BỘ NÔNG NGHIỆP VÀ PHÁT TRIỂN NÔNG THÔN

TRƯỜNG CAO ĐẲNG CƠ GIỚI NINH BÌNH

GIÁO TRÌNH

MÔ ĐUN: LẬP TRÌNH C# 1

NGHỀ: LẬP TRÌNH MÁY TÍNH

TRÌNH ĐỘ: CAO ĐẲNG

Ban hành kèm theo Quyết định số: /QĐ-TCĐCGNB ngày….tháng….năm 2017 của Trường Cao đẳng Cơ giới Ninh Bình

Visual C#

Tam Điệp, năm 2017

TUYÊN BỐ BẢN QUYỀN

Tài liệu này thuộc loại sách giáo trình nên các nguồn thông tin có thể được phép dùng nguyên bản hoặc trích dùng cho các mục đích về đào tạo và tham khảo.

Mọi mục đích khác mang tính lệch lạc hoặc sử dụng với mục đích kinh doanh thiếu lành mạnh sẽ bị nghiêm cấm.

LỜI GIỚI THIỆU

Trong hệ thống kiến thức chuyên ngành trang bị cho sinh viên nghề Lập trình máy tính, mô đun góp phần cung cấp những nội dung liên quan đến ngôn ngữ lập trình C#.

Các nội dung chính được trình bày trong tài liệu này gồm các bài:

Bài 1: Microsoft .NET

Bài 2: Ngôn ngữ C#

Bài 3: Cơ bản về C#

Bài 4: Xây dựng lớp – Đối tượng

Bài 5: Kế thừa – Đa hình

Bài 6: Nạp chồng toán tử

Bài 7: Mảng, chỉ mục và tập hợp

Bài 8: Xử lý chuỗi

Mặc dầu có rất nhiều cố gắng, nhưng không tránh khỏi những khiếm khuyết, rất mong nhận được sự đóng góp ý kiến của độc giả để giáo trình được hoàn thiện hơn.

Ninh Bình, ngày tháng năm 2017

THAM GIA BIÊN SOẠN

1. Chủ biên – CN Phạm Thị Thoa

2. Th.S Nguyễn Xuân Khôi

3. Th.S Nguyễn Anh Văn

MỤC LỤC

TRANG

GIÁO TRÌNH MÔ ĐUN

Tên mô đun: Lập trình C# 1

Mã số mô đun: MĐ20

Vị trí, tính chất, ý nghĩa và vai trò của mô đun:

- Vị trí: Đây là mô đun chứa đựng kiến thức nền tảng về ngôn ngữ lập trình C#, là mô đun hỗ trợ cho hai mô đun lập trình Windows 2 và lập trình Windows 3.

- Tính chất. Mô đun này yêu cầu đã học qua các kiến thức về lập trình hướng đối tượng, cấu trúc dữ liệu.

- Ý nghĩa và vai trò của đun:

Mục tiêu của mô đun:

- Kiến thức:

+ Trình bày được các kiến thức về nền tảng Microsoft .NET;

+ Trình bày được các kiến thức và kỹ năng về lập trình hướng đối tượng trên C#;

+ Trình bày được cú pháp các lệnh cơ bản trong C#;

+ Trình bày được các bước xây dựng lớp đối tượng;

+ Mô tả được các bước nạp chồng hàm, chồng toán tử;

+ Trình bày các kiến thức cơ bản về mảng, chuỗi.

- Kỹ năng:

+ Sử dụng thành thạo các toán tử;

+ Sử dụng được các chỉ dẫn biên dịch;

+ Sử dụng thành thạo các cấu trúc điều khiển;

+ Xây dựng được lớp, xây dựng lớp có chồng toán tử;

+ Sử dụng được các lớp cơ sở của NET.

+ Khai báo và thao tác được với mảng và chuỗi;

- Năng lực tự chủ và trách nhiệm: Rèn luyện tính cẩn thận, chính xác, đảm bảo an toàn cho người và trang thiết bị.

Nội dung của môn học/mô đun:

Bài 1. Microsoft .NET

Mã bài: MĐ20_B01

Giới thiệu : Trong bài này chúng ta sẽ nghiên cứu khái quát về Microsoft.NET, biên dịch và ngôn ngữ C#.

