

TỰ HỌC LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG & LẬP TRÌNH CƠ SỞ DỮ LIỆU



MỘT CÁCH NHANH CHÓNG VÀ CÓ HIỆU QUẢ NHẤT

Qua các chương trình mẫu :

- LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG
- LẬP TRÌNH DÙNG CƠ SỞ DỮ LIỆU ACCESS 2003
- LẬP TRÌNH DÙNG CƠ SỞ DỮ LIỆU SQL SERVER 2000
- PHÂN PHỐI ỨNG DỤNG QUA SETUP WIZARD
- BÁO CÁO QUA CRYSTAL REPORT

● (Dùng cho kỹ thuật viên & sinh viên ngành tin học) ●

CUNG CÁCH TRÌNH BÀY :

- *Ngắn gọn và dễ hiểu*
- *Hướng dẫn chi tiết và rõ ràng*
- *Các chương trình mẫu thực tế và đa dạng*



Đậu Quang Tuấn

Cử nhân toán – Kỹ sư tin học

Tự học lập trình hướng đối tượng và lập trình cơ sở dữ liệu

C#

MỘT CÁCH NHANH CHÓNG VÀ CÓ HIỆU QUẢ NHẤT

Qua các chương trình mẫu

- ☒ Lập trình hướng đối tượng
- ☒ Lập trình dùng cơ sở dữ liệu Access 2003
- ☒ Lập trình dùng cơ sở dữ liệu SQL Server 2000
- ☒ Phân phối ứng dụng qua Setup Wizard
- ☒ Báo cáo qua Crystal Report

(Dùng cho kỹ thuật viên và sinh viên ngành tin học)

Cung cách trình bày:

- Ngắn gọn và dễ hiểu
- Hướng dẫn chi tiết và rõ ràng
- Các chương trình mẫu thực tế và đa dạng

NHÀ XUẤT BẢN GIAO THÔNG VẬN TẢI 2006

Lời mở đầu

Xin chào các bạn học sinh, sinh viên và quý độc giả yêu thích bộ môn tin học. Lời đầu tiên, chúng tôi xin chân thành cảm ơn các bạn vì trong thời gian qua đã đón đọc những tập sách về ứng dụng và lập trình của chúng tôi biên soạn. Những ý kiến đóng góp và nhận xét của các bạn sinh viên và quý độc giả xa gần, đã khích lệ vô cùng to lớn và động viên chúng tôi tiếp tục nghiên cứu những khía cạnh mới trong bộ môn tin học này. Sau các cuốn "Tự học lập trình Visual Basic.NET", "Tự học lập trình cơ sở dữ liệu Access 2003 và Visual Basic.NET", "Tự học lập trình cơ sở dữ liệu SQL Server 2000 và Visual Basic.NET", "Tự học lập trình Visual C++.NET" đã được xuất bản. Hôm nay chúng tôi hoàn thành tập sách "Tự học lập trình hướng đối tượng và lập trình cơ sở dữ liệu Visual C#" qua các chương trình mẫu sao cho thật ngắn gọn và dễ hiểu. Nội dung của tập sách sẽ hướng dẫn các bạn lập trình hướng đối tượng căn bản qua các bài tập mẫu, lập trình dùng CSDL Access 2003, lập trình dùng CSDL SQL Server 2000 theo hai cách, tự thiết kế hoặc dùng Data Form Wizard. Chúng tôi có hướng dẫn thêm cách in các bảng bằng Crystal Report và cách thực hiện để phân phối ứng dụng khi đã hoàn thành xong chương trình. Nghiên cứu tập sách này, chắc chắn các bạn sẽ cảm thấy hài lòng, những kiến thức hướng dẫn trong các chương trình mẫu sẽ giúp các bạn tự thiết kế những chương trình khác một cách dễ dàng và các bạn có thể nghiên cứu thêm để viết tiểu luận hoặc đồ án ra trường. Hy vọng tập sách này sẽ là động cơ thúc đẩy các bạn yêu thích lập trình và sẽ đưa các bạn tiến sâu hơn trên con đường tin học của mình.

Mặc dầu đã có nhiều cố gắng khi biên soạn, nhưng cũng không tránh khỏi thiếu sót, chúng tôi mong được sự đóng góp ý kiến của quý độc giả xa gần, để lần tái bản sau sẽ hoàn chỉnh hơn. Chúc các bạn học giỏi và đóng góp trí tuệ mình cho đất nước thân yêu của chúng ta.

