**Bài 1: Giới thiệu ngôn ngữ C++, cài đặt môi trường lập trình, và viết chương trình đầu tiên**

1. **Giới Thiệu Về C++: Lịch Sử và Ứng Dụng**

### ****1. Lịch Sử của C++****

**C++** là một ngôn ngữ lập trình được phát triển bởi **Bjarne Stroustrup** tại **AT&T Bell Labs** vào đầu những năm 1980. C++ là sự mở rộng của ngôn ngữ lập trình **C**, được thiết kế để thêm vào các tính năng lập trình hướng đối tượng (OOP), trong khi vẫn giữ lại các tính năng của C như hiệu suất cao và khả năng điều khiển tài nguyên hệ thống.

* **1983**: Stroustrup bắt đầu phát triển C++ dưới cái tên "C with Classes" (C với Các Lớp). Mục tiêu là tạo ra một ngôn ngữ có khả năng lập trình hướng đối tượng trong khi vẫn duy trì tính tương thích với C.
* **1985**: Phiên bản chính thức đầu tiên của C++ được phát hành. Nó bao gồm các tính năng cơ bản của lập trình hướng đối tượng như lớp (class), kế thừa (inheritance), và đa hình (polymorphism).
* **1989**: C++ được chuẩn hóa và công nhận chính thức với phiên bản C++ 2.0, bao gồm các tính năng như lớp ảo (virtual classes) và kế thừa đa hình (multiple inheritance).
* **1998**: **ISO/IEC 14882:1998**, thường được gọi là C++98, là phiên bản đầu tiên được tiêu chuẩn hóa của C++.
* **2003**: C++03 được phát hành với các sửa đổi và cập nhật nhỏ.
* **2011**: Phiên bản **C++11** ra mắt với các cải tiến lớn như auto keyword, smart pointers, và lambda expressions.
* **2014**: **C++14** được phát hành, tập trung vào sửa lỗi và cải thiện tính năng.
* **2017**: **C++17** đưa vào nhiều cải tiến như std::optional, std::variant, và std::string\_view.
* **2020**: **C++20** được chuẩn hóa, với các tính năng mới như concepts, ranges, và coroutines.
* **2023**: **C++23** tiếp tục mở rộng các tính năng và cải thiện khả năng của ngôn ngữ.

### ****2. Ứng Dụng của C++****

C++ là một ngôn ngữ lập trình mạnh mẽ và linh hoạt với nhiều ứng dụng trong các lĩnh vực khác nhau. Dưới đây là một số ứng dụng tiêu biểu của C++:

* **Phát Triển Phần Mềm Hệ Thống**:
  + C++ được sử dụng để phát triển các hệ điều hành, trình điều khiển thiết bị, và phần mềm hệ thống nhờ vào khả năng điều khiển tài nguyên hệ thống và hiệu suất cao.
* **Ứng Dụng Phần Mềm**:
  + C++ là sự lựa chọn phổ biến để phát triển các ứng dụng phần mềm lớn, chẳng hạn như các phần mềm máy tính để bàn (desktop applications) và ứng dụng doanh nghiệp.
* **Lập Trình Game**:
  + Nhiều động cơ game nổi tiếng như Unreal Engine được phát triển bằng C++ nhờ vào hiệu suất và khả năng tối ưu hóa.
* **Ứng Dụng Khoa Học và Kỹ Thuật**:
  + C++ được sử dụng trong các ứng dụng tính toán khoa học, mô phỏng, và phân tích dữ liệu nhờ vào khả năng xử lý tính toán phức tạp và hiệu suất cao.
* **Ứng Dụng Tài Chính**:
  + Trong ngành tài chính, C++ thường được dùng để phát triển các hệ thống giao dịch tần suất cao (high-frequency trading systems) và phần mềm phân tích tài chính.
* **Phát Triển Hệ Thống Nhúng**:
  + C++ cũng rất phổ biến trong phát triển hệ thống nhúng (embedded systems), như các thiết bị điện tử, máy móc công nghiệp, và hệ thống IoT.
* **Phát Triển Ứng Dụng Web**:
  + Một số ứng dụng web và dịch vụ backend sử dụng C++ để xử lý các yêu cầu hiệu suất cao và yêu cầu tài nguyên lớn.