Mục tiêu :

- Trình bày được nền tảng Microsoft .NET, trình biên dịch và MSIL, lý do và lịch sử về sự ra đời của ngôn ngữ C#;
- Mô tả được khái quát về ngôn ngữ C#;
- Đảm bảo an toàn cho người và thiết bị.

1. Microsoft .NET

Microsoft .NET gồm 2 phần chính : Framework và Integrated Development Environment (IDE).

Framework cung cấp những gì cần thiết và căn bản, chữ Framework có nghĩa là khung hay khung cảnh trong đó ta dùng những hạ tầng cơ sở theo một qui ước nhất định để công việc được trôi chảy.

IDE thì cung cấp một môi trường giúp chúng ta triển khai dễ dàng, và nhanh chóng các ứng dụng dựa trên nền tảng .NET. Nếu không có IDE chúng ta cũng có thể dùng một trình soạn thảo ví như Notepad hay bất cứ trình soạn thảo văn bản nào và sử dụng command line để biên dịch và thực thi, tuy nhiên việc này mất nhiều thời gian. Tốt nhất là chúng ta dùng IDE phát triển các ứng dụng, và cũng là cách dễ sử dụng nhất.

Tính năng của Micosoft.NET

- Một mô hình lập trình cho phép nhà phát triển xây dựng các ứng dụng dịch vụ web và ứng dụng client với Extensible Markup Language (XML).
- Tập hợp dịch vụ XML Web, như Microsoft .NET My Services cho phép nhà phát triển đơn giản và tích hợp người dùng kinh nghiệm.
- Cung cấp các server phục vụ bao gồm: Windows 2000, SQL Server, và BizTalk Server, tất cả điều tích hợp, hoạt động, và quản lý các dịch vụ XML Web và các ứng dụng.
- Các phần mềm client như Windows XP và Windows CE giúp người phát triển phân phối sâu và thuyết phục người dùng kinh nghiệm thông qua các dòng thiết bị.

➢ Nhiều công cụ hỗ trợ như Visual Studio .NET, để phát triển các dịch vụ Web XML, ứng dụng trên nền Windows hay nền web một cách dể dàng và hiệu quả.

Kiến trúc .NET Framework

.NET Framework là một platform mới làm đơn giản việc phát triển ứng dụng trong môi trường phân tán của Internet. .NET Framework được thiết kế đầy đủ để đáp ứng theo quan điểm sau:

- Để cung cấp một môi trường lập trình hướng đối tượng vững chắc, trong đó mã nguồn đối tượng được lưu trữ và thực thi một cách cục bộ. Thực thi cục bộ nhưng được phân tán trên Internet, hoặc thực thi từ xa.
- Để cung cấp một môi trường thực thi mã nguồn mà tối thiểu được việc đóng gói phần mềm và sự tranh chấp về phiên bản.
- Để cung cấp một môi trường thực thi mã nguồn mà đảm bảo việc thực thi an toàn mã nguồn, bao gồm cả việc mã nguồn được tạo bởi hãng thứ ba hay bất cứ hãng nào mà tuân thủ theo kiến trúc .NET.
- Để cung cấp một môi trường thực thi mã nguồn mà loại bỏ được những lỗi thực hiện các script hay môi trường thông dịch.
- Để làm cho những người phát triển có kinh nghiệm vững chắc có thể nắm vững nhiều kiểu ứng dụng khác nhau. Như là từ những ứng dụng trên nền Windows đến những ứng dụng dựa trên web.
- Để xây dựng tất cả các thông tin dựa triên tiêu chuẩn công nghiệp để đảm bảo rằng mã nguồn trên .NET có thể tích hợp với bất cứ mã nguồn khác.

.NET Framework có hai thành phần chính: Common Language Runtime (CLR) và thư viện lớp

Common Language Runtime (CLR): thực hiện quản lý bộ nhớ, quản lý thực thi tiểu trình, thực thi mã nguồn, xác nhận mã nguồn an toàn, biên bịch và các dịch vụ hệ thống khác. Những đặc tính trên là nền tảng cơ bản cho những mã nguồn được quản lý chạy trên CLR.

Thư viện lớp .NET Framework là một tập hợp những kiểu dữ liệu được dùng lại và được kết hợp chặt chẽ với Common Language Runtime.

