MỤC LỤC

1. GIỚI THIỆU 2

2. CÀI ĐẶT VÀ CẤU HÌNH CPPUNIT 2

2.1. Tải xuống CppUnit 2

2.2. Biên dịch CppUnit trong Visual Studio hoặc CMD 3

2.3. Cấu hình Dự án trong Visual Studio 4

3. DEMO 5

3.1. Mô tả Lớp Số Nguyên 5

3.2. Thiết kế Test Case cho Các Phép Toán 5

3.3. Chạy Test và Phân tích Kết quả 6

4. CÁC VẤN ĐỀ THƯỜNG GẶP VÀ CÁCH KHẮC PHỤC 6

5. KẾT LUẬN 7

6. TÀI LIỆU THAM KHẢO 8

# 1. GIỚI THIỆU

Kiểm thử đơn vị (unit testing) là một phương pháp quan trọng trong phát triển phần mềm, giúp đảm bảo các thành phần mã nguồn hoạt động chính xác trước khi tích hợp vào hệ thống lớn hơn. CppUnit là một framework kiểm thử đơn vị mã nguồn mở, được thiết kế để hỗ trợ lập trình viên C++ viết và thực thi các bài kiểm thử một cách có tổ chức. Framework này cho phép phát hiện lỗi sớm, cải thiện chất lượng mã, và phù hợp với các phương pháp phát triển hiện đại như Test-Driven Development (TDD).

Báo cáo này tập trung vào việc sử dụng CppUnit trong Visual Studio để kiểm thử các phép toán cơ bản (cộng, trừ, nhân, chia) của một lớp số nguyên. Mục tiêu là cung cấp hướng dẫn chi tiết về cách cài đặt, cấu hình, và áp dụng CppUnit để kiểm tra tính đúng đắn của các phép toán này. Báo cáo cũng mô tả cách thiết kế các bài kiểm thử, chạy chúng trong Visual Studio, và xử lý các lỗi thường gặp, nhằm minh họa cách áp dụng CppUnit trong thực tiễn.

**Mục tiêu của báo cáo**:

* Hướng dẫn cài đặt và tích hợp CppUnit vào Visual Studio trên Windows, bao gồm sử dụng Command Prompt (CMD).
* Mô tả cách thiết kế và kiểm thử các phép toán cộng, trừ, nhân, chia của một lớp số nguyên.
* Phân tích kết quả kiểm thử và cách xử lý các vấn đề phát sinh.
* Đánh giá hiệu quả của CppUnit và đề xuất cải tiến.

# 2. CÀI ĐẶT VÀ CẤU HÌNH CPPUNIT

## 2.1. Tải xuống CppUnit

CppUnit có thể được tải từ trang web chính thức trên SourceForge hoặc kho mã nguồn Git của LibreOffice. Phiên bản mới nhất hiện tại là CppUnit 1.15.1, được khuyến nghị để đảm bảo các bản vá lỗi và tính năng mới nhất.

Quá trình tải xuống bao gồm:

* Truy cập trang SourceForge để tải file nén (.zip) hoặc sử dụng lệnh Git để lấy mã nguồn.
* Giải nén file vào một thư mục cố định trên Windows, ví dụ: thư mục C:\cppunit-1.15.1.
* Kiểm tra tính toàn vẹn của file tải về bằng các giá trị băm (hash) như MD5 (9dc669e6145cadd9674873e24943e6dd) hoặc SHA256 (89c5c6665337f56fd2db36bc3805a5619709d51fb136e51937072f63fcc717a7) để đảm bảo không bị lỗi hoặc giả mạo.

## 2.2. Biên dịch CppUnit trong Visual Studio hoặc CMD

CppUnit có thể được biên dịch thông qua Visual Studio hoặc Command Prompt (CMD) bằng các công cụ như MSBuild. Dưới đây là các bước chi tiết:

**Biên dịch bằng Visual Studio**

Bước 1: Mở CMD

Nhấn Win + R → gõ cmd → Enter.

Hoặc mở Developer Command Prompt for VS (nếu muốn chắc chắn VS nhận toolchain).

Bước 2: Clone vcpkg

Trong CMD gõ:

git clone https://github.com/microsoft/vcpkg.git

cd vcpkg

bootstrap-vcpkg.bat

👉 Sau bước này trong thư mục vcpkg sẽ có file vcpkg.exe.

Bước 3: Cài CppUnit

Vẫn trong CMD, gõ:

vcpkg install cppunit

Nếu bạn dùng Visual Studio 64-bit thì nên cài bản x64 luôn:

vcpkg install cppunit:x64-windows

Bước 4: Tích hợp vào Visual Studio

vcpkg integrate install

Lúc này Visual Studio sẽ tự động tìm thư viện đã cài (bao gồm cppunit) mà không cần chỉnh Include/Library Path thủ công.

