main objective of the research is to identify the advantages and disadvantages of each language in order to make an informed choice when developing web projects.

The thesis analyzes various aspects of both languages, including their performance and productivity, reliability and security, support for integrated development environments, scalability and extensibility, as well as the availability of standard libraries and frameworks.

To achieve the set goal, a research method based on the analysis of scientific literature, official documentation, web resources, and practical examples is used. Experiments and testing will also be conducted to compare the performance and speed between C++ and C#.

The expected outcome of the research is to obtain an objective comparative characterization of C++ and C# as web programming languages, which will help developers, project teams, and organizations make informed decisions when developing web projects. The obtained results and conclusions will be presented in a structured form and supported by supporting evidence.

Keywords: web development, C++, C#, comparative characteristics, performance, productivity, reliability, security, scalability, extensibility, standard libraries, frameworks

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

1. Інкапсуляція — один з трьох основних механізмів об'єктно-орієнтованого програмування. Йдеться про те, що об’єкт вміщує не тільки дані, але і правила їх обробки, оформлені в вигляді виконуваних фрагментів.
2. Швидкодія - Вимірювання продуктивності програми, її швидкості виконання та оптимізації часу роботи.
3. Масштабованість - Здатність системи адаптуватися до збільшення обсягів даних, навантаження або користувацького трафіку без втрати продуктивності.
4. Розширюваність - Здатність системи до додавання нової функціональності, модифікації та розширення без зміни основного коду.
5. Програмний каркаси - Загальна структура та набір інструментів для розробки програмного забезпечення, які спрощують роботу розробників та надають готові рішення для певних завдань.
6. HTTP - протокол передачі даних, що використовується в комп'ютерних мережах. Назва скорочена від HyperText Transfer Protocol.
7. PHP, попередня назва: Personal Home Page Tools — скриптова мова програмування, була створена для генерації HTML-сторінок на стороні вебсервера.
8. .NET Framework — програмна технологія, запропонована фірмою Microsoft як платформа для створення як звичайних програм, так і веб-застосунків.
9. ECMA — стандарт мови програмування, затверджений міжнародною організацією ECMA згідно зі специфікацією ECMA-262.
10. ASP.NET — технологія створення вебзастосунків і вебсервісів від компанії Майкрософт. Вона є складовою частиною платформи Microsoft.NET і розвитком старішої технології Microsoft ASP.
11. SQL — декларативна мова програмування для взаємодії користувача з базами даних, що застосовується для формування запитів.

ВСТУП

1.1 Актуальність теми

Веб програмування стало невід'ємною складовою розвитку сучасного суспільства. З кожним роком зростає попит на веб-додатки та веб-сайти, що надають широкий спектр функціональності, від інформаційних порталів до електронних комерційних систем. Водночас, існує ряд мов програмування, які можна використовувати для розробки веб-додатків, зокрема мови C++ і C#, які вважаються одними з найпопулярніших і ефективних мов програмування.

Однак, вибір мови програмування для веб-розробки є важливим і складним завданням. Хоча C++ і C# мають багато спільних рис і призначення, вони мають свої особливості, які можуть впливати на продуктивність, швидкодію, безпеку та розширюваність веб-додатків. Тому, порівняльний аналіз мов веб програмування C++ і C# є важливим кроком для розуміння їх особливостей та визначення найбільш оптимального варіанту для конкретних веб-проектів.

Метою цієї дипломної роботи є проведення порівняльного аналізу мов веб програмування C++ і C#. Дослідження спрямоване на визначення основних характеристик кожної мови, їх синтаксису, структури та можливостей для розробки веб-додатків. Додатково, дослідження включатиме порівняльний аналіз швидкодії, продуктивності, налагодження, безпеки, розширюваності та інших аспектів, що впливають на ефективність веб-проектів, розроблених з використанням мов C++ і C#.

Об'єктом дослідження є мови веб програмування C++ і C#, їх синтаксис, структура, особливості та можливості для розробки веб-додатків.

Предметом дослідження є порівняльний аналіз мов C++ і C# у контексті веб програмування, зокрема їх характеристики, продуктивність, швидкодія, налагодження, безпека та розширюваність.

Методологія дослідження: для досягнення поставленої мети дослідження будуть використані наступні методи:

Аналіз літератури: буде проведено систематичний огляд наукової літератури та джерел, що стосуються мов веб програмування C++ і C#. Цей аналіз надасть теоретичну базу для порівняльного аналізу мов.

Експертна оцінка: будуть проведені консультації з досвідченими розробниками, які використовують мови C++ і C# для веб програмування, щоб отримати їхній погляд на переваги і недоліки кожної мови.

Практичні експерименти: будуть розроблені та реалізовані прості веб-додатки з використанням мов C++ і C#. Ці експерименти дадуть змогу перевірити продуктивність, швидкодію та інші аспекти розроблених додатків.

Аналіз результатів: будуть проаналізовані отримані дані та зроблено висновки щодо порівняльних характеристик мов C++ і C# у веб програмуванні.

Загальний підхід до дослідження буде науковим та об'єктивним, з використанням як кількісних, так і якісних методів аналізу даних. Отримані результати дослідження будуть використані для надання порад та рекомендацій щодо вибору між мовами C++ і C# для веб-розробки.

РОЗДІЛ 1 ВЕБ ПРОГРАМУВАННЯ

1.1 Поняття веб програмування

Веб програмування є процесом розробки веб-додатків, що забезпечують взаємодію користувача з інформацією через Інтернет. Веб-додатки можуть включати різноманітні функціональні можливості, такі як відображення динамічного контенту, обробка форм, взаємодія з базами даних, автентифікація користувачів та інші.

У веб програмуванні використовуються спеціальні мови програмування, такі як JavaScript, Python, PHP, Ruby та інші, що дають змогу розробникам створювати функціональні та інтерактивні веб-додатки. Однак, вибір мови програмування залежить від потреб проекту, його характеристик, швидкодії, масштабованості та інших факторів.

Основні принципи веб програмування включають в себе використання стандартних мов програмування, розробку клієнт-серверної архітектури, використання протоколів передачі даних, таких як HTTP (HyperText Transfer Protocol), та використання програмний каркасів та бібліотек для спрощення розробки та підтримки веб-додатків.

Дослідження літератури засвідчує, що веб програмування має суттєвий вплив на розвиток сучасного суспільства, сприяючи створенню ефективних та інтерактивних веб-додатків, які забезпечують зручний доступ до інформації та послуг через Інтернет.

1.2 Основні мови веб програмування

В цьому розділі проведений огляд основних мов програмування, що використовуються для розробки веб-додатків. Основний акцент буде зроблений на

мовах JavaScript, Python, PHP та Ruby, які широко використовуються в сфері веб програмування. Для кожної з цих мов будуть розглянуті основні характеристики та переваги.

JavaScript є однією з найпопулярніших мов програмування для веб-розробки. Вона використовується для створення динамічного та інтерактивного контенту на веб-сторінках. JavaScript дозволяє взаємодіяти з HTML-документами, обробляти події, створювати анімацію, валідувати форми та виконувати інші завдання, що покращують користувацький досвід.

Python є мовою програмування, яка набула популярності в останні роки у веб-розробці. Вона має простий і зрозумілий синтаксис, що полегшує розробку веб-додатків. Python забезпечує широкий спектр програмних каркасів, таких як Django та Flask, які спрощують розробку веб-додатків шляхом надання готової функціональності та інструментів.

PHP є однією з найпопулярніших мов програмування для веб-розробки. Вона широко використовується для розробки динамічних веб-сайтів та веб-додатків. PHP працює на більшості серверів із підтримкою PHP (Personal Home Page Tools) та має велику кількість програмних каркасів, таких як Laravel та Symfony, що спрощують розробку та підтримку веб-додатків.

Ruby є мовою програмування, яка набула популярності завдяки програмному каркасу Ruby on Rails. Вона відома своєю простотою та елегантним синтаксисом. Ruby on Rails надає готову архітектуру та шаблони, що дають змогу розробникам швидко створювати веб-додатки та зосередитися на бізнес-логіці.

Кожна з цих мов має свої переваги та обмеження, і вибір мови залежить від потреб проекту, вміння розробника та вимог до продуктивності, безпеки та масштабованості веб-додатків. Детальний аналіз цих мов дозволить зрозуміти їхні можливості та вибрати найбільш підходящу мову для конкретного веб-проекту.

1.3 Історія та розвиток мов C++ і C#

«C++» була розроблена в 1980-х роках Бьярном Страуструпом як розширення мови програмування C. Головною метою було поєднання можливостей мови C з об'єктно-орієнтованими концепціями. У 1983 році вийшла перша версія мови «C++» з назвою "C with Classes", а пізніше у 1985 році була випущена остаточна версія мови під назвою «C++.»