Thư viện lớp là hướng đối tượng cung cấp những kiểu dữ liệu mà mã nguồn được quản lý của chúng ta có thể dẫn xuất

2. Biên dịch và MSIL

Trong .NET Framework, chương trình không được biên dịch vào các tập tin thực thi mà thay vào đó chúng được biên dịch vào những tập tin trung gian gọi là Microsoft Intermediate Language (MSIL). Những tập tin MSIL được tạo ra từ C# cũng tương tự như các tập tin MSIL được tạo ra từ những ngôn ngữ khác của .NET, platform ở đây không cần biết ngôn ngữ của mã nguồn. Điều quan trọng chính yếu của CLR là chung (common), cùng một runtime hỗ trợ phát triển trong C# cũng như trong VB.NET.

Mã nguồn C# được biên dịch vào MSIL khi chúng ta build project. Mã MSIL này được lưu vào trong một tập tin trên đĩa. Khi chúng ta chạy chương trình, thì MSIL được biên dịch một lần nữa, sử dụng trình biên dịch Just-In-Time (JIT). Kết quả là mã máy được thực thi bởi bộ xử lý của máy.

Trình biên dịch JIT tiêu chuẩn thì thực hiện theo yêu cầu. Khi một phương thức được gọi, trình biên dịch JIT phân tích MSIL và tạo ra sản phẩm mã máy có hiệu quả cao, mã này có thể chạy rất nhanh. Trình biên dịch JIT đủ thông minh để nhận ra khi một mã đã được biên dịch, do vậy khi ứng dụng chạy thì việc biên dịch chỉ xảy ra khi cần thiết, tức là chỉ biên dịch mã MSIL chưa biên dịch ra mã máy. Khi đó một ứng dụng .NET thực hiện, chúng có xu hướng là chạy nhanh và nhanh hơn nữa, cũng như là những mã nguồn được biên dịch rồi thì được dùng lại.

Do tất cả các ngôn ngữ .NET Framework cùng tạo ra sản phẩm MSIL giống nhau, nên kết quả là một đối tượng được tạo ra từ ngôn ngữ này có thể được truy cập hay được dẫn xuất từ một đối tượng của ngôn ngữ khác trong .NET

3. Ngôn ngữ C#

Ngôn ngữ C# khá đơn giản, chỉ khoảng 80 từ khóa và hơn mười mấy kiểu dữ liệu được xây dựng sẵn. Tuy nhiên, ngôn ngữ C# có ý nghĩa cao khi nó thực thi những khái niệm lập trình hiện đại. C# bao gồm tất cả những hỗ trợ cho cấu trúc, thành phần component, lập trình hướng đối tượng.

C# được xây dựng trên nền tảng của hai ngôn ngữ mạnh nhất là C++ và Java.

Ngôn ngữ C# được phát triển bởi đội ngũ kỹ sư của Microsoft, trong đó người dẫn đầu là Anders Hejlsberg và Scott Wiltamuth.

Ngôn ngữ C# chứa những từ khóa cho việc khai báo những kiểu lớp đối tượng mới và những phương thức hay thuộc tính của lớp, và cho việc thực thi đóng gói, kế thừa, và đa hình, ba thuộc tính cơ bản của bất cứ ngôn ngữ lập trình hướng đối tượng.

Trong ngôn ngữ C# mọi thứ liên quan đến khai báo lớp điều được tìm thấy trong phần khai báo của nó. Định nghĩa một lớp trong ngôn ngữ C# không đòi hỏi phải chia ra tập tin header và tập tin nguồn giống như trong ngôn ngữ C++. Hơn thế nữa, ngôn ngữ C# hỗ trợ kiểu XML, cho phép chèn các tag XML để phát sinh tự động các document cho lớp.

C# cũng hỗ trợ giao diện interface, nó được xem như một cam kết với một lớp cho những dịch vụ mà giao diện quy định. Trong ngôn ngữ C#, một lớp chỉ có thể kế thừa từ duy nhất một lớp cha, tức là không cho đa kế thừa như trong ngôn ngữ C++, tuy nhiên một lớp có thể thực thi nhiều giao diện. Khi một lớp thực thi một giao diện thì nó sẽ hứa là nó sẽ cung cấp chức năng thực thi giao diện.

Trong ngôn ngữ C#, những cấu trúc cũng được hỗ trợ, nhưng khái niệm về ngữ nghĩa của nó thay đổi khác với C++. Trong C#, một cấu trúc được giới hạn, là kiểu dữ liệu nhỏ gọn, và khi tạo thể hiện thì nó yêu cầu ít hơn về hệ điều hành và bộ nhớ so với một lớp. Một cấu trúc thì không thể kế thừa từ một lớp hay được kế thừa nhưng một cấu trúc có thể thực thi một giao diện.