Đây là một cuốn sách không thể thiếu được trong tủ sách tin học của bạn. Nếu cảm nhận dễ hiểu xin giới thiệu với quý độc giả khác để cùng đón đọc.

Tác giả
Đậu Quang Tuấn

Phần I

cách vào cửa sổ

MS Development Environment

Tìm hiểu cửa sổ thiết kế

MS Development Environment

Đầu tiên các bạn cài **Visual Studio .NET** vào máy của mình, việc cài đặt bình thường như những ngôn ngữ khác.

Trong phần lý thuyết ngắn gọn này, chúng tôi sẽ giải thích về cửa sổ, nơi bạn sẽ thiết kế một chương trình, đó là **Microsoft Development Environment**.

Ghi chú:

Khi chúng tôi nói click chuột, các bạn hiểu là click nút trái của chuột.

☞ Microsoft Development Environment

Đây là một môi trường triển khai tổng thể, cho phép làm việc ngay trên mọi khía cạnh của một dự án. Bạn tiến hành các bước để thấy được giao diện của Microsoft Development Environment.

Bạn *khởi động Visual C# .NET* một trong những cách sau:

① Cách 1:

⇒ Vào **Start**.

⇒ Đưa vật sáng đến **Programs**.

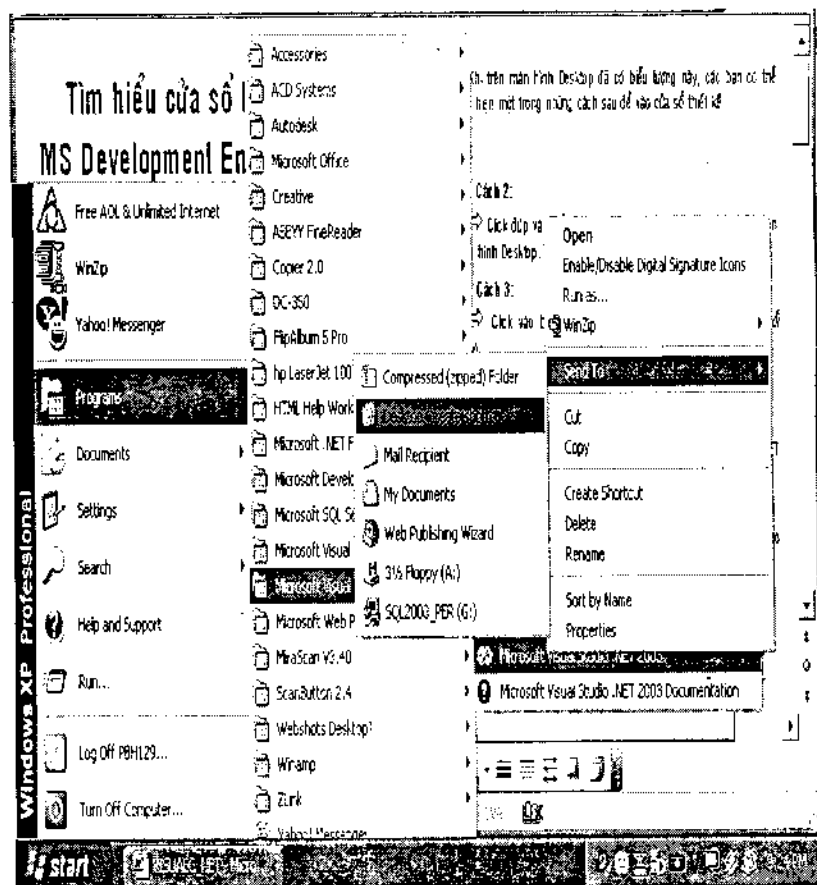
⇒ Đưa vật sáng đến **Microsoft Visual Studio .NET 2003**.

⇒ Click chọn **Microsoft Visual Studio .NET 2003**.

Nếu biểu tượng Microsoft Visual Studio .NET 2003 chưa có trên màn hình Desktop, các bạn có thể tạo như sau:

⇒ Vào **Start**.

- ⇒ Đưa vệt sáng đến **Programs**.
 - ⇒ Đưa vệt sáng đến **Microsoft Visual Studio .NET 2003**.
 - ⇒ Click *nút phải chuột* tại **Microsoft Visual Studio .NET 2003**.
 - ⇒ Di chuyển vệt sáng đến **Send To** trong menu hiện ra.
 - ⇒ Click chọn **Desktop (create shortcut)**. Trên Desktop sẽ có biểu tượng của Microsoft Visual Studio .NET 2003.
- Khi trên màn hình Desktop đã có biểu tượng này, các bạn có thể thực hiện một trong những cách sau để vào cửa sổ thiết kế.