Протягом наступних десятиліть C++ зазнала багатьох змін та розширень. У 1998 році була випущена стандартна специфікація «C++98», яка включала нові можливості, такі як шаблони, виняткові ситуації та інші. У 2011 році був випущений стандарт «C++11», який вніс ще більше покращень, включаючи розширену підтримку для паралельного програмування, розширені шаблони та підтримку рухомих семантик.

Сьогодні «C++» є однією з найпопулярніших мов програмування у світі. Вона використовується не тільки у веб-розробці, але й у системному програмуванні, ігровій індустрії, фінансових технологіях та інших областях. Існують численні програмний каркаси та бібліотеки, такі як «Qt,» «Boost» і «POCO», які допомагають розробникам веб-додатків на базі «C++».

«C#» була розроблена компанією Microsoft в кінці 1990-х років як мова програмування для платформи .NET. Вона була представлена в 2000 році разом із випуском .NET Framework. Ціль створення «C#» була в тому, щоб мати просту, сучасну та ефективну мову програмування для розробки програм на платформі .NET.

C# поєднує в собі елементи мов «C++» та «Java», заснована на стандартах ECMA (European Computer Manufacturers Association) та ISO (International Organization for Standardization). Вона має сильну підтримку об'єктно-орієнтованого програмування, включає в себе ряд продуктивних функцій та має велику кількість інструментів та програмний каркасів, що сприяють розробці веб-додатків.

З часом C# отримала багато оновлень і випусків нових версій. Кожна нова версія вносила нові можливості та покращення в мову. Наприклад, «C# 2.0» додав

генерики та інші поліпшення, «C# 3.0» представив LINQ (Language-Integrated Query) та анонімні типи, а «C# 5.0» вніс підтримку асинхронного програмування.

Сьогодні «C#» є однією з найпопулярніших мов програмування для розробки веб-додатків на платформі .NET. Розробники використовують програмний каркаси, такі як «ASP.NET» та «ASP.NET Core», щоб створювати потужні та масштабовані веб-додатки за допомогою «C#».

ОЗДІЛ 2 ОСНОВНІ ХАРАКТЕРИСТИКИ МОВИ С++

2.1 Синтаксис та структура

«С++» базується на мові С та додає до неї об'єктно-орієнтовані можливості. Синтаксис «С++» включає в себе використання крапки з комою (;) для закінчення виразів, фігурні дужки ({}) для групування блоків коду, оператори, такі як if, for, while для управління потоком виконання програми, та інші ключові слова, оператори та вирази.

Структура мови «С++» може бути організована на рівні файлів, класів, функцій та модулів. Файли у «С++» зазвичай містять оголошення класів та функцій, а також виконуваний код. Класи в «С++» описують структуру та поведінку об'єктів, вони містять поля, методи та конструктори. Функції в «С++» використовуються для виконання певних операцій і можуть бути визначені як частини класів або як окремі функції.

2.2 Основні конструкції

Змінні та типи даних: «С++» підтримує широкий спектр типів даних, включаючи цілі числа, дійсні числа, символи, рядки, булеві значення та вказівники. Використовуються оператори присвоєння (=), арифметичні оператори (+, -, *, /), оператори порівняння (==, !=, <, >) та логічні оператори (&&, ||, !) для роботи зі змінними.

Умовні конструкції: Умовні конструкції, такі як if-else, switch, дають змогу виконувати певний блок коду на основі умови. Наприклад, за допомогою if-else можна визначати різні гілки виконання коду в залежності від заданої умови.

Цикли: «С++» підтримує цикли, такі як for, while, do-while, які дають змогу повторювати певний блок коду. Цикли використовуються для ітерації через масиви, колекції або для виконання певного коду певну кількість разів.

Функції: В «C++» можна оголошувати та визначати функції, які приймають параметри та повертають значення. Функції дають змогу організувати логічний розподіл коду та повторне використання.

2.3 Об'єктно-орієнтоване програмування

У цьому пункті розглядаються основні принципи об'єктно-орієнтованого програмування (ООП) і їх використання в мовах веб-програмування «C++» і «C#». Об'єктно-орієнтоване програмування - це парадигма програмування, що базується на концепції об'єктів, які взаємодіють між собою для виконання певних завдань. Розглянемо детальніше основні аспекти об'єктно-орієнтованого програмування:

Класи та об'єкти: Класи є основними будівельними блоками ООП і представляють структуру, яка описує поведінку та стан об'єктів. Класи містять властивості (змінні) та методи (функції), що описують характеристики та дії об'єктів. Вони дають змогу організувати код у логічні блоки, що спрощує його управління та розширення. Об'єкти, з іншого боку, є конкретними екземплярами класів. Вони мають свій стан (значення змінних) і можуть виконувати методи, взаємодіяти з іншими об'єктами та зовнішніми системами.

Інкапсуляція: Інкапсуляція є принципом ООП, що дозволяє об'єднувати дані та методи, що оперують ними, в межах класу. Це означає, що дані можуть бути приховані (приватні) і недоступні для зовнішнього доступу безпосередньо. Замість цього, доступ до даних здійснюється через публічні методи (гетери та сетери), що забезпечують контрольований доступ до даних. Інкапсуляція дозволяє зберігати дані в безпечному стані і полегшує зміну реалізації класу без впливу на зовнішній інтерфейс.

Наслідування: Наслідування є ключовим аспектом ООП, що дозволяє створювати нові класи на основі вже існуючих (батьківських) класів. Це спадкування властивостей та методів від батьківського класу до дочірнього. Наслідування дозволяє створювати ієрархію класів, де батьківський клас є загальним для дочірніх класів і надає спільні характеристики та функціональність. Дочірні класи можуть

розширювати або модифікувати функціонал батьківського класу, а також додавати власні властивості та методи. Наслідування спрощує повторне використання коду і дозволяє створювати більш структуровані та модульні програми.

Поліморфізм: Поліморфізм означає можливість об'єктів різних класів використовувати одну і ту саму назву методу для виконання різних дій. Це дозволяє реалізувати багатоманітність і варіативність виконання коду. Поліморфізм може бути реалізований через віртуальні методи, абстрактні класи та інтерфейси. Він дозволяє створювати загальні методи та класи, які можуть взаємодіяти з об'єктами різних класів без необхідності визначення конкретного типу об'єкта.

Об'єктно-орієнтоване програмування надає багато переваг, таких як модульність, легкість управління, розширення та повторне використання коду. Ці принципи можуть бути використані для розробки ефективних та масштабованих веб-додатків на мовах «C++» і «C#».

2.4 Можливості веб програмування на C++

Висока продуктивність: C++ є мовою програмування з високою продуктивністю, що дозволяє розробляти веб-додатки, які працюють швидко та ефективно. Це особливо корисно для додатків, які вимагають обробки великих обсягів даних, виконання складних алгоритмів або потребують низької латентності.

Багатопоточність: C++ має потужність роботи з потоками, що дозволяє легко створювати багатопотокові веб-додатки. Це особливо корисно для веб-додатків, які мають велику кількість паралельних операцій або вимагають одночасної обробки багатьох запитів.

Доступ до системних ресурсів: C++ надає прямий доступ до системних ресурсів, таких як файлова система, мережеві операції, реєстри, додаткові пристрої тощо. Це дозволяє створювати веб-додатки, які взаємодіють з системою на низькому рівні або виконують специфічні операції, які необхідні для додатка.

Кросплатформність: C++ є кросплатформною мовою програмування, що означає, що веб-додатки, написані на C++, можуть бути запущені на різних операційних

системах без необхідності великих змін або перекомпіляції. Це полегшує розгортання та поширення додатків на різних серверах.

Наявність бібліотек та програмний каркасів: Існують деякі бібліотеки та програмний каркаси, які допомагають у розробці веб-додатків на C++. Наприклад, Wt є повнофункціональним програмний каркасом, який надає інструменти для створення веб-додатків з багатопотоковою обробкою, роботою з базами даних, шаблонами, обробкою запитів та іншими функціями.

Інтеграція з іншими мовами програмування: C++ може бути використаний як частина бекенду веб-додатка, інтегруючись з іншими мовами програмування. Наприклад, можна використовувати C++ для написання швидких алгоритмів або модулів, які потім можуть бути викликані з мови, таких як Python або JavaScript.

РОЗДІЛ 3 ОСНОВНІ ХАРАКТЕРИСТИКИ МОВИ C#

В той час, коли такі мови, як Python і PHP, існують досить тривалий час, C# вважається молодою мовою програмування. Данський інженер-програміст Андерс Хейлсберг розробив її у 2000 році. Сьогодні він продовжує працювати у компанії Microsoft як провідний архітектор C#. Андерс Хейлсберг також відомий як головний архітектор Delphi й перший автор Turbo Pascal.