Ngôn ngữ C# cung cấp những đặc tính hướng thành phần (component-oriented), như là những thuộc tính, những sự kiện. Lập trình hướng thành phần được hỗ trợ bởi CLR cho phép lưu trữ metadata với mã nguồn cho một lớp. Metadata mô tả cho một lớp, bao gồm những phương thức và những thuộc tính của nó, cũng như những sự bảo mật cần thiết và những thuộc tính khác.

BÀI 2. NGÔN NGỮ C#

Mã bài: MĐ20_B02

Giới thiệu: Trong bài này chúng ta sẽ nghiên cứu khái quát về ngôn ngữ C#: tại sao phải dùng C#, các bước để chuẩn bị cho chương trình và giới thiệu cho các bạn biết sơ qua 1 chương trinh C# đơn giản.

Mục tiêu:

- Mô tả được các đặc điểm nổi bật của ngôn ngữ C#;
- Trình bày được các bước để viết một chương trình trong C#;
- Viết được chương trình C# đơn giản;
- Đảm bảo an toàn cho người và thiết bị.

1. Tại sao phải sử dụng ngôn ngữ C#

Một số mục đích khi xây dựng ngôn ngữ **C#:**

- C# là ngôn ngữ đơn giản
- C# là ngôn ngữ hiện đại
- C# là ngôn ngữ hướng đối tượng
- C# là ngôn ngữ mạnh mẽ và mềm dẻo
- C# là ngôn ngữ có ít từ khóa
- C# là ngôn ngữ hướng module
- C# sẽ trở nên phổ biến

a. C# là ngôn ngữ đơn giản

- C# loại bỏ một vài sự phức tạp và rối rắm của những ngôn ngữ như Java và c++, bao gồm việc loại bỏ những macro, những template, đa kế thừa, và lớp cơ sở ảo (virtual base class).

- Ngôn ngữ C# đơn giản vì nó dựa trên nền tảng C và C++. Nếu chúng ta thân thiện với C và C++ hoặc thậm chí là Java, chúng ta sẽ thấy C# khá giống về diện mạo, cú pháp, biểu thức, toán tử và những chức năng khác

được lấy trực tiếp từ ngôn ngữ C và C++, nhưng nó đã được cải tiến để làm cho ngôn ngữ đơn giản hơn

b. C# là ngôn ngữ hiện đại

Những đặc tính của C#: xử lý ngoại lệ, thu gom bộ nhớ tự động, những kiểu dữ liệu mở rộng, và bảo mật mã nguồn. Đây những đặc tính được mong đợi trong một ngôn ngữ hiện đại.

c. C# là ngôn ngữ hướng đối tượng

Những đặc điểm chính của ngôn ngữ hướng đối tượng (Object-oriented language) là sự đóng gói (encapsulation), sự kế thừa (inheritance), và đa hình (polymorphism). C# hỗ trợ tất cả những đặc tính trên.

d. C# là ngôn ngữ mạnh mẽ và cũng mềm dẻo

Với ngôn ngữ C# chúng ta chỉ bị giới hạn ở chính bởi bản thân hay là trí tưởng tượng của chúng ta. Ngôn ngữ này không đặt những ràng buộc lên những việc có thể làm. C# được sử dụng cho nhiều các dự án khác nhau như là tạo ra ứng dụng xử lý văn bản, ứng dụng đồ họa, bản tính, hay thậm chí những trình biên dịch cho các ngôn ngữ khác.

e. C# là ngôn ngữ ít từ khóa

C# là ngôn ngữ sử dụng giới hạn những từ khóa.

Bảng sau liệt kê các từ khóa của ngôn ngữ C#.

abstr	defaul	foreac	Objec	sizeof	unsaf
As	delega	goto	Opera	stacka	ushor
Base	do	if	Out	static	using
Bool	doubl	implic	Overri	string	virtu
Brea	else	in	Param	struct	volat
Byte	enum	int	Privat	switch	void
Case	event	interfa	Protec	this	while
Catc	explic	intern	Public	throw	
Char	extern	is	Readо	true	
chec	false	lock	Ref	try	
Clas	finally	long	Retur	typeof	

Cons	fixed	names	Sbyte	uint	
conti	float	new	Seale	ulong	
deci	for	null	Short	unche	

f. C# là ngôn ngữ hướng module

Mã nguồn C# có thể được viết trong những phần được gọi là những lớp, những lớp này chứa các phương thức thành viên của nó. Những lớp và những phương thức có thể được sử dụng lại trong ứng dụng hay các chương trình khác.

g. C# là một ngôn ngữ phổ biến

C# là ngôn ngữ đứng thứ 5 trong biểu đồ khảo sát 15 ngôn ngữ sử dụng phổ biến nhất năm 2014.