### ****3. Tầm Quan Trọng của C++****

* **Hiệu Suất Cao**: C++ cung cấp khả năng kiểm soát tài nguyên hệ thống và hiệu suất tối ưu, rất quan trọng cho các ứng dụng yêu cầu hiệu suất cao.
* **Lập Trình Hướng Đối Tượng**: Các tính năng lập trình hướng đối tượng của C++ giúp tổ chức và quản lý mã nguồn một cách hiệu quả, làm tăng khả năng tái sử dụng và mở rộng.
* **Tương Thích với C**: C++ hỗ trợ tích hợp mã nguồn C, cho phép các nhà phát triển tận dụng các thư viện và mã nguồn C có sẵn.
* **Thư Viện Tiêu Chuẩn**: Thư viện tiêu chuẩn C++ (STL) cung cấp nhiều cấu trúc dữ liệu và thuật toán mạnh mẽ, giúp tăng tốc quá trình phát triển phần mềm.
* **Cộng Đồng Năng Động**: C++ có một cộng đồng phát triển lớn và năng động, với nhiều tài liệu, diễn đàn, và công cụ hỗ trợ.

1. **Sự khác biệt giữa ngôn ngữ lập trình C++ và các ngôn ngữ lập trình khác**

### ****1. C vs. C++****

**C** là ngôn ngữ lập trình thủ tục (procedural programming language) và **C++** mở rộng C bằng cách thêm các tính năng lập trình hướng đối tượng (object-oriented programming). Dưới đây là một số điểm khác biệt chính:

* **C**:
  + **Cú pháp**: Cú pháp cơ bản, tập trung vào lập trình thủ tục. Các chương trình được tổ chức thành các hàm.
  + **Lập trình hướng đối tượng**: Không hỗ trợ lập trình hướng đối tượng.
  + **Hiệu suất**: Cung cấp hiệu suất cao và kiểm soát tài nguyên hệ thống tốt.
  + **Ứng dụng**: Thường dùng cho phát triển hệ điều hành, phần mềm hệ thống, và các ứng dụng cần hiệu suất cao.
* **C++**:
  + **Cú pháp**: Kế thừa cú pháp của C nhưng bổ sung thêm các tính năng như lớp (class), kế thừa (inheritance), và đa hình (polymorphism).
  + **Lập trình hướng đối tượng**: Hỗ trợ lập trình hướng đối tượng, cho phép tổ chức mã nguồn thành các lớp và đối tượng.
  + **Hiệu suất**: Vẫn duy trì hiệu suất cao, đồng thời hỗ trợ các kỹ thuật lập trình tiên tiến.
  + **Ứng dụng**: Được dùng trong phát triển phần mềm hệ thống, game, ứng dụng desktop, và các ứng dụng yêu cầu tính chất lập trình hướng đối tượng.

### ****2. C++ vs. Java****

**Java** là một ngôn ngữ lập trình hướng đối tượng được phát triển bởi Sun Microsystems, hiện thuộc Oracle. Dưới đây là các khác biệt chính giữa C++ và Java:

* **C++**:
  + **Quản lý bộ nhớ**: C++ yêu cầu lập trình viên quản lý bộ nhớ bằng tay thông qua cấp phát và giải phóng bộ nhớ (new, delete).
  + **Hiệu suất**: Thực thi nhanh hơn do không có lớp máy ảo, mã máy được biên dịch trực tiếp.
  + **Thư viện tiêu chuẩn**: STL cung cấp các cấu trúc dữ liệu và thuật toán cơ bản.
  + **Ứng dụng**: Tốt cho phát triển phần mềm hệ thống, ứng dụng hiệu suất cao, game, và phần mềm nhúng.
* **Java**:
  + **Quản lý bộ nhớ**: Java sử dụng Garbage Collection để tự động quản lý bộ nhớ, giảm thiểu lỗi liên quan đến bộ nhớ.
  + **Hiệu suất**: Thực thi chậm hơn so với C++ do mã nguồn được biên dịch thành bytecode và chạy trên máy ảo Java (JVM).
  + **Thư viện tiêu chuẩn**: Java Standard Library cung cấp một bộ thư viện phong phú cho các ứng dụng mạng, giao diện người dùng, và cơ sở dữ liệu.
  + **Ứng dụng**: Phù hợp cho phát triển ứng dụng web, ứng dụng doanh nghiệp, ứng dụng di động (Android), và các hệ thống phân tán.

### ****3. C++ vs. Python****

**Python** là một ngôn ngữ lập trình cao cấp, dễ đọc và dễ viết, thường được sử dụng trong các lĩnh vực khác nhau từ phát triển web đến khoa học dữ liệu.