Спочатку C# називалася COOL. Цей акронім в оригіналі походить від «C-style Object-Oriented Language», що означає «об'єктноорієнтована мова у стилі C». На жаль, компанія Microsoft не змогла зберегти цю «круту назву» (cool — англійською мовою має значення «крутий, класний» — Прим. ред.) через законодавство про торгові марки.

C# вимовляється «Сі-шарп». Назву взяли з музичної нотації, де символ «#» — октоторп або дієз — вказує на те, що ноту слід зіграти на півтону вище. Суфікс «шарп» також використовувався кількома іншими мовами програмування платформи .NET, а саме виданнями сучасних мов, наприклад, J#, A# та функціональна мова програмування F#.

Базовий синтаксис C# подібний до мов стилю C, таких як C, C++ і Java. Ця мова програмування найбільше відповідає стандарту Common Language Infrastructure (CLI).

3.1 Синтаксис мови C#

«C#» є більш сучасною мовою програмування, яка поєднує в собі елементи мов «C++» та «Java». Синтаксис «C#» також використовує крапку з комою (;) для закінчення виразів, фігурні дужки ({}) для групування блоків коду, ключові слова, оператори та вирази.

Структура мови «C#» базується на класах та об'єктах. Класи в «C#» описують об'єкти та включають поля, властивості, методи та конструктори. Класи можуть бути організовані в простори імен (namespaces), що дозволяє логічно групувати

класи відповідно до їх функціональності. Крім того, «C#» підтримує поняття інтерфейсів, які визначають контракти, які класи можуть реалізувати, та дають змогу реалізувати поліморфізм та спадкування.

Кожна програма C# має метод Main, який є вхідною точкою програми.

Код в C# розділяється на блоки за допомогою фігурних дужок {}.

Вирази та оператори розділяються крапкою з комою ;

Коментарі можуть бути однорядковими (за допомогою //) або багаторядковими (за допомогою /* ... */).

Змінні в C# повинні бути оголошені перед використанням, з вказанням їх типу.

Ключове слово var може використовуватись для визначення типу змінної на підставі присвоєнного значення.

Функції оголошуються за допомогою ключового слова void (якщо немає повертаючого значення) або типу даних, який вони повертають.

Структура C# програми:

- Підключення просторів імен (за допомогою директиви using).
- Оголошення глобальних змінних та констант.
- Оголошення та визначення класів, структур та інтерфейсів.
- Оголошення та визначення методів класу.
- Визначення властивостей (properties) та індексаторів.
- Оголошення та визначення подій (events).
- Оголошення та визначення делегатів (delegates).
- Оголошення та визначення додаткових класів, структур та інтерфейсів, які використовуються в програмі.
- Визначення додаткових методів, властивостей, подій та делегатів, які використовуються в програмі.
- Виклик методу Main, який є точкою входу в програму.

Це загальна структура програми на C#, інші частини програми можуть бути додані або змінені в залежності від потреб розробки.

3.2. Основні конструкції мови C#:

Змінні та типи даних: «C#» надає різноманітні вбудовані типи даних, включаючи цілі числа, дійсні числа, символи, рядки, булеві значення. Існують також розширені типи даних, такі як масиви, структури та класи.

Умовні конструкції: Як і в «C++», «C#» підтримує умовні конструкції, такі як if-else, switch, які дають змогу залежно від умови виконувати певні дії або гілки коду.

- Умовний оператор if: Використовується для виконання певного блоку коду, якщо задана умова істинна.

```
if (умова)
{
    // Блок коду, який виконується, якщо умова істинна
}
```

Рисунок 3.1 — Умовний оператор if.

- Умовний оператор else: Використовується для виконання певного блоку коду, якщо умова в операторі if не є істинною.

```
if (умова)
{
    // Блок коду, який виконується, якщо умова істинна
}
else
{
    // Блок коду, який виконується, якщо умова не є істинною
}
```

Рисунок 3.2 — Умовний оператор else

- Умовний оператор else if: Використовується для перевірки додаткових умов, якщо попередні умови не є істинними.

```
if (умова1)
{
    // Блок коду, який виконується, якщо умова1 істинна
}
else if (умова2)
{
    // Блок коду, який виконується, якщо умова2 істинна
}
else
{
    // Блок коду, який виконується, якщо жодна з умов не є істинною
}
```

Рисунок 3.3. — Умовний оператор else if.

Цикли: «C#» надає цикли, такі як for, while, do-while, які використовуються для повторення певного блоку коду. Цикли дають змогу ітерувати через масиви, колекції або виконувати певний код певну кількість разів.

- Цикл while: Використовується для повторення блоку коду, поки задана умова істинна.

```
while (умова)
{
    // Блок коду, який повторюється, поки умова істинна
}
```

Рисунок 3.4. — Цикл while

- Цикл do-while: Використовується для повторення блоку коду, поки задана умова істинна, принаймні один раз.

```
do
{
    // Блок коду, який повторюється, поки умова істинна
}
while (умова);
```

Рисунок 3.5. — Цикл do-while

- Цикл for: Використовується для повторення блоку коду зазначену кількість разів або на основі заданої послідовності.

```
for (ініціалізація; умова; ітерація)
{
    // Блок коду, який повторюється
}
```

Рисунок 3.6. — Цикл for

- Цикл foreach: Використовується для ітерації по елементам колекції або масиву.

```
foreach (тип елементу зі змінною ітерації in колекція)
{
    // Блок коду, який виконується для кожного елементу
}
```

Рисунок 3.7. — Цикл foreach

Конструкції для роботи з масивами та колекціями:

- Масиви: Масиви в C# дають змогу зберігати кілька значень одного типу даних у зручній формі.

```
тип_даних[] назва_масиву = new тип_даних[розмір];
назва_масиву[індекс] = значення;
```

Рисунок 3.8. — Приклад оголошення та використання масиву

- Колекції: Колекції в C# представляють гнучкі збереження та керування групами об'єктів.

```
List<тип_даних> назва_колекції = new List<тип_даних>();
назва_колекції.Add(елемент);
```

Рисунок 3.9. — Приклад використання колекції List

Конструкції для обробки виняткових ситуацій:

- Блок try-catch: Використовується для обробки виняткових ситуацій, що можуть виникнути під час виконання програми.

```
try
{
    // Блок коду, в якому може виникнути виняткова ситуація
}
catch (ТипВинятковоїСитуації1)
{
    // Обробка виняткової ситуації ТипВинятковоїСитуації1
}
catch (ТипВинятковоїСитуації2)
{
    // Обробка виняткової ситуації ТипВинятковоїСитуації2
}
finally
{
    // Блок коду, який виконується завжди, незалежно від того, чи виникла
}
```

Рисунок 3.10. — Блок try-catch

Методи та класи: «C#» базується на об'єктно-орієнтованому підході, тому він має багато можливостей для створення класів та методів. Класи дають змогу групувати дані та функціональність, а методи представляють дії, які можна виконати над об'єктами.

Ці конструкції використовуються для створення функціональних та ефективних веб-додатків на мовах «C++» і «C#». Розуміння цих конструкцій дозволяє розробникам ефективно використовувати можливості цих мов при розробці веб-додатків.

Конструкції для роботи з об'єктами та класами:

- Оголошення класу: Використовується для створення нового класу.

```
class Назва_класу
{
    // Поля, властивості, методи та ін.
}
```

Рисунок 3.11. — Оголошення класу

- Створення об'єкта: Використовується для створення нового екземпляра класу.

```
Назва_класу об'єкт = new Назва_класу();
```

Рисунок 3.12. — Створення об'єкта

- Властивості: Використовуються для забезпечення доступу до стану об'єкта.

```
тип_даних Назва_властивості { get; set; }
```

Рисунок 3.13. — Властивості об'єкта

- Методи: Використовуються для виконання певних дій або операцій.

```
тип_даних Назва_методу(параметри)
{
    // Блок коду методу
}
```

Рисунок 3.14. — Методи

Ці конструкції є основою мови C# і дають змогу розробникам створювати потужні, структуровані та ефективні програми. Знання цих конструкцій є важливим для ефективного програмування на C#.

3.3 Можливості веб програмування на C#

Мова програмування C# надає розширені можливості для веб-програмування. За допомогою програмний каркасів, таких як ASP.NET та ASP.NET Core, розробники можуть створювати потужні веб-додатки та сервіси. Ось деякі з основних можливостей веб-програмування на C#:

Веб-сервер: За допомогою ASP.NET та ASP.NET Core можна створювати веб-сервери, які надають з'єднання між клієнтами та сервером.