H1 Tạo biểu tượng Microsoft Visual Studio .NET trên Desktop



H2 Biểu tượng Microsoft Visual Studio nằm trong hình tròn

② Cách 2:

⇒ Click đúp vào biểu tượng **Microsoft Visual Studio .NET 2003** trên màn hình **Desktop**.

③ Cách 3:

⇒ Click vào biểu tượng **Microsoft Visual Studio .NET 2003** (**Đổi màu**)

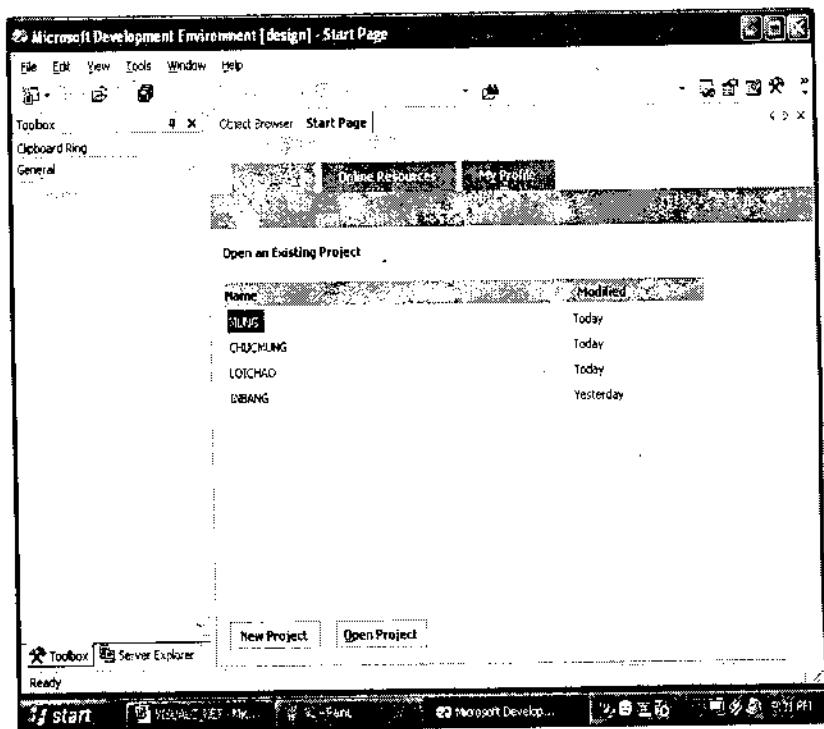
⇒ Nhấn phím **Enter**.

④ Cách 4:

⇒ Click phải chuột vào biểu tượng **Microsoft Visual Studio .NET 2003**

⇒ Click vào nút **Open** trong menu hiện ra

Khi thực hiện một trong những cách trên, các bạn sẽ thấy giao diện của Microsoft Development Environment như hình sau:



H3 Cửa sổ Microsoft Development Environment

Trên cửa sổ này các bạn có thể thiết kế các chương trình trên Visua Basic .NET, Visual C++ .NET, Visual C# .NET, ...

Ghi chú:

Trong Visual Studio .NET cửa sổ để vào các ngôn ngữ .NET chung, từ cửa sổ này, chúng ta có thể chọn ngôn ngữ mà mình muốn.