Bước 5: Kiểm tra

Mở Visual Studio → tạo Project C++ → thử #include <cppunit/extensions/HelperMacros.h>

## 2.3. Cấu hình Dự án trong Visual Studio

Để sử dụng CppUnit trong một dự án C++, cần cấu hình Visual Studio để nhận diện thư viện và file header:

1. **Tạo dự án mới**: Tạo một dự án rỗng (Empty Project) hoặc ứng dụng giao diện dòng lệnh (Console Application) trong Visual Studio, đặt tên phù hợp, ví dụ: IntegerNumberTest.
2. **Cấu hình đường dẫn include**: Thêm đường dẫn đến thư mục chứa file header của CppUnit (ví dụ: C:\cppunit-1.15.1\include) vào phần cài đặt dự án, đảm bảo Visual Studio có thể tìm thấy các định nghĩa của CppUnit.
3. **Cấu hình liên kết thư viện**: Thêm đường dẫn đến thư mục chứa thư viện CppUnit (ví dụ: C:\cppunit-1.15.1\lib) và chỉ định tên thư viện (như cppunit.lib hoặc cppunitd.lib) trong cài đặt liên kết.
4. **Bật RTTI**: Kích hoạt tính năng Run-Time Type Information trong cài đặt ngôn ngữ C++ để đảm bảo CppUnit hoạt động đúng.
5. **Sao chép file DLL**: Đặt file thư viện động (như cppunit.dll hoặc cppunitd.dll) vào thư mục thực thi của dự án (thư mục Debug hoặc Release) để tránh lỗi không tìm thấy thư viện khi chạy.

# 3. DEMO

## 3.1. Mô tả Lớp Số Nguyên

Nhóm đã thiết kế một lớp số nguyên để minh họa cách sử dụng CppUnit trong việc kiểm thử các phép toán cơ bản. Lớp số nguyên bao gồm:

* Một thuộc tính biểu diễn giá trị số nguyên.
* Các phương thức để truy xuất giá trị số nguyên.
* Các phép toán cơ bản: cộng, trừ, nhân, và chia hai số nguyên.
* Phép so sánh để kiểm tra hai số nguyên có bằng nhau hay không.

Lớp này được chọn vì nó đơn giản, dễ hiểu, và phù hợp để kiểm tra các phép toán số học cơ bản, đồng thời cho phép kiểm tra các trường hợp đặc biệt như chia cho 0.

## 3.2. Thiết kế Test Case cho Các Phép Toán

Nhóm đã thiết kế một tập hợp các bài kiểm thử để xác minh tính đúng đắn của các phép toán cộng, trừ, nhân, và chia của lớp số nguyên. Các bài kiểm thử bao gồm:

* **Kiểm tra phép cộng**: Đảm bảo rằng tổng của hai số nguyên được tính đúng bằng cách so sánh với kết quả mong đợi, ví dụ: cộng 10 và 5 cho kết quả 15.
* **Kiểm tra phép trừ**: Xác minh rằng hiệu của hai số nguyên trả về kết quả chính xác, ví dụ: trừ 10 cho 5 cho kết quả 5.
* **Kiểm tra phép nhân**: Kiểm tra xem tích của hai số nguyên có khớp với công thức nhân hay không, ví dụ: nhân 4 với 3 cho kết quả 12.
* **Kiểm tra phép chia**: Đảm bảo rằng phép chia hai số nguyên xử lý đúng các trường hợp, ví dụ: chia 10 cho 2 cho kết quả 5. Đồng thời kiểm tra trường hợp mẫu số khác 0.
* **Kiểm tra trường hợp đặc biệt**: Kiểm tra các giá trị mặc định, ví dụ: số nguyên khởi tạo không tham số có giá trị bằng 0.
* **Kiểm tra lỗi chia cho 0**: Xác minh rằng phép chia cho số 0 gây ra lỗi và được xử lý đúng bằng cơ chế ngoại lệ.

Mỗi bài kiểm thử được thiết kế để kiểm tra một chức năng cụ thể, sử dụng các phương thức khởi tạo để thiết lập dữ liệu kiểm thử và phương thức dọn dẹp để giải phóng tài nguyên. Các công cụ kiểm tra của CppUnit được sử dụng để so sánh kết quả thực tế với kết quả mong đợi, đồng thời cung cấp thông điệp lỗi chi tiết nếu kiểm thử thất bại.

## 3.3. Chạy Test và Phân tích Kết quả

Các bài kiểm thử được tích hợp vào một chương trình chính để thực thi tất cả các test case trong dự án Visual Studio. Quy trình chạy kiểm thử bao gồm:

1. Thêm các file kiểm thử và chương trình chính vào dự án trong Visual Studio.
2. Biên dịch dự án để kiểm tra lỗi cú pháp hoặc liên kết.
3. Chạy chương trình để thực thi các bài kiểm thử. Kết quả được hiển thị trên cửa sổ Console, cho biết số lượng test case đã pass hoặc fail, ví dụ: "OK (6 tests)" nếu tất cả test thành công.
4. Trong trường hợp test thất bại, thông tin chi tiết như tên bài kiểm thử, vị trí lỗi, và thông điệp lỗi được hiển thị để hỗ trợ nhóm phân tích và sửa lỗi.