* **C++**:
  + **Cú pháp**: Cú pháp phức tạp hơn với nhiều chi tiết về quản lý tài nguyên.
  + **Hiệu suất**: Nhanh hơn do mã được biên dịch thành mã máy và khả năng tối ưu hóa.
  + **Ứng dụng**: Tốt cho các ứng dụng yêu cầu hiệu suất cao và kiểm soát chi tiết hệ thống.
* **Python**:
  + **Cú pháp**: Cú pháp đơn giản, dễ đọc và viết. Hỗ trợ lập trình hướng đối tượng, lập trình chức năng, và lập trình thủ tục.
  + **Hiệu suất**: Thực thi chậm hơn do Python là ngôn ngữ thông dịch.
  + **Thư viện tiêu chuẩn**: Có một số lượng lớn các thư viện và gói mở rộng (thông qua PyPI) cho các ứng dụng khác nhau.
  + **Ứng dụng**: Phù hợp cho phát triển web, khoa học dữ liệu, học máy, và tự động hóa.

1. **C++ là một ngôn ngữ biên dịch hay thông dịch?**

### ****1. C++: Ngôn Ngữ Biên Dịch****

* **Quá Trình Biên Dịch**:
  1. **Viết mã nguồn**: Bạn viết mã nguồn C++ trong các tệp với phần mở rộng .cpp (hoặc .h cho các tệp tiêu đề).
  2. **Biên dịch**: Mã nguồn được biên dịch bằng trình biên dịch (compiler) như GCC, Clang, hoặc MSVC. Trình biên dịch chuyển đổi mã nguồn C++ thành mã máy (machine code) hoặc mã đối tượng (object code). Quá trình này có thể bao gồm nhiều bước, như phân tích cú pháp, tối ưu hóa, và liên kết (linking).
  3. **Liên kết**: Mã đối tượng được liên kết lại để tạo ra một chương trình thực thi hoàn chỉnh (executable file). Quá trình liên kết kết hợp mã đối tượng với các thư viện và mô-đun khác để tạo ra tệp thực thi cuối cùng.
  4. **Chạy**: Chương trình thực thi (executable) được chạy trực tiếp trên hệ điều hành mà không cần thông qua một máy ảo hoặc môi trường thực thi.
* **Ưu Điểm**:
  1. **Hiệu Suất Cao**: Do mã máy được tối ưu hóa và biên dịch trước khi thực thi, chương trình C++ thường chạy nhanh hơn.
  2. **Kiểm Soát Tài Nguyên**: C++ cung cấp khả năng kiểm soát chi tiết tài nguyên hệ thống, như bộ nhớ và CPU.
  3. **Tính Tương Thích Ngược**: C++ hỗ trợ nhiều kiểu lập trình, bao gồm lập trình hướng đối tượng, lập trình thủ tục, và lập trình hàm.

### ****2. Ngôn Ngữ Thông Dịch****

* **Quá Trình Thông Dịch**:
  1. **Viết mã nguồn**: Bạn viết mã nguồn trong các tệp có phần mở rộng như .py cho Python hoặc .js cho JavaScript.
  2. **Thông dịch**: Mã nguồn không được biên dịch thành mã máy trước khi chạy. Thay vào đó, mã nguồn được thông dịch (interpreted) bởi một trình thông dịch (interpreter) như CPython (cho Python) hoặc V8 (cho JavaScript). Trình thông dịch thực hiện mã nguồn trực tiếp, dòng bởi dòng.
  3. **Chạy**: Mã nguồn được chạy ngay lập tức trong môi trường thông dịch, như trình duyệt web cho JavaScript hoặc môi trường chạy Python.
* **Ưu Điểm**:
  1. **Dễ Dàng Phát Triển và Gỡ Lỗi**: Mã nguồn có thể được chạy và kiểm tra ngay lập tức mà không cần biên dịch trước, giúp tăng tốc quá trình phát triển và gỡ lỗi.
  2. **Tính Đa Nền Tảng**: Ngôn ngữ thông dịch thường có thể chạy trên nhiều nền tảng mà không cần thay đổi mã nguồn, nhờ vào môi trường thông dịch.

### ****3. Ví Dụ So Sánh****

* **C++**:

Copy code

#include <iostream>

int main() {  
    std::cout << "Hello, World!" << std::endl;  
    return 0;  
}

* + **Biên Dịch**: Mã nguồn được biên dịch thành mã máy và chạy trên hệ điều hành.
  + **Hiệu Suất**: Chạy nhanh và hiệu quả.
* **Python**:

Copy code

print("Hello, World!")

* + **Thông Dịch**: Mã nguồn được thông dịch và thực thi trực tiếp bởi trình thông dịch Python.
  + **Hiệu Suất**: Có thể chậm hơn so với C++ do thông dịch trực tiếp.