Phần 2

Dùng Visual C# Projects và Console Application

BÀI 1

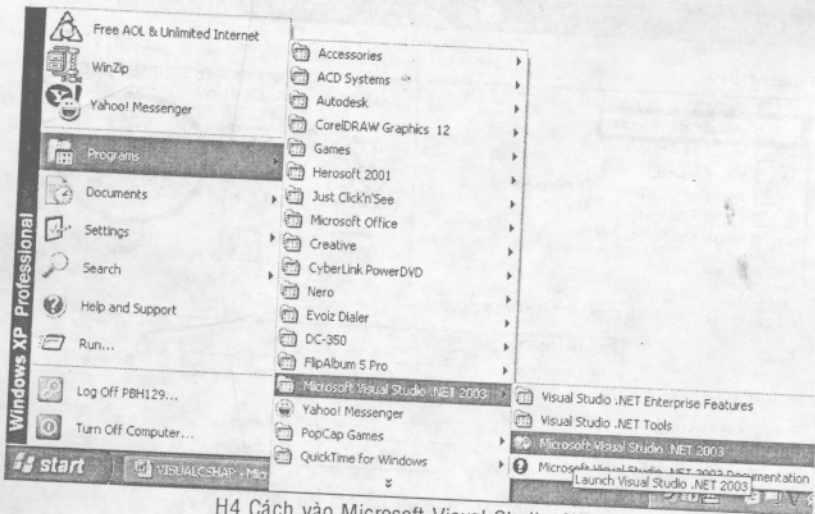
CHO XUẤT HIỆN MỘT CÂU CHÀO

Bài đầu tiên là một chương trình đơn giản, sẽ cho xuất hiện trên màn hình một câu "CHAO MUNG BAN DEN VOI VISUAL C# .NET"

Các chương trình dưới đây được tạo trong Visual C# dùng **Console Application**.

Bạn sẽ tiến hành như sau:

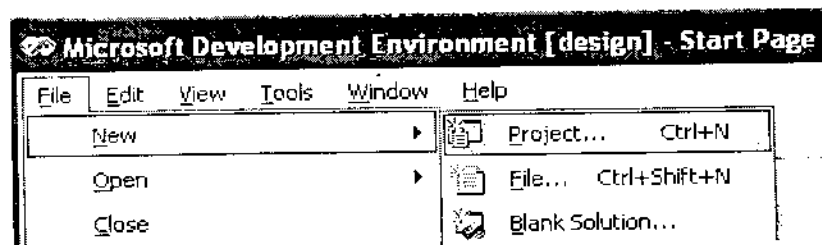
* Khởi động **Microsoft Visual Studio.NET**, bằng cách click nút **Start**, di chuyển vật sáng đến **Programs**, di chuyển vật sáng đến **Microsoft Visual Studio.NET 2003**, chọn **Microsoft Visual Studio.NET 2003** như hình sau:



H4 Cách vào Microsoft Visual Studio .NET

Khi chọn xong, bạn sẽ thấy cửa sổ **Microsoft Development Environment** như hình H3 ở trên

* Click chuột vào menu **File**, chọn **New**, chọn **Project** như hình sau:



H5 Tạo Project mới

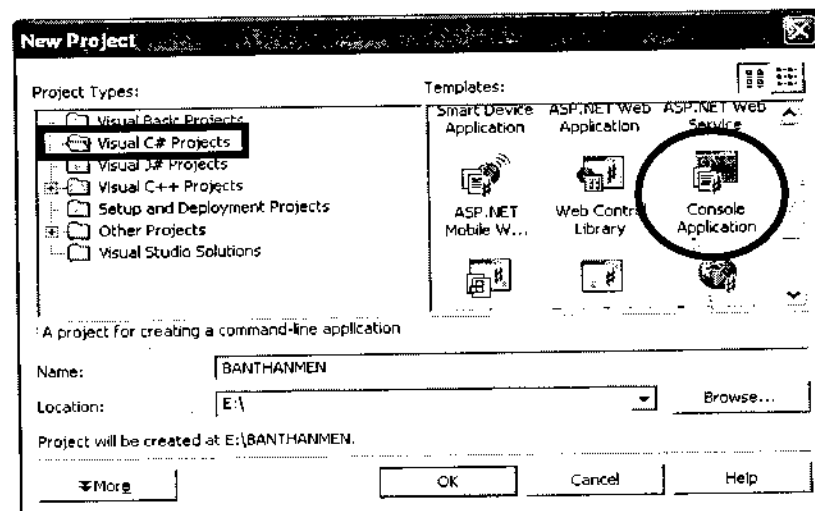
* Trong cửa sổ **New Project** hiện ra, bạn chọn:

* **Visual C# Project** trong khung bên trái có tên **Project Types**.

* Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.