Nhóm cũng đã thử nghiệm xuất kết quả kiểm thử dưới dạng tệp XML để hỗ trợ tích hợp với các hệ thống tự động hóa như Jenkins. Kết quả kiểm thử cho thấy tất cả các phép toán cộng, trừ, nhân, chia của lớp số nguyên hoạt động đúng như kỳ vọng trong các trường hợp thông thường. Nhóm cũng phát hiện và xử lý một số trường hợp biên, chẳng hạn như đảm bảo phép chia không gây lỗi khi mẫu số khác 0, và xác minh cơ chế xử lý ngoại lệ khi chia cho 0.

# 4. CÁC VẤN ĐỀ THƯỜNG GẶP VÀ CÁCH KHẮC PHỤC

Trong quá trình thực hiện, nhóm đã gặp một số vấn đề phổ biến khi cài đặt và sử dụng CppUnit trong Visual Studio. Dưới đây là các vấn đề chính và cách khắc phục:

* **Lỗi không tìm thấy file header**: Xảy ra khi Visual Studio không nhận diện được thư mục chứa file header của CppUnit. Nhóm đã khắc phục bằng cách kiểm tra và cập nhật chính xác đường dẫn trong cài đặt dự án, ví dụ: C:\cppunit-1.15.1\include.
* **Lỗi liên kết thư viện**: Visual Studio không tìm thấy file thư viện (.lib) do cấu hình sai đường dẫn hoặc thiếu tên thư viện. Nhóm đã đảm bảo thêm đúng đường dẫn (như C:\cppunit-1.15.1\lib) và tên thư viện (như cppunit.lib hoặc cppunitd.lib) trong cài đặt liên kết.
* **Lỗi thiếu file DLL**: Khi chạy chương trình, hệ thống báo lỗi không tìm thấy file thư viện động (.dll). Nhóm đã sao chép file DLL (như cppunit.dll hoặc cppunitd.dll) vào thư mục thực thi hoặc thêm đường dẫn (như C:\cppunit-1.15.1\bin) vào biến PATH của Windows.
* **Test không thực thi đúng**: Một số test case không chạy do thiếu lệnh đăng ký tập hợp kiểm thử. Nhóm đã kiểm tra và bổ sung các lệnh cần thiết để đảm bảo tất cả test case được thực thi.
* **Lỗi biên dịch qua CMD**: Khi sử dụng MSBuild trong CMD, lỗi có thể xảy ra nếu môi trường không được thiết lập đúng. Nhóm đã đảm bảo mở Developer Command Prompt và kiểm tra các tham số như cấu hình và nền tảng.

# 5. KẾT LUẬN

CppUnit là một công cụ kiểm thử đơn vị hiệu quả, cho phép lập trình viên C++ xác minh tính đúng đắn của các phép toán cộng, trừ, nhân, chia trong lớp số nguyên. Qua báo cáo, nhóm đã thành công trong việc cài đặt CppUnit bằng cả Visual Studio và Command Prompt, cấu hình dự án, và thiết kế các bài kiểm thử cẩn thận, bao quát cả các trường hợp thông thường và biên. Quá trình này giúp đảm bảo chất lượng mã nguồn và phát hiện lỗi sớm trong phát triển phần mềm.

Tuy nhiên, nhóm nhận thấy CppUnit có một số hạn chế, chẳng hạn như việc tích hợp với Visual Studio Test Explorer không được hỗ trợ trực tiếp và quá trình cấu hình phức tạp hơn so với các framework hiện đại như Google Test. Trong tương lai, nhóm đề xuất khám phá các công cụ kiểm thử khác để so sánh hiệu quả, đồng thời nghiên cứu cách tự động hóa quy trình kiểm thử bằng cách tích hợp CppUnit với các hệ thống CI/CD như Jenkins.

# 6. TÀI LIỆU THAM KHẢO

1. CppUnit Project. (2025). *CppUnit - The C++ Unit Testing Framework*. Truy cập từ: https://cppunit.sourceforge.net/
2. LibreOffice Git Repository. (2025). *CppUnit Source Code*. Truy cập từ: https://anongit.freedesktop.org/git/libreoffice/cppunit/
3. Microsoft. (2025). *Visual Studio Documentation: Configuring C++ Projects*. Truy cập từ: https://docs.microsoft.com/en-us/cpp/build/vscpp-step-by-step
4. Martin, R. C. (2008). *Clean Code: A Handbook of Agile Software Craftsmanship*. Prentice Hall.
5. Myers, G. J., Sandler, C., & Badgett, T. (2011). *The Art of Software Testing* (3rd ed.). Wiley.
6. Microsoft. (2025). *MSBuild Command-Line Reference*. Truy cập từ: https://docs.microsoft.com/en-us/visualstudio/msbuild/msbuild-command-line-referen