1. **Trình Biên Dịch C++ Phổ Biến và tải trình biên dịch**

### ****Trình Biên Dịch C++ Phổ Biến****

1. **GCC (GNU Compiler Collection)**:
   * **GCC** là một trình biên dịch mã nguồn mở và phổ biến cho C++ (và các ngôn ngữ khác như C, Fortran).
   * **Hệ điều hành**: Linux, macOS, và Windows (thông qua MinGW hoặc Cygwin).
   * **Cách sử dụng**:

bash

Copy code

g++ -o myprogram myprogram.cpp

Lệnh trên biên dịch tệp myprogram.cpp và tạo ra tệp thực thi myprogram.

1. **Clang**:
   * **Clang** là một trình biên dịch mã nguồn mở được phát triển bởi nhóm LLVM. Nó cung cấp thông báo lỗi chi tiết và hỗ trợ các tiêu chuẩn C++ mới.
   * **Hệ điều hành**: macOS, Linux, và Windows.
   * **Cách sử dụng**:

Copy code

clang++ -o myprogram myprogram.cpp

Lệnh trên biên dịch tệp myprogram.cpp và tạo ra tệp thực thi myprogram.

1. **Microsoft Visual C++ (MSVC)**:
   * **MSVC** là trình biên dịch của Microsoft được tích hợp trong Visual Studio.
   * **Hệ điều hành**: Windows.
   * **Cách sử dụng**:

Copy code

cl myprogram.cpp

Lệnh trên biên dịch tệp myprogram.cpp và tạo ra tệp thực thi myprogram.exe.

1. **Intel C++ Compiler (ICC)**:
   * **ICC** là trình biên dịch của Intel, tập trung vào tối ưu hóa hiệu suất trên các vi xử lý Intel.
   * **Hệ điều hành**: Windows, Linux, và macOS.
   * **Cách sử dụng**:

Copy code

icc -o myprogram myprogram.cpp

Lệnh trên biên dịch tệp myprogram.cpp và tạo ra tệp thực thi myprogram.

1. **Cài Đặt Môi trường lập trình IDE (Integrated Development Environment)**

### ****Cài Đặt IDE (Integrated Development Environment)****

Mặc dù bạn có thể viết mã C++ bằng bất kỳ trình soạn thảo văn bản nào, sử dụng IDE sẽ giúp quản lý dự án dễ dàng hơn. Dưới đây là một số IDE phổ biến cho C++:

* **Visual Studio**:
  + **Windows**: Tải từ trang [Visual Studio](https://visualstudio.microsoft.com/). Chọn Desktop development with C++ trong phần cài đặt.
* **Code::Blocks**:
  + **Tải từ trang**[**Code::Blocks**](http://www.codeblocks.org/). Có phiên bản tích hợp GCC đi kèm.
* **CLion**:
  + **CLion** là một IDE thương mại của JetBrains hỗ trợ C++. Tải từ trang [JetBrains CLion](https://www.jetbrains.com/clion/).
* **Eclipse CDT**:
  + **Tải Eclipse CDT** từ trang Eclipse. Đây là một plugin cho Eclipse IDE để phát triển C++.
* **Visual Studio Code**:
  + **Windows**: Tải từ trang [Visual Studio Code](https://code.visualstudio.com/). Chọn Desktop development with C++ trong phần cài đặt.

### ****Cài Đặt Các Tiện Ích Mở Rộng cho** Visual Studio Code **(Extensions)****

* **Mở Visual Studio Code** và đi đến phần **Extensions** (hoặc nhấn Ctrl+Shift+X).
* **Cài Đặt Tiện Ích C++**:
  1. **C/C++ Extension** của Microsoft:
     + Tìm và cài đặt tiện ích **C/C++** của Microsoft. Đây là tiện ích chính cung cấp tính năng tự động hoàn thành mã, kiểm tra lỗi, và nhiều tính năng khác cho C++.
  2. **CMake Tools** (Tuỳ Chọn):
     + Nếu bạn sử dụng CMake, cài đặt tiện ích **CMake Tools** để hỗ trợ quản lý dự án CMake.

### ****3. Cài Đặt và Cấu Hình****

* **Cấu Hình IDE**:
  + Sau khi cài đặt IDE, bạn có thể cần cấu hình đường dẫn đến trình biên dịch (như g++) trong các cài đặt của IDE.
  + Thông thường, IDE sẽ tự động phát hiện trình biên dịch nếu bạn đã cài đặt đúng cách.

1. **Viết chương trình C++ đầu tiên sao đó biên dịch mã nguồn**

#include <iostream>

int main() {  
    std::cout << "Hello, World!" << std::endl;  
    return 0;  
}