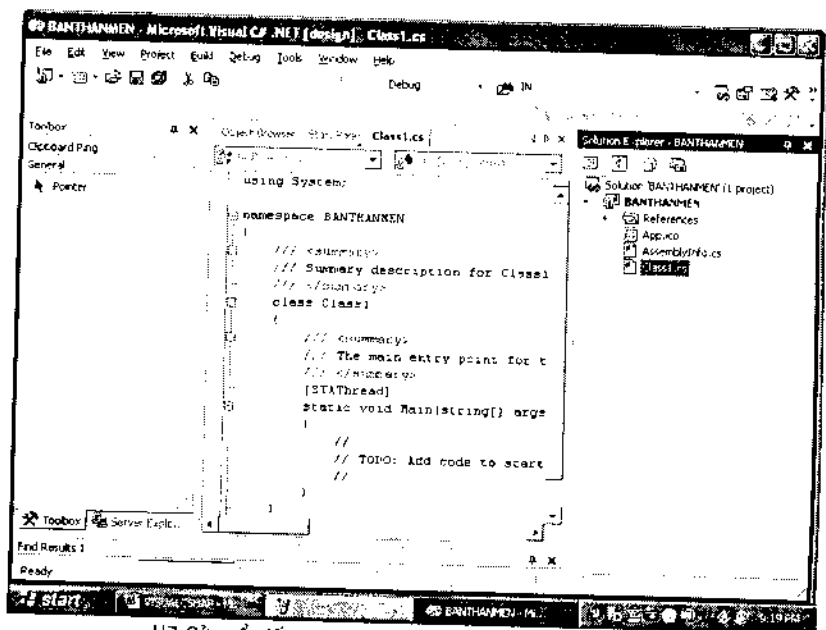
* Đặt tên trong khung **Name**, giả sử là tên **BANTHANMEN**

* Chọn đường dẫn chứa **Project**, giả sử chúng tôi chọn **E:** trong khung **Location**, bạn sẽ thấy **Project** chứa trong **E:\BANTHANMEN** như hình sau:



H6 Chọn cách tạo và nơi lưu trữ Project

* Click nút **OK**. Bạn sẽ có cửa sổ hiện ra như hình sau:



H7 Cửa sổ viết code hiện ra với những thông tin có sẵn

Bạn thấy phía bên trái là cửa sổ *viết code*, bên phải là cửa sổ **Solution Explorer** chứa thông tin của **Project**

Đoạn code có sẵn như sau:

```
using System;
namespace BANTHANMEN
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    class Class1
    {
        /// <summary>
        /// The main entry point for the
        /// application.
        /// </summary>
```

```

[STAThread]
static void Main(string[] args)
{
    //
    // TODO: Add code to start
    // application here
    //
}
}

```

* Những dòng chữ ghi sau // là những dòng ghi chú, trình biên dịch sẽ bỏ qua.

* Nếu bắt đầu bằng /* và kết thúc là */. Kiểu này cho bạn kéo dài lời ghi chú lên nhiều hàng.

* Loại ghi chú thứ ba là bắt đầu bằng 3 dấu gạch chéo ///
 Bạn sẽ viết đoạn code như sau vào dưới dòng

// TODO: Add code to start application here

```

Console.WriteLine("CHAO MUNG BAN DEN VOI VISUAL C# .NET");
Console.Read();

```

Nhập xong, bạn sẽ có nguyên đoạn code như sau:

```

using System;
namespace BANTHANMEN
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]

```

```

static void Main(string[] args)
{
    //
    // TODO: Add code to start application here
    // Hai lenh sau duoc them vao
    Console.WriteLine("CHAO MUNG BAN DEN VOI
        VISUAL C# .NET");
    Console.Read();
}
}
}

```

Lưu ý:

Trong Visual C# .NET tên của lớp hoặc tên của các phương thức cũng phân biệt chữ hoa và chữ thường như đặc thù của ngôn ngữ C hoặc C++. Nếu trong câu lệnh trên, bạn gõ write, read (w và r là kí tự thường thì trình biên dịch sẽ báo sai)

Giải thích:

☐ Việc nhập xuất dữ liệu thông qua thiết bị nhập xuất chuẩn (console) được thực hiện chủ yếu bởi các phương thức của lớp

Console

☐ Phương thức **Write()** để xuất ra dữ liệu (có thể xuất ra chuỗi ký tự, số nguyên, số thực ...) ở trên là xuất ra chuỗi "CHAO MUNG BAN DEN VOI VISUAL C# .NET", chuỗi được bao bởi cặp nháy kép.

☐ Nếu không có **Console.Read()** hoặc **Console.ReadLine()** cuối chương trình, khi thực hiện chương trình, bạn thấy màn hình nháy lên và trở về cửa sổ viết code, không thấy được kết quả trên màn hình. Tương tự như trong C++ ta phải dùng hàm getch(), hoặc trong Pascal dùng Readln.