2. Ngôn ngữ C# với ngôn ngữ khác

Microsoft nói rằng C# mang đến sức mạnh của ngôn ngữ C++ với sự dễ dàng của ngôn ngữ Visual Basic. Có thể nó không dễ như Visual Basic, nhưng với phiên bản Visual Basic.NET (Version 7) thì ngang nhau. Bởi vì chúng được viết lại từ một nền tảng. Chúng ta có thể viết nhiều chương trình với ít mã nguồn hơn nếu dùng C#.

Một điều quan trọng khác với C++ là mã nguồn C# không đòi hỏi phải có tập tin header.

Tất cả mã nguồn được viết trong khai báo một lớp.

Như đã nói ở bên trên. .NET runtime trong C# thực hiện việc thu gom bộ nhớ tự động. Do điều này nên việc sử dụng con trỏ trong C# ít quan trọng hơn trong C++. Những con trỏ cũng có thể được sử dụng trong C#, khi đó những đoạn mã nguồn này sẽ được đánh dấu là không an toàn (unsafe code).

C# cũng từ bỏ ý tưởng đa kế thừa như trong C++. Và sự khác nhau khác là C# đưa thêm thuộc tính vào trong một lớp giống như trong Visual Basic. Và những thành viên của lớp được gọi duy nhất bằng toán tử "." khác với C++ có nhiều cách gọi trong các tình huống khác nhau.

Điểm giống nhau C# và Java là cả hai cùng biên dịch ra mã trung gian: C# biên dịch ra MSIL còn Java biên dịch ra bytecode. Sau đó chúng được

thực hiện bằng cách thông dịch hoặc biên dịch just-in-time trong từng máy ảo tương ứng.

Sự khác nhau giữ Java và C#: trong ngôn ngữ C# nhiều hỗ trợ được đưa ra để biên dịch mã ngôn ngữ trung gian sang mã máy. C# chứa nhiều kiểu dữ liệu cơ bản hơn Java và cũng cho phép nhiều sự mở rộng với kiểu dữ liệu giá trị.

3. Các bước chuẩn bị cho chương trình

- Xác định mục tiêu của chương trình.
- Xác định những phương pháp giải quyết vấn đề.
- Tạo một chương trình để giải quyết vấn đề.
- Thực thi chương trình để xem kết quả.

4. Chương trình C# đơn giản

* Lớp, đối tượng và kiểu dữ liệu

- Điều cốt lõi của lập trình hướng đối tượng là tạo ra các kiểu mới. Kiểu là một thứ được xem như trừu tượng.

- Kiểu được định nghĩa như một dạng vừa có thuộc tính chung (properties) và các hành vi ứng xử (behavior) của nó.

- Kiểu trong kiểu trong C# được định nghĩa là một lớp (class), và các thể hiện riêng của từng lớp được gọi là đối tượng (object).

- Khai báo lớp trong C# không có dấu ; sau ngoặc } cuối cùng của lớp. Và khácvới lớp trong C/C++ là chia thành 2 phần header và phần định nghĩa. Trong C# , định nghĩa một lớp được gói gọn trong dấu { } sau tên lớp và trong cùng một tập tin.

* Phương thức

- Phương thức chính là các hàm được định nghĩa trong lớp.

- Các phương thức này chỉ ra rằng các hành động mà lớp có thể làm được cùng với cách thức làm hành động đó.

- Khi chương trình thực thi, CLR gọi hàm Main() đầu tiên, hàm Main() là đầu vào của chương trình, và mỗi chương trình phải có một hàm Main().

Cấu trúc hàm main:

```
static void Main( )
```

{

}

* Chú thích

Một chương trình được viết tốt thì cần phải có chú thích các đoạn mã được viết. Các đoạn chú thích này sẽ không được biên dịch và cũng không tham gia vào chương trình. Mục đích chính là làm cho đoạn mã nguồn rõ ràng và dễ hiểu.