Ці програмний каркаси підтримують різні протоколи, такі як HTTP та HTTPS, і дають змогу розробникам створювати веб-додатки з різними функціональними можливостями.

Маршрутизація:ASP.NET та ASP.NET Core надають механізми маршрутизації, які дають змогу визначати, які дії або методи контролера повинні бути викликані при отриманні певного URL-запиту.

Це дозволяє розробникам зручно визначати логіку обробки запитів та керувати шляхами URL-адрес.

Модель-представлення-контролер (MVC):ASP.NET та ASP.NET Core використовують архітектурний шаблон MVC (Model-View-Controller), що дозволяє розподілити код на логічні компоненти.

Модель представляє дані, Представлення відповідає за відображення інтерфейсу користувача, а Контролер обробляє запити та взаємодіє з моделлю та представленням.

Цей шаблон сприяє структурованому розподілу коду та полегшує розробку та підтримку веб-додатків.

Шаблони сторінок та майстерні:ASP.NET та ASP.NET Core підтримують шаблони сторінок, які дають змогу розробникам створювати статичні та динамічні веб-сторінки з використанням розмітки та коду C#.

Крім того, вони надають майстерні, що спрощують створення певних типових елементів сторінок, таких як форми, таблиці, меню тощо.

Взаємодія з базами даних:C# має багатий набір інструментів для роботи з базами даних. Розробники можуть використовувати ADO.NET або Entity Framework для взаємодії з реляційними базами даних, такими як SQL Server, MySQL, PostgreSQL та іншими.

Це дозволяє зберігати та отримувати дані з баз даних для веб-додатків.

Безпека:ASP.NET та ASP.NET Core надають механізми для забезпечення безпеки веб-додатків, включаючи аутентифікацію та авторизацію користувачів.

Розробники можуть використовувати механізми ідентифікації та ролевого доступу для обмеження доступу до функціональності додатків.

Підтримка сторонніх бібліотек та сервісів: C# та .NET мають велику екосистему сторонніх бібліотек та сервісів, які можна використовувати у веб-додатках.

Розробники можуть використовувати бібліотеки для роботи з API, обробки зображень, розсилки електронних листів та багато іншого.

Це лише кілька можливостей веб-програмування на C#. Завдяки своїй потужності та багатому функціоналу, мова C# є популярним вибором для розробки веб-додатків та сервісів.

РОЗДІЛ 4 ПОРІВНЯЛЬНИЙ АНАЛІЗ МОВ C++ І C# У КОНТЕКСТІ ВЕБ ПРОГРАМУВАННЯ

У цьому розділі проведено детальне порівняння мов веб-програмування «C++» і «C#». Розглянуті різні аспекти, включаючи синтаксис, продуктивність, екосистему, підтримку, специфічні особливості та інші фактори, що впливають на веб-розробку.
Синтаксис:«C++» має складніший синтаксис порівняно з «C#». Він базується на мові «C», що робить його потужним, але вимагає більшої уваги до деталей і може мати більший поріг входження для нових розробників.
«C#» має простіший синтаксис, який більше нагадує мови програмування високого рівня. Він має багато удосконалень порівняно з мовою «C++», що полегшує розробку та зрозумілість коду.
Продуктивність:«C++» відома своєю високою продуктивністю, що робить його популярним для системного програмування та розробки продуктивних додатків. Він дозволяє ближче працювати з апаратним забезпеченням та має більш прямий доступ до системних ресурсів.
«C#»також є продуктивною мовою програмування, але може мати деяку втрату продуктивності порівняно з «C++» через додатковий шар абстракції та використання віртуальної машини «.NET.» Проте завдяки оптимізаціям та покращенням у виконанні, вона досить швидка для більшості веб-додатків.
Екосистема:«C++» має багато розширень, бібліотек та програмний каркасів, що сприяють веб-розробці. Проте, в порівнянні з «C#», екосистема «C++» менш розгалужена та менш спеціалізована на веб-розробці.
«C#» має потужну екосистему, зокрема «.NET Framework» та «.NET Core», які надають багато інструментів, програмний каркасів та бібліотек для розробки веб-додатків. «ASP.NET» і «ASP.NET Core» є популярними програмний каркасами для веб-розробки на базі «C#», що дають змогу розробникам швидко створювати потужні та масштабовані веб-додатки.

Підтримка:«C++» має широку підтримку від різних інструментів та платформ, але веб-розробка на «C++» може бути складнішою через меншу кількість спеціалізованих ресурсів та інструментів, порівняно з «C#».

«C#»має активну спільноту розробників та підтримку від Microsoft. Інструменти розробки, такі як Visual Studio, розширюють можливості розробки веб-додатків на «C#». Також, існують різні онлайн-ресурси та документація, що допомагають розробникам у веб-розробці на «C#».

Специфічні особливості:«C++» надає більший контроль над пам'яттю та низькорівневими операціями, що дозволяє оптимізувати продуктивність. Проте, це також може призводити до складнощів у вирішенні проблем безпеки та потенційних помилок.

«C#»забезпечує автоматичне управління пам'яттю та має механізми, що полегшують роботу зі збірником сміття. Це робить його більш безпечним та зручним для розробки веб-додатків, але може мати вплив на продуктивність у деяких сценаріях.

Усі ці фактори потрібно враховувати при виборі мови веб-програмування для конкретного проекту. При вирішенні, яка мова краще відповідає потребам проекту, варто звернутися до його вимог, розміру, продуктивності, зручності розробки та доступних ресурсів.

4.1 Швидкодія та продуктивність мови C++

Однією з головних переваг мови програмування C++ є її швидкодія та висока продуктивність. Ось деякі фактори, які сприяють цим характеристикам:

Компіляція: C++ є мовою програмування, яка компілюється безпосередньо в машинний код, що дозволяє досягти більшої продуктивності порівняно з інтерпретованими або мовами, що використовують віртуальну машину. Компіляція під час розробки дозволяє оптимізувати код для конкретної платформи та апаратного забезпечення, що покращує продуктивність.

Низькорівневий доступ до ресурсів: C++ надає можливість прямого доступу до пам'яті та системних ресурсів. Це дозволяє оптимізувати використання ресурсів, виконувати оптимізовані операції вводу-виводу та маніпулювати даними на низькому рівні, що сприяє збільшенню швидкодії програм.

Ефективна робота з пам'яттю: C++ дозволяє розміщувати дані у пам'яті та управляти життєвим циклом об'єктів вручну. Це дає можливість уникнути витоків пам'яті та надмірного використання ресурсів, а також забезпечує більш точне управління пам'яттю, що сприяє ефективності програм.

Потужна багатопотоковість: C++ надає потужні засоби для роботи з багатопотоковими додатками. З використанням бібліотек, таких як STL (Standard Template Library) або бібліотеки для роботи з потоками, можна легко створювати багатопотокові програми, що використовують паралельну обробку та забезпечують високу ефективність виконання.

Оптимізація та оптимізовані бібліотеки: Велика кількість оптимізованих бібліотек та програмний каркасів, доступних для C++, дозволяє використовувати швидкі та ефективні алгоритми для вирішення різних завдань. Наприклад, бібліотеки, такі як Boost, Eigen, Intel Math Kernel Library (MKL) та багато інших, пропонують оптимізовані функції для чисельних обчислень, обробки зображень, мережевого програмування та інших задач.

Загалом, мова програмування C++ володіє потужними засобами оптимізації, низькорівневим доступом до ресурсів та ефективною роботою з пам'яттю, що робить її однією з найшвидших і найпродуктивніших мов програмування. Однак, важливо мати на увазі, що для досягнення максимальної швидкодії та продуктивності важливо враховувати оптимізацію алгоритмів та правильне використання ресурсів. Вона компілюється безпосередньо в машинний код, що дозволяє оптимізувати виконання програми.

C# зазвичай виконується на віртуальній машині .NET, що може призвести до невеликого зниження швидкодії порівняно з C++. Однак, з випуском .NET Core та покращенням компілятора Just-In-Time (JIT), швидкодія C# значно покращилася.

4.2 Швидкодія та продуктивність мови C#

Мова програмування C# (C-Sharp) також має деякі переваги щодо швидкодії та продуктивності. Ось деякі фактори, які впливають на ці характеристики:
Компіляція JIT (Just-in-Time): Код на C# компілюється в проміжний байт-код, який потім виконується в середовищі виконання .NET (Common Language Runtime). В цьому випадку компіляція в машинний код відбувається в процесі виконання програми (JIT-компіляція). Це дозволяє оптимізувати код під конкретну платформу та апаратне забезпечення, що покращує швидкодію програми під час виконання.
Багатопотоковість: C# має потужні засоби для роботи з багатопотоковими додатками. За допомогою класів і бібліотек, таких як Task та Parallel, можна легко створювати багатопотокові програми, що виконують обчислення паралельно та покращують продуктивність виконання завдань.