Để biên dịch và chạy chương trình, bạn thực hiện như sau:

- * Vào menu **Build**
- * Click **Build BANTHANMEN**
- * Nhấn phím **F5** để chạy thử chương trình

Nếu thực hiện đúng tất cả những điều trên, bạn sẽ có kết quả như sau:



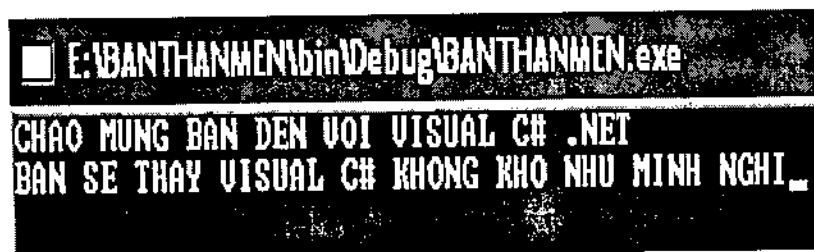
H8 Kết quả khi chạy chương trình

Bạn thử xóa bỏ câu lệnh `Console.Read()`; hoặc đánh kí hiệu ghi chú phía trước dòng này, nhấn phím F5 để chạy chương trình, các bạn sẽ không thấy màn hình thể hiện kết quả, nó về cửa sổ viết code.

Cũng chương trình như trên, giả sử bạn muốn dòng một xuất hiện lời chào như trên, dòng hai xuất hiện câu "BAN SE THAY VISUAL C# KHONG KHO NHU MINH NGHI". Bạn thực hiện như thế nào? Chỉ đơn giản vào cửa sổ viết code, thêm Line vào sau Write của câu trên và viết thêm câu lệnh như sau:

```
Console.WriteLine("CHAO MUNG BAN DEN VOI VISUAL C# .NET");  
Console.Write("BAN SE THAY VISUAL C# KHONG KHO NHU MINH  
NGHI");  
Console.Read();
```

☐ Biên dịch và nhấn phím F5 để chạy thử chương trình, bạn sẽ có kết quả như hình sau:



H9 Kết quả nếu viết lại code như trên

☐ Nếu viết như sau (thêm Line vào Write của câu lệnh thứ hai) bạn cũng có kết quả như trên

```
Console.WriteLine("CHAO MUNG BAN DEN VOI VISUAL C# .NET");
```

```
Console.WriteLine("BAN SE THAY VISUAL C# KHONG KHO NHU MINH  
NGHI");
```

```
Console.Read();
```

☒ Nếu bỏ Line sau Write của câu lệnh thứ nhất, chúng ta sẽ có kết quả như hình sau:



H10 Nó sẽ thể hiện trên một dòng

Vì không có chữ "Line" sau câu lệnh thứ nhất, chuỗi của lệnh Write thứ hai sẽ in tiếp ngay sau chuỗi của lệnh WriteLine thứ nhất. Bạn thấy câu thứ nhất và thứ hai liền nhau, khi thực hiện như vậy, bạn phải để thêm một khoảng trắng sau chuỗi thứ nhất, hoặc trước chuỗi thứ hai.

Vậy phương thức WriteLine() sẽ xuống dòng mới sau khi in ra màn hình, cũng giống như trong ngôn ngữ Pascal (Write hoặc WriteLn), trong ngôn ngữ C và C++ muốn xuống dòng thì dùng "\n"

Các bạn có thể viết chương trình để xuất ra trên màn hình một bài thơ mà bạn thích.

Giả sử chúng ta viết chương trình để xuất hiện trên màn hình bài thơ của Trịnh Công Sơn tặng cho Hoàng Anh như sau:

Đường xa vạn dặm em ngổi!