Một chuỗi chú thích trên một dòng thì bắt đầu bằng ký tự "//"

Ví dụ: // Xuat ra man hinh.

Ngoài ra C# còn cho phép kiểu chú thích cho một hay nhiều dòng, và ta phải khai báo "/*" ở phần đầu chú thích và kết thúc chú thích là ký tự "*/".

* Ứng dụng Console

Ứng dụng này giao tiếp với người dùng thông quan bàn phím và không có giao diện người dùng (UI)

* Lớp Console

- Đưa dữ liệu ra màn hình chúng ta dùng phương thức Write hoặc WriteLine của lớp Console

+ Cú pháp:

```
Console.Write(dòng diều khiển[, danh sách đối]);
Console.WriteLine(dòng diều khiển[, danh sách đối]);
```

Trong đó:

dòng diều khiển: là một hằng xâu ký tự đặt trong hai dấu "....", nó bao gồm các loại đối tượng sau: Các ký tự thông thường, các ký tự đặc biệt và các đặc tả có dạng: {i,-j:dt } có thể đặt bất kỳ ở vị trí nào trong xâu với: i là thứ tự của một đối mà ta cần đưa giá trị của chúng ra tại vị trí đặt đặc tả(các đối có thứ tự bắt đầu từ không), j là độ rộng dành cho đối cần đưa ra(nếu j mà lớn hơn độ dài của dữ liệu cần đưa ra thì giá trị đó *được căn phải, nếu muốn căn trái thì ta thêm dấu trừ phía trước, còn nếu j mà nhỏ hơn độ dài thực tế của dữ liệu cần đưa ra thì không có gì thay đổi), d là định dạng dữ liệu đưa ra, ví dụ định dạng là C định* dạng theo kiểu tiền tệ, N định dạng kiểu số, G định dạng chuẩn....., t là số chữ số thập phân(chỉ áp dụng cho số thực)

danh sách các đối: các đối có thể là hằng, biến, biểu thức... và đặt cách nhau một dấu phẩy

+ Sự khác nhau giữa Write(...) và WriteLine(...) là:

Write(...) sau khi viết dữ liệu ra màn hình thì con trỏ đặt ở cuối dòng còn

WriteLine(...) sau khi viết dữ liệu ra màn hình thì con trỏ đặt ở đầu dòng tiếp theo.

Để nhập dữ liệu vào từ bàn phím chúng ta dùng phương thức ReadLine có trong lớp Console như sau:

```
Kiểudữliệu Tênbiến;
Tênbiến=Kiểudữliệu.Parse(Console.ReadLine());
```

Thông thường trong qua trình nhập dữ liệu chúng ta thương kết hợp phương thức Write với ReadLine trong lớp Console. Phương thức này trả về dữ liệu kiểu string.

Ví dụ:

```
double a;
Console.Write("Nhap a=");
a = double.Parse(Console.ReadLine());
```

* ***Namespace –Không gian tên***

Namespace cung cấp cho ta cách tổ chức quan hệ giữa các lớp và các kiểu khác nhau. Đây là kĩ thuật cho phép .NET tránh việc các tên lớp, tên biến, tên hàm... đụng độ vì trùng tên với nhau giữa các lớp. Để khai báo không gian tên chúng ta sử dụng từ khóa namespace.

Đối tượng Console bị hạn chế bởi namespace bằng việc sử dụng mã lệnh:

```
System.Console.WriteLine();
```

* Toán tử ".": được sử dụng để truy cập đến phương thức hay dữ liệu trong một lớp

Ví dụ: System.Console.WriteLine("Chao Mung");

* ***Từ khóa using***

Để làm cho chương trình gọn hơn, và không cần phải viết từng namespace cho từng đối tượng, C# cung cấp từ khóa là using, sau từ khóa

này là một namespace hay subnamespace với mô tả đầy đủ trong cấu trúc phân cấp của nó.

Ta có thể dùng dòng lệnh :

```
using System;
```

ở đầu chương trình và khi đó trong chương trình nếu chúng ta có dùng đối tượng Console thì không cần phải viết đầy đủ : System.Console. mà chỉ cần viết Console. thôi.