Оптимізація JIT-компілятора: JIT-компілятор .NET може виконувати різні оптимізації під час компіляції в машинний код. Він може виявляти та оптимізувати часто використовувані шляхи виклику методів, виконувати інлайнінг (підстановку коду), проводити оптимізації пам'яті та багато іншого. Це сприяє поліпшенню продуктивності програми.
Використання оптимізованих бібліотек: В .NET-екосистемі доступно багато оптимізованих бібліотек для різних завдань, таких як обробка зображень, математичні обчислення, робота з базами даних та інші. Використання цих бібліотек дозволяє забезпечити високу продуктивність та швидкодію відповідних операцій.
Асинхронне програмування: C# має підтримку асинхронного програмування за допомогою ключових слів async та await. Це дозволяє створювати асинхронні операції, які не блокують головний потік виконання. З використанням асинхронності можна забезпечити ефективне використання ресурсів та покращити відгук системи.
Оптимізація базових бібліотек: Базові бібліотеки .NET, такі як .NET Framework та .NET Core, постійно оптимізуються та покращуються. Це дозволяє забезпечити

високу продуктивність у виконанні основних операцій, таких як робота з колекціями даних, обробка рядків, ввод-вивід та мережеві операції.

Використання нативного коду: У C# також є можливість використання нативного коду, наприклад, за допомогою інтероперабельності з мовою C++. Це дозволяє використовувати оптимізований нативний код для важливих компонентів або для оптимізації критичних за швидкодією частин програми.

Хоча C# може бути трохи повільнішою в порівнянні з мовою C++ або мови низькорівневого програмування, вона все ж надає достатньо швидкодії та продуктивності для багатьох завдань. Багатопотоковість, оптимізації JIT-компілятора, оптимізовані бібліотеки та асинхронне програмування є інструментами, які допомагають покращити продуктивність програм на C#.

4.3 Налагодження та відладка мови C++

Налагодження та відладка є важливими аспектами розробки програм на мові C++. Варто навести кілька основних засобів та практик, які допоможуть у відлагодженні програм на C++:

Використання відладчика: C++ має потужні відладчики, такі як GDB (GNU Debugger) або LLDB (LLVM Debugger), які надають розширені можливості відлагодження. Відладчики дають змогу крокувати по коду, переглядати значення змінних, встановлювати точки зупину, аналізувати стек викликів та багато іншого. Вони допомагають виявляти та виправляти помилки та проблеми у програмі.

Виведення на консоль: Вставка в код виведення повідомлень на консоль (наприклад, за допомогою функції std::cout у стандартній бібліотеці C++) є простим, але ефективним способом відлагодження. Ви можете вивести значення змінних, повідомлення про проміжні результати або статус виконання програми для з'ясування, де можуть бути проблеми.

Перевірка передумов та післяумов: Використання передумов (preconditions) та післяумов (postconditions) у вашому коді може допомогти виявляти проблеми та помилки. Передумови - це перевірки, які слід виконувати на початку функції,

наприклад, перевірка допустимих діапазонів аргументів. Післяумови - це перевірки, які слід виконувати перед виходом з функції, наприклад, перевірка очікуваних результатів. Використання передумов та післяумов допомагає виявляти некоректну поведінку та допомагає у відлагодженні.

Запис у журнал (logging): Використання системи журналювання (logging) дозволяє записувати повідомлення про події, помилки та стан програми у файл або консоль. Це може бути корисно при відлагодженні довготривалих або розподілених програм, де виведення на консоль не є практичним. Запис у журнал може надати додаткову інформацію про стан програми та його виконання.

Тестування: Написання тестів (unit tests) допомагає виявляти проблеми у коді та підтверджувати правильність його роботи. Розробка тестів дозволяє вам перевіряти окремі фрагменти коду на очікувані результати та виявляти помилки раніше. Тестування може бути автоматизоване та включати широкий спектр сценаріїв та вхідних даних.

Використання цих засобів та практик у поєднанні з добрим розумінням мови C++ допоможе вам ефективно відлагоджувати програми, знаходити та виправляти проблеми, що виникають у вашому коді.

4.4 Налагодження та відладка мови C#

Налагодження та відладка є важливими аспектами розробки програм на мові C#. Ось деякі основні засоби та практики, які допоможуть у відлагодженні програм на C#:

Використання відладчика: Основним інструментом для відлагодження програм на C# є вбудований відладчик у середовищі розробки, наприклад, в Visual Studio або Visual Studio Code. Відладчик дозволяє встановлювати точки зупину, крокувати по коду, переглядати значення змінних та об'єктів, аналізувати стек викликів та виконувати багато інших дій для відлагодження програми.

Виведення на консоль або у файл: Якщо ви не можете використовувати відладчик або хочете простіший спосіб відлагодження, виведення повідомлень на консоль або

у файл може бути корисним. Ви можете використовувати функцію Console.WriteLine для виведення повідомлень у консоль або використовувати бібліотеки журналювання (logging libraries), такі як Serilog або NLog, для записування повідомлень у файл.

Використання винятків: Використання винятків (exceptions) дозволяє виявляти та обробляти помилки у програмі. Ви можете створювати та викидати винятки, коли виникають некоректні стани або ситуації, і використовувати блоки try-catch-finally для обробки цих винятків та відлагодження проблемних ділянок коду.

Використання інструментів для профілювання: Інструменти для профілювання, такі як dotTrace, ANTS Performance Profiler або Visual Studio Profiler, дають змогу вам аналізувати продуктивність вашої програми. Вони надають інформацію про використання пам'яті, швидкодію методів та інші показники, що допомагають знайти та оптимізувати проблемні частини коду.

Тестування: Написання тестів (unit tests) допомагає виявляти проблеми у вашому коді та підтверджувати правильність його роботи. З використанням програмний каркасів для тестування, таких як NUnit або xUnit, ви можете автоматизувати процес виконання тестів та перевіряти різні сценарії та вхідні дані.

Ці засоби та практики допомагають відлагоджувати програми на C# та виявляти і виправляти проблеми у коді. Вони дають змогу зосередитися на пошуку та виправленні помилок, покращенні продуктивності та забезпеченні правильності роботи вашої програми.

4.5 Надійність та безпека мови C++

Мова програмування C++ надає можливості для реалізації надійних та безпечних програм. Ось декілька основних аспектів, які сприяють надійності та безпеці мови C++:

Контроль пам'яті: Одна з відмінних особливостей C++ - це можливість прямого керування пам'яттю. Однак, це також може бути потенційною причиною помилок, таких як витоки пам'яті, помилки доступу до пам'яті або некоректна робота з

покажчиками. Однак, у C++ є ряд інструментів та практик, які допомагають уникнути цих проблем, таких як використання смарт-покажчиків (smart pointers) для автоматичного керування пам'яттю, правильне використання операторів new та delete, а також застосування стандартних бібліотек для роботи з контейнерами та рядками, які автоматично керують пам'яттю.

Перевірка меж масивів: C++ не надає вбудованих перевірок меж масивів під час доступу до елементів. Це може призвести до виходу за межі масиву, що є потенційно небезпечним. Проте, ви можете використовувати стандартні бібліотеки, такі як STL (Standard Template Library), які надають безпечні альтернативи для роботи з масивами, наприклад, вектори (std::vector) або рядки (std::string), які автоматично перевіряють межі доступу.

Обробка винятків: C++ надає можливості для обробки винятків, що дозволяє виявляти та обробляти помилки та некоректні стани програми. Використання винятків дозволяє контролювати поток виконання програми та виконувати дії в разі виникнення помилок. Однак, правильна обробка винятків вимагає від програміста уваги та дбайливості, щоб уникнути витоку пам'яті або некоректного стану програми.

Перевірка типів: C++ є статично типізованою мовою, що дозволяє виявляти багато помилок на етапі компіляції. Перевірка типів допомагає уникнути помилок, пов'язаних з некоректним використанням типів даних, неправильними приведеннями типів або використанням неіснуючих методів чи змінних.

Бібліотеки та програмний каркаси: У C++ існує широкий спектр стандартних бібліотек та сторонніх програмний каркасів, які сприяють надійності та безпеці програм. Наприклад, Boost - це популярна стороння бібліотека, яка надає різноманітні функціональні можливості, включаючи роботу зі рядками, масивами, потоками, мережевими операціями та багато іншого. Також існують спеціалізовані бібліотеки, які забезпечують безпеку програмування, такі як Crypto++ для криптографії або POCO для мережевого програмування.