Nỗi đời xa vắng nỗi đời dần đo

Em là nhật nguyệt từ đây

Tuổi mười chín ấy cũng phai phai người

Em ơi hồng sẽ phai hồng

Đóa hoa hàm tiếu phiêu bồng nỗi đau

Hoa vàng một đóa lạ lùng

Gió chiều tịch mịch sẽ trùng trùng xa

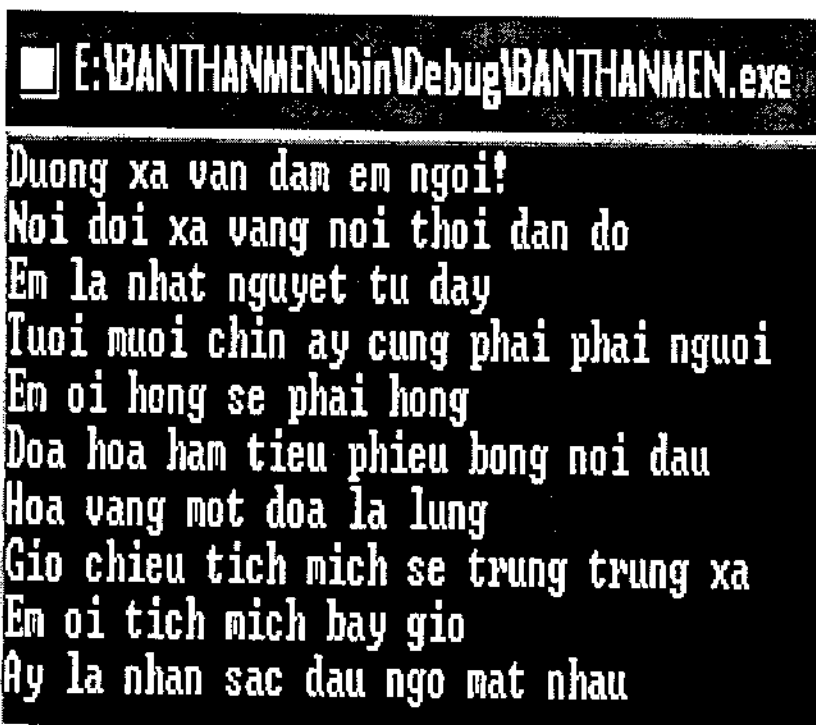
Em ơi tịch mịch bây giờ

Ấy là nhan sắc đâu ngờ mất nhau

Bạn mở chương trình trên và sửa lại code như sau:

```
Console.WriteLine("Duong xa van dam em ngoi!");  
Console.WriteLine("Noi doi xa vang noi thoi dan do");  
Console.WriteLine("Em la nhat nguyet tu day");  
Console.WriteLine("Tuoi muoi chin ay cung phai phai nguoi");  
Console.WriteLine("Em oi hong se phai hong");  
Console.WriteLine("Doa hoa ham tieu phieu bong noi dau");  
Console.WriteLine("Hoa vang mot doa la lung");  
Console.WriteLine("Gio chieu tích mich se trung trung xa");  
Console.WriteLine("Em oi tích mich bay gio");  
Console.WriteLine("Ay la nhan sac dau ngo mat nhau");  
Console.Read();
```

▣ Biên dịch lại chương trình, nhấn phím F5 chúng ta sẽ có kết quả như hình sau:



```
E:\BANTHANMEN\bin\Debug\BANTHANMEN.exe  
Duong xa van dam em ngoi!  
Noi doi xa vang noi thoi dan do  
Em la nhat nguyet tu day  
Tuoi muoi chin ay cung phai phai nguoi  
Em oi hong se phai hong  
Doa hoa ham tieu phieu bong noi dau  
Hoa vang mot doa la lung  
Gio chieu tích mich se trung trung xa  
Em oi tích mich bay gio  
Ay la nhan sac dau ngo mat nhau
```

H11 Bài thơ vừa nhập

BÀI 2

CẤU TRÚC IF

Trước khi làm ví dụ, chúng tôi giới thiệu đến các bạn Kiểu dữ liệu cơ sở và các phép toán trong C#.

Các kiểu số nguyên với những phép toán: Cộng (dấu +), trừ (dấu -), nhân (dấu *), chia (dấu /) tính thương số và tính số dư (dấu %).

Các kiểu dữ liệu trong C#

Kiểu	Ý nghĩa
Byte	Kiểu số nguyên không dấu 1 byte (8 bit) Phạm vi từ 0 đến 255
Sbyte	Kiểu số nguyên có dấu 1 byte (8 bit) Phạm vi từ -128 đến 127
Short	Số nguyên có dấu, độ dài 16 bit Phạm vi từ -32768 đến 32767
Ushort	Số nguyên không dấu, độ dài 16 bit Phạm vi từ 0 đến 65535
Int	Số nguyên có dấu, độ dài 32 bit Phạm vi từ -2^{31} đến $2^{31}-1$
UInt	Số nguyên không dấu 32 bit Phạm vi từ 0 đến $2^{32}-1$
Long	Số nguyên có dấu 64 bit Phạm vi từ -2^{63} đến $2^{63}-1$
Ulong	Số nguyên không dấu 64 bit Phạm vi từ 0 đến $2^{64}-1$
Char	Lưu các kí tự unicode từ U+0000 đến U+FFFF
Float	Số thực 32 bit, chính xác đến 7 chữ số, số có trị tuyệt đối lớn nhất là $3,4 \times 10^{32}$. Số dương nhỏ nhất là $1,5 \cdot 10^{45}$.
Double	Số thực 64 bit, chính xác đến 15-16 chữ số, số có trị tuyệt đối lớn nhất là $1,7 \times 10^{308}$. Số dương nhỏ nhất là

	$4,9.10^{324}$.
Bool	Nhận giá trị true (đúng) hoặc false (sai).
String	Kiểu chuỗi ký tự, có thể dùng phép cộng chuỗi (+) để ghép hai chuỗi thành một. Các phép toán so sánh bằng (==) và khác (!=) để so sánh giá trị của các biến kiểu chuỗi.