Ví dụ:

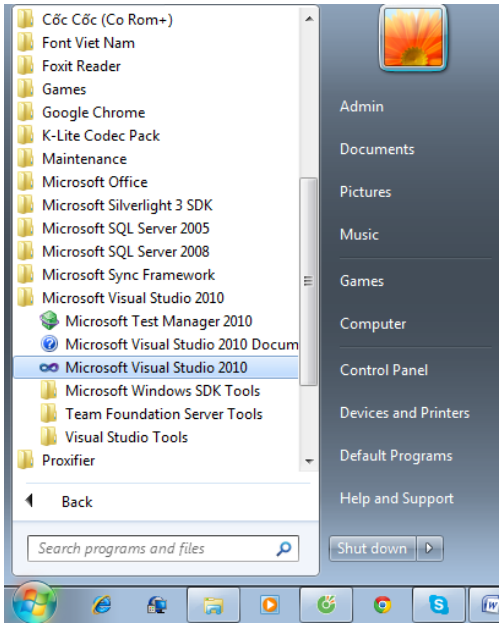
```
using System; class ChaoMung
{
    static void Main()
    {
        //Xuat ra man hinh chuoi thong bao
Console.WriteLine("Chao Mung");
    }
}
```

* Trong C# phân biệt chữ hoa chữ thường

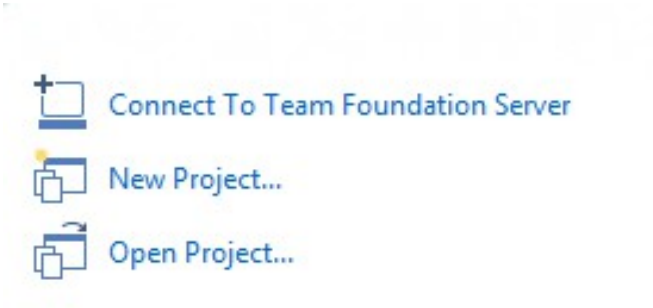
Trình tự thực hiện

- Khởi động chương trình Visual Studio 2010

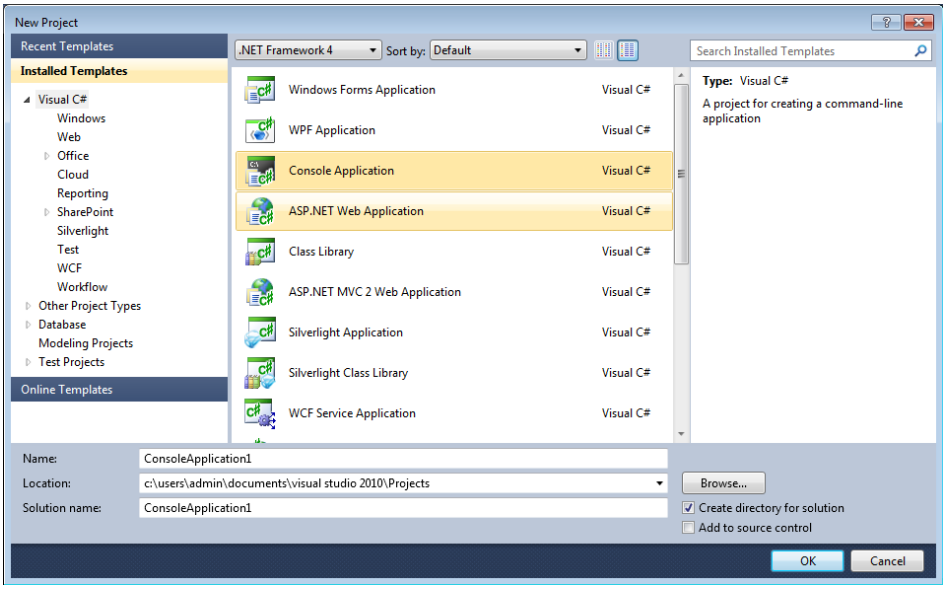
Start\ All Programs\ Microsoft Visual Studio 2010\ Microsoft Visual Studio 2010



Chọn New Project



Chọn Console Application\ Gõ tên project vào mục Name\OK



- Gõ lệnh vào hàm Main

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace vi_du1
{
    class Program
    {
        static void Main(string[] args)
        {

        }
    }
}
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace vi_du1
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Chao mung ban den voi C#");
          //Lệnh dừng màn hình để xem
            Console.ReadKey();
        }
    }
}
```

Biên dịch: Nhấn F6

Chạy chương trình: Nhấn F5