Ці аспекти спільно забезпечують надійність та безпеку програм, розроблених на мові C++. Проте, варто пам'ятати, що правильна реалізація цих практик і

використання відповідних інструментів залежить від уважності та досвіду програміста.C++ дозволяє розробникам мати більший контроль над ресурсами та пам'яттю, що може впливати на надійність програми. Проте, через використання вказівників та низькорівневих операцій, помилки програміста можуть призводити до вразливостей безпеки.

4.6 Надійність та безпека мови C

Мова програмування C# має вбудовані функції та механізми, які сприяють надійності та безпеці програм. Ось декілька основних аспектів, які сприяють надійності та безпеці мови C#:

Управління пам'яттю: В C# використовується система управління пам'яттю, яка автоматично відслідковує та вивільняє пам'ять, звільняючи програміста від необхідності явно керувати пам'яттю. Це допомагає уникнути багатьох проблем, пов'язаних з пам'яттю, таких як витоки пам'яті або помилки доступу до пам'яті.

Виключення та обробка помилок: C# має потужну систему обробки винятків, що дозволяє виявляти та обробляти помилки у програмі. Ви можете використовувати блоки try-catch-finally для обробки винятків та виконання відповідних дій в разі їх виникнення. Це дозволяє контролювати потік виконання програми та забезпечувати надійну обробку помилок.

Статична типізація: C# є статично типізованою мовою, що дозволяє виявляти багато помилок на етапі компіляції. Перевірка типів допомагає уникнути помилок, пов'язаних з некоректним використанням типів даних, неправильними приведеннями типів або використанням неіснуючих методів чи змінних.

Захист від вразливостей: C# має ряд вбудованих механізмів та бібліотек, які допомагають уникати вразливостей програм. Наприклад, в C# є можливість застосовувати засоби криптографії для забезпечення безпеки даних, використовувати захищені з'єднання з базами даних, перевіряти та валідувати вхідні дані для запобігання атакам врядування коду (code injection) або переповненню буфера (buffer overflow).

Тестування: C# має різні програмний каркаси для написання та виконання тестів (наприклад, NUnit, xUnit), що допомагають виявляти помилки та підтверджувати правильність роботи коду. Тестування є важливим аспектом надійності та безпеки програм і дозволяє перевірити різні сценарії та вхідні дані.

Ці аспекти спільно сприяють надійності та безпеці програм, розроблених на мові C#

4.7 Підтримка інтегрованих середовищ розробки мови C++

Мова програмування C++ має підтримку різноманітних інтегрованих середовищ розробки (IDEs), які надають зручні та потужні інструменти для розробки програм на C++. Ось кілька популярних інтегрованих середовищ розробки, які підтримують мову C++:

Visual Studio: Visual Studio від Microsoft є одним з найпопулярніших інтегрованих середовищ розробки для C++. Воно надає широкий спектр функціональності, такий як редактор коду з підсвічуванням синтаксису, автодоповненням та інтегрованим дебагером. Visual Studio також має багато інших корисних інструментів, таких як профілер, систему керування версіями, та підтримку для розробки веб-та мобільних додатків.

CLion: CLion, розроблений компанією JetBrains, є спеціалізованим інтегрованим середовищем розробки для мови C++. Воно надає багато корисних функцій, таких як інтелектуальне автодоповнення коду, рефакторинг, підтримка CMake та інші. CLion також має інтегрований дебагер та можливості для тестування коду.

Qt Creator: Qt Creator є інтегрованим середовищем розробки, спеціалізованим для розробки програм з використанням програмному каркасу Qt. Воно надає зручні інструменти для розробки інтерфейсу користувача, редактор коду з підсвічуванням синтаксису, автодоповненням та дебагером. Qt Creator також має можливості для розробки мобільних та вбудованих додатків.

Xcode: Xcode є інтегрованим середовищем розробки, розробленим для платформи macOS та iOS. Воно підтримує розробку програм на мові C++ та має інструменти

для розробки додатків для macOS, iOS, watchOS та tvOS. Xcode надає редактор коду, дебагер, інструменти для профілювання та багато іншого.

Ці інтегровані середовища розробки надають зручні та потужні інструменти для розробки програм на C++. Вони спрощують процес розробки, покращують продуктивність програміста та надають можливості для налагодження, аналізу та тестування коду. Вибір конкретного інтегрованого середовища розробки залежить від особистих потреб та вподобань.

4.8 Підтримка інтегрованих середовищ розробки мови C#

Мова програмування C# теж має розширену підтримку відомих інтегрованих середовищ розробки (IDEs), які надають зручні та потужні інструменти для розробки програм на C#. Ось кілька популярних інтегрованих середовищ розробки, які підтримують мову C#:

Visual Studio: Visual Studio є основним інтегрованим середовищем розробки для C#. Воно надає широкий набір функціональності, включаючи редактор коду з підсвічуванням синтаксису, автодоповненням та інтегрованим дебагером. Visual Studio також має інструменти для розробки веб-додатків, мобільних додатків, хмарних додатків та багато іншого.

JetBrains Rider: Rider є інтегрованим середовищем розробки, розробленим компанією JetBrains. Воно надає потужні інструменти для розробки програм на C#, включаючи редактор коду з інтелектуальним автодоповненням, рефакторингом, підтримкою управління пакетами NuGet та вбудованим дебагером. Rider також підтримує розробку ASP.NET, ASP.NET Core, Xamarin та Unity.

Visual Studio Code: Visual Studio Code (VS Code) є легким інтегрованим середовищем розробки, яке надає розширену підтримку для мови C#. Воно має багато розширень, які дають змогу налаштувати середовище розробки під власні потреби. VS Code надає можливості для редакції коду, дебагу, керування версіями, інтеграції з Git та багато іншого.

MonoDevelop: MonoDevelop є відкритим інтегрованим середовищем розробки, яке спеціалізується на розробці програм з використанням технологій Mono та Xamarin. Воно надає інструменти для розробки кросплатформових додатків на основі мови C#.

Ці інтегровані середовища розробки надають зручність, продуктивність та потужність для розробки програм на мові C#. Вони підтримують функціональність, таку як підсвічування синтаксису, автодоповнення, дебагери та багато іншого, що спрощує процес розробки.

4.9 Масштабованість та розширюваність мови C++

Мова програмування C++ є мовою загального призначення, яка має широкі можливості щодо масштабованості та розширюваності програм. Ось кілька аспектів, які впливають на масштабованість та розширюваність мови C++:

Об'єктно-орієнтоване програмування: C++ підтримує об'єктно-орієнтоване програмування, що дозволяє створювати класи та об'єкти зі своїми власними методами та властивостями. Це сприяє організації програм у модульну структуру, що полегшує масштабування та розширення коду.

Шаблони: C++ має потужну систему шаблонів, яка дозволяє писати загальні алгоритми та структури даних, що можуть бути параметризовані типами. Шаблони дають змогу створювати загальні рішення, які можна використовувати для різних типів даних, що забезпечує гнучкість та розширюваність коду.

Бібліотеки: C++ має багато розширюваних бібліотек, які надають готові реалізації різних функціональних можливостей. Ці бібліотеки дають змогу розширювати функціональність програм шляхом використання готових компонентів, що спрощує розробку та забезпечує масштабованість.

Взаємодія з іншими мовами: C++ має можливість взаємодіяти з іншими мовами програмування, такими як C, Python, Java та інші. Це дозволяє використовувати різні мови для різних частин проекту та забезпечує гнучкість та масштабованість розробки.

Низькорівневі можливості: C++ надає доступ до низькорівневих операцій та функцій, таких як керування пам'яттю, маніпуляція бітами та оптимізація коду. Це дозволяє розробникам максимально контролювати ресурси та оптимізувати продуктивність програм, що важливо для масштабованих систем.

Паралелізм: C++ має підтримку паралельного програмування, включаючи можливості багатопоточності, мультипотоковості та розподіленого обчислення. Це дозволяє використовувати потужність сучасних багатоядерних та розподілених систем для ефективної обробки завдань.

4.10 Масштабованість та розширюваність мови C#

Мова програмування C# має вбудовану підтримку для масштабування та розширюваності програм. Ось кілька аспектів, які сприяють цим характеристикам:

Об'єктно-орієнтоване програмування: C# є об'єктно-орієнтованою мовою, що дозволяє створювати класи, об'єкти, інтерфейси та спадкування. Це сприяє організації програм у модульну структуру, де окремі компоненти можна розширювати та змінювати без впливу на інші частини системи.

Інтерфейси та абстракції: C# підтримує використання інтерфейсів та абстрактних класів, що дають змогу визначати контракти та стандартизувати поведінку. Це спрощує розширення функціональності системи шляхом реалізації нових класів, які відповідають існуючим інтерфейсам або наслідуються від абстрактних класів.