Giống như C++, C# cũng có các phép toán như sau:

Các phép toán về số học

- * là phép toán nhân
- / là phép toán chia
- + là phép toán cộng
- là phép toán trừ
- % là phép toán tìm số dư

Toán tử gán

- Trong C# toán tử gán là "="

Ví dụ: $x=2;$
 $y=5;$

Các phép toán về luận lý

- && phép toán And
Ví dụ: $(x>10)\&\&(y<1)$
- || phép toán Or
Ví dụ: $(x>=5)\|\!(y<=4)$
- ! Phép toán phủ định

Các phép toán so sánh

- Trong C# các toán tử so sánh như sau:

- < nhỏ hơn
- > lớn hơn
- == bằng
- <= nhỏ hơn hoặc bằng
- >= lớn hơn hoặc bằng
- != khác

Ví dụ: $x<y$, $m>n$, $a==b$, $j<=l$, $z>=0$, $u!=v$

Tác tử tăng giảm

Biến ++

Tăng Biến lên một đơn vị

Biến --

Giảm Biến một đơn vị

Ví dụ:

Nếu $x = 5$. $x ++$ sẽ làm cho $x = 6$.

Nếu $y = 8$. $y --$ sẽ làm cho $y = 7$

Tác tử tăng giảm

Nếu tăng a lên 2000, bạn có thể làm như sau:

$a = a + 2000$;

Phép tính cộng xảy ra trước khi gán. Bạn có thể thực hiện

$a = a - 2000$;

$a = a * 2000$;

$a = a / 2000$;

$a = a \% 2000$;

Bạn có thể viết lại các dòng lệnh trên ngắn gọn hơn như sau:

$a += 2000$; // a tăng thêm 2000

$a -= 2000$; // trừ a bớt 2000

$a *= 2000$; // nhân a cho 2000

$a /= 2000$; // chia a cho 2000

$a %= 2000$; // a mod cho 2000

Tác tử tiền tố và hậu tố

Bạn xem ví dụ sau để hiểu

Giả sử $B = 5$

Lệnh 1: $A = B++$; // dấu ++ nằm liền sau biến gọi là hậu tố

Lệnh 2: $A = ++B$; // dấu ++ nằm liền trước biến gọi là tiền tố

Lệnh 1 **gán trước rồi tăng sau**, nghĩa là $A = 10$, $B = 11$

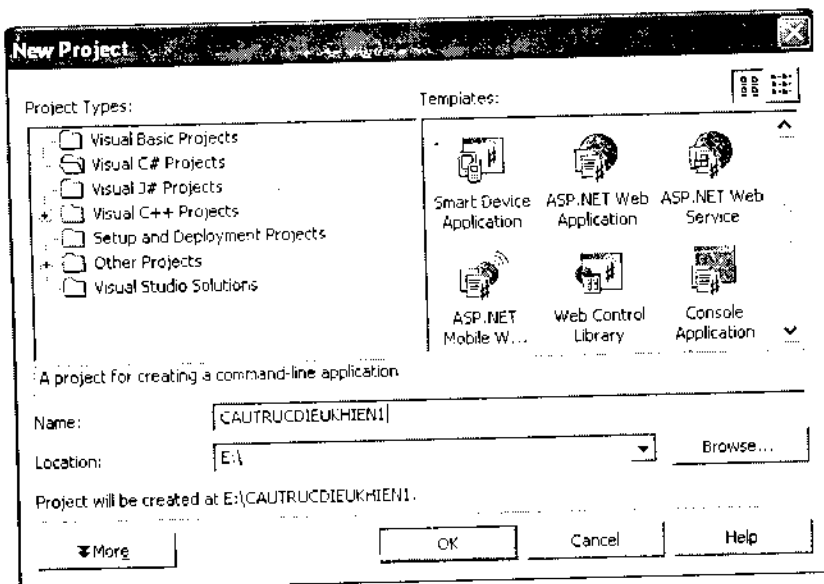
Lệnh 2 **tăng trước gán sau**, nghĩa là $A