Розширення методів: C# дозволяє розширювати функціональність існуючих типів за допомогою механізму розширення методів. Це означає, що ви можете створювати нові методи для існуючих типів без необхідності модифікувати їх вихідний код. Це дозволяє додавати нові функціональні можливості до класів, навіть якщо у вас немає доступу до їх вихідного коду.

Компонентна модель: C# підтримує компонентну модель розробки, що дозволяє створювати компоненти, які можуть бути повторно використані та легко інтегровані в різні проекти. Це спрощує розширення системи шляхом додавання нових компонентів або заміни існуючих без необхідності модифікації весь код.

Пакети та залежності: C# має підтримку пакетного менеджера NuGet, який дозволяє легко керувати залежностями програмного забезпечення. Ви можете використовувати пакети з NuGet для розширення функціональності вашої програми шляхом додавання готових бібліотек та компонентів.

Взаємодія з платформою .NET: C# базується на платформі .NET, яка надає широкий спектр функціональних можливостей та багато вбудованих бібліотек. Це дозволяє легко розширювати програми за допомогою використання функціональності, яка вже доступна в платформі.

Загалом, мова C# має потужні засоби для масштабування та розширення програм. Це дозволяє легко створювати модульні, гнучкі та розширювані системи, що задовольняють зростаючі потреби проектів.

.4.11 Наявність стандартних бібліотек та програмний каркасів мови C++

Мова програмування C++ має багато стандартних бібліотек і програмний каркасів, які надають різноманітні функціональні можливості та допомагають в розробці програм, наприклад:

Стандартна бібліотека C++ (STL): Це основна бібліотека, яка входить до стандарту мови C++. Вона надає широкий спектр загальноприйнятих контейнерів (таких як вектори, списки, асоціативні масиви), алгоритмів (таких як сортування, пошук, маніпуляція даними) і функцій обробки рядків, введення/виведення та інших корисних інструментів.

Boost: Це потужний набір кросплатформових бібліотек для C++, який розширює функціональність мови. Boost включає в себе бібліотеки для роботи зі рядками, файлами, мережевими протоколами, числами, алгоритмами, серіалізації даних та багато іншого. Вона широко використовується у C++-проектах для покращення продуктивності та розширення функціональності.

Qt: Qt є кросплатформовим програмний каркасом, який дозволяє розробляти графічні інтерфейси користувача (GUI) та кросплатформні додатки на C++. Він надає багато готових компонентів для розробки програмного забезпечення, таких як

вікна, кнопки, меню, таблиці тощо. Qt також надає можливості для роботи з мережами, базами даних, зображеннями та багато іншого.

OpenGL: Це відкрите стандартне API для рендерингу 2D та 3D графіки. C++ має потужну підтримку для OpenGL, що дозволяє створювати швидкі та ефективні графічні програми. OpenGL дозволяє взаємодіяти з графічним обладнанням комп'ютера, що дозволяє розробникам створювати складні візуалізації та графічні ефекти.

OpenCV: Це бібліотека комп'ютерного зору, яка надає функціональність для обробки зображень та відео. OpenCV дозволяє розпізнавати об'єкти, виконувати розпізнавання облич, розпізнавати жести, вимірювати відстані та багато іншого. Вона широко використовується у проектах комп'ютерного зору та машинного навчання.

Це лише декілька прикладів стандартних бібліотек та програмний каркасів, які використовуються в мові C++. Враховуючи широку популярність мови, існує також багато інших сторонніх бібліотек та програмний каркасів, які розширюють функціональність та допомагають у розробці програм на C++.

4.12 Наявність стандартних бібліотек та програмний каркасів мови C#

C# також має широкий вибір стандартних бібліотек та програмний каркасів ,наприклад:

.NET Framework: Це один з найбільш використовуваних програмний каркасів для розробки програм на C#. Він надає велику кількість класів і бібліотек для різних потреб, таких як робота з рядками, колекціями, файлами, мережами, базами даних, шифруванням, графікою та багато іншого. .NET Framework також має розширення, такі як ASP.NET для розробки веб-додатків та Windows Presentation Foundation (WPF) для розробки графічних десктопних додатків.

.NET Core: Це кросплатформовий програмний каркас, який є наступником .NET Framework. Він надає спрощену та легку версію .NET Framework і може використовуватись для розробки програм на C# на різних платформах, включаючи

Windows, macOS та Linux. .NET Core також включає ASP.NET Core для розробки кросплатформних веб-додатків.

ASP.NET: Це програмний каркас для розробки веб-додатків на C#. ASP.NET надає широкі можливості для створення веб-сайтів, веб-сервісів та інших веб-додатків. Він має вбудовану підтримку для моделі MVC (Model-View-Controller) та моделі Web API для створення RESTful веб-сервісів. ASP.NET також надає засоби для роботи з базами даних, безпеки, маршрутизації, кешування та іншого.

Entity Framework: Це ORM (Object-Relational Mapping) програмний каркас для роботи з базами даних в програмах на C#. Він надає спрощений спосіб взаємодії з базами даних, дозволяючи використовувати об'єктно-орієнтовану модель для доступу до даних. Entity Framework автоматично генерує SQL-запити та виконує маппінг об'єктів на таблиці бази даних.

Xamarin: Це програмний каркас, що дозволяє розробляти кросплатформні мобільні додатки на C#. Він дозволяє створювати додатки для платформ Android та iOS, використовуючи спільний код на C#. Xamarin надає доступ до нативних API та можливості для створення інтерфейсу користувача, доступу до баз даних та іншого.

Як і C++ , C# має ще безліч стандартних бібліотек та програмний каркасів, які використовуються у розробці програм. За допомогою цих інструментів можна розробити різноманітні додатки, включаючи веб-додатки, десктопні програми, мобільні додатки та багато іншого.

Загалом, обидві мови C++ та C# мають свої переваги та області застосування у веб програмуванні. Вибір між ними залежить від конкретних потреб проекту, швидкодії, надійності, безпеки, розширюваності та наявності потрібних інструментів та бібліотек.

РОЗДІЛ 5 ПОРІВНЯННЯ ХАРАКТЕРИСТИК ПРОЕКТІВ

C++ використовується у багатьох промислових проектах, особливо там, де вимагається висока продуктивність, доступ до низькорівневих ресурсів та контроль над пам'яттю. Він часто застосовується в розробці вбудованих систем, компіляторів, графічних двигунів та ігор.C# також знаходить широке застосування у промислових проектах, зокрема веб-додатках, мобільних додатках, системах управління базами даних та багатьох інших. Він є популярним в розробці додатків для платформи .NET, включаючи ASP.NET та Xamarin.

5.1 Результати ефективності та продуктивності:

Ефективність та продуктивність проектів на C++ може бути дуже високою, оскільки він дозволяє розробникам мати прямий доступ до апаратних ресурсів та використовувати оптимізовані алгоритми. Великі проекти, що вимагають обробки великої кількості даних або високої продуктивності, часто вибирають C++.
C# забезпечує швидкий процес розробки завдяки своїм вбудованим інструментам та програмний каркасам. Він надає високорівневі абстракції та готові компоненти, що спрощують розробку та зменшують час розробки. Проекти, де пріоритетом є швидкість розробки та простота утримання коду, часто вибирають C#.

5.4 Аналіз переваг та недоліків застосування мови C++

C++ є досить складною мовою програмування, вона вимагає від розробників глибокого розуміння понять, таких як показники, деструктори, управління пам'яттю та інші. Це може збільшити час розробки та складність розуміння коду. Завдяки прямому керуванню пам'яттю, C++ може бути вразливим до помилок, таких як витоки пам'яті, неправильний доступ до пам'яті, переповнення буфера та інші.

Некоректний використання пам'яті може призвести до нестабільної роботи програми та потенційних вразливостей безпеки. C++ не має вбудованого автоматичного збирання сміття, що означає, що розробники повинні вручну керувати життєвим циклом об'єктів та відповідно вивільняти пам'ять. Це може призвести до складнішого управління пам'яттю та потенційних проблем з витоками пам'яті. Щодо переваг то розробники мають прямий контроль над керуванням пам'яттю, що дозволяє ефективно використовувати ресурси та уникнути витоків пам'яті. C++ підтримує об'єктно-орієнтований підхід, дозволяючи розробникам створювати класи, об'єкти та наслідування. Це полегшує розширення та повторне використання коду. Мова C++ дозволяє створювати кросплатформні програми, що працюють на різних операційних системах, таких як Windows, macOS та Linux. (Таблиця 5.1)

Таблиця 5.1

Переваги	Недоліки
• Швидкодія • Керування пам'яттю • Розширюваність • Кросплатформність • Багатопотоковість • Широкий вибір бібліотек та розширень • Підтримка розробки вбудованих систем та графічних додатків	• Складність • Вразливість до помилок Відсутність автоматичного збирання сміття Висока вимогливість до розробників

5.3 Аналіз переваг та недоліків застосування мови C#

Хоча C# є високорівневою мовою програмування, вона може мати обмежену продуктивність порівняно з мовами нижчого рівня, такими як C++. Це може бути

проблемою у випадках, коли необхідна максимальна швидкодія. Розробка на C# зазвичай вимагає встановлення платформи .NET Framework або .NET Core. Це може створювати деякі обмеження, особливо при розробці для вбудованих систем або окремих платформ. Хоча C# може використовуватися для розробки веб-додатків за допомогою програмний каркаса ASP.NET, існують інші мови, такі як JavaScript, які мають більшу популярність та підтримку для веб-розробки. C# тісно пов'язана з екосистемою Microsoft, тому розробка на C# може бути більш залежною від інструментів та технологій, наданих Microsoft. Це може обмежити свободу вибору для розробників. Незважаючи на недоліки C# має багато переваг, наприклад, мова C# має простий синтаксис та високий рівень абстракції, що робить її легкою для вивчення та розуміння. Вона надає розробникам потужні інструменти для швидкого створення програм. C# підтримує об'єктно-орієнтований підхід, що дозволяє розробникам створювати модульні програми з високим рівнем повторного використання коду. Це полегшує розширення та підтримку великих проектів. Мова C# використовує систему керування пам'яттю з автоматичним збиранням сміття, що зменшує витрати часу на ручне керування пам'яттю. Це спрощує розробку та дозволяє уникнути багатьох проблем, пов'язаних з керуванням пам'яттю. C# підтримує платформу .NET, яка надає можливість розробляти кросплатформні програми, що працюють на різних операційних системах, таких як Windows, macOS та Linux. Мова C# має вбудовану систему безпеки, включаючи перевірку типів під час компіляції та механізми обробки винятків. Це сприяє створенню більш надійних та безпечних програм.

Вибір між C++ та C# для промислових проектів залежить від конкретних вимог проекту, його масштабу, часу розробки та вимог до продуктивності. Обидві мови мають свої сильні сторони та області застосування, і правильний вибір залежить від унікальних потреб проекту та відповідних факторів. (Таблиця 5.2)

Таблиця 5.2

Переваги	Недоліки
• Простота використання • Масштабованість • Керування пам'яттю • Кросплатформність • Безпека	• Обмежена продуктивність • Залежність від платформи .NET • Обмеження веб-розробки Висока залежність від інструментів Microsoft:

5.4 Рекомендації щодо вибору мови веб програмування

Вибір мови програмування для веб-розробки має велике значення, оскільки це визначає ефективність, продуктивність та якість розробленого веб-додатка. Різні мови мають свої особливості, синтаксис та функціональні можливості, які можуть впливати на швидкодію, надійність, розширюваність та інші аспекти проекту. Тому, правильний вибір мови є критичним для досягнення поставлених цілей та успіху проекту. Якщо потрібна висока продуктивність, доступ до низькорівневих ресурсів та контроль над пам'яттю, а також ви плануєте розробляти вбудовані системи або графічні додатки, то C++ може бути кращим вибором. Якщо ви цінуєте швидкий процес розробки, простоту утримання коду та маєте низькі вимоги до системних ресурсів, то C# є гарним варіантом. Він також підходить для розробки веб-додатків, мобільних додатків та систем управління базами даних. Крім того, рекомендується враховувати такі фактори:

Розмір та складність проекту: Для великих проектів з великою кількістю коду та потребою в оптимізації C++ може бути кращим вибором. Для менших проектів або проектів з короткими термінами розробки C# може бути ефективнішим.

Екосистема та підтримка: Варто перевірити, які бібліотеки, програмний каркаси та інструменти доступні для кожної мови, а також яка підтримка і документація є відповідно для C++ та C#.

Командний склад та навички: Варто врахувати навички команди розробників. Якщо вони вже мають досвід у роботі з однією мовою, це може спростити розробку та підтримку проекту.

Кінцевий вибір між C++ та C# для веб програмування залежить від конкретних вимог проекту та ресурсів, які ви маєте. Потрібно ретельно зважати на переваги та недоліки кожної мови, а також на особливості проекту, щоб зробити найкращий вибір для успішної веб-розробк

ВИСНОВОК

У дипломній роботі на тему «Порівняльна характеристика мов веб програмування C++ і C#» були проведені детальний аналіз та порівняння двох мов програмування - C++ і C# - з точки зору веб-розробки. Отримані результати дають змогу зробити наступні висновки:

Швидкодія та продуктивність: Обидві мови мають свої переваги та недоліки щодо швидкодії та продуктивності. C++, як мова нижчого рівня, може забезпечити вищу швидкодію, але вимагає більшої уваги до деталей та керування пам'яттю. C#, з іншого боку, надає вищий рівень абстракції та має вбудовану систему автоматичного збирання сміття, що полегшує розробку, але може мати меншу швидкодію порівняно з C++.

Надійність та безпека: C++ та C# мають різні підходи до надійності та безпеки програм. C++ надає більший контроль над ресурсами та пам'яттю, але може бути вразливим до помилок у використанні. C# має вбудовані механізми безпеки, такі як перевірка типів під час компіляції та обробка винятків, що сприяє створенню більш безпечних програм.

Підтримка інтегрованих середовищ розробки: Обидві мови мають підтримку популярних інтегрованих середовищ розробки, таких як Visual Studio. C++ має багато інструментів та плагінів для розробки, що полегшує роботу з проектами. C# має високоінтегровану розробку у середовищі Visual Studio, що сприяє зручності та продуктивності розробки.

Розширюваність та масштабованість: Обидві мови мають підтримку об'єктно-орієнтованого підходу, що сприяє розширюваності та повторному використанню коду. Проте, C# має покращену масштабованість завдяки вбудованій платформі .NET, яка дозволяє розробляти кросплатформні програми та має багатий вибір програмний каркасів та бібліотек для розширення функціональності.

Враховуючи ці фактори, можна зробити висновок, що обидві мови мають свої переваги та недоліки в контексті веб-розробки. Вибір мови залежить від специфіки проекту, вимог до швидкодії, надійності та особистих навичок розробника. Обидві мови мають потужний набір інструментів та можуть бути успішно використані для розробки веб-програм.

Рекомендації щодо вибору між C++ та C# для веб програмування.

Якщо потрібна висока продуктивність, доступ до низькорівневих ресурсів та контроль над пам'яттю, а також в планах розробляти вбудовані системи або графічні додатки, то C++ може бути кращим вибором. Якщо в пріоритеті швидкий процес розробки, простоту утримання коду та низькі вимоги до системних ресурсів, то C# є гарним варіантом. Він також підходить для розробки веб-додатків, мобільних додатків та систем управління базами даних. Окрім того, рекомендується враховувати такі фактори:

Розмір та складність проекту: Для великих проектів з великою кількістю коду та потребою в оптимізації C++ може бути кращим вибором. Для менших проектів або проектів з короткими термінами розробки C# може бути ефективнішим.

Екосистема та підтримка: Варто перевірити, які бібліотеки, програмний каркаси та інструменти доступні для кожної мови, а також яка підтримка і документація є відповідно для C++ та C#.

Командний склад та навички: Варто врахувати навички команди розробників. Якщо вони вже мають досвід у роботі з однією мовою, це може спростити розробку та підтримку проекту.

Кінцевий вибір між C++ та C# для веб програмування залежить від конкретних вимог проекту та ресурсів, які ви маєте. Потрібно ретельно зважати на переваги та недоліки кожної мови, а також на особливості проекту, щоб зробити найкращий вибір для успішної веб-розробки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документація C++ [Електронний ресурс] – Режим доступу до ресурсу: https://en.cppreference.com/
2. Документація C# [Електронний ресурс] – Режим доступу до ресурсу:https://docs.microsoft.com/en-us/dotnet/csharp/
3. "C++ Primer" авторства Stanley B. Lippman, Josée Lajoie, Barbara E. Moo
4. "The C++ Programming Language" Bjarne Stroustrup
5. "C# 9 and .NET 5 - Modern Cross-Platform Development" Mark J. Price
6. Stack Overflow [Електронний ресурс] – Режим доступу до ресурсу: https://stackoverflow.com/
7. Microsoft Developer Network [Електронний ресурс] – Режим доступу до ресурсу: https://docs.microsoft.com/
8. C++ Reference [Електронний ресурс] – Режим доступу до ресурсу:https://www.cplusplus.com/
9. "ACM Transactions on Programming Languages and Systems"
10. "Journal of Object Technology"
11. Стаття "C# vs C++: A Battle of Programming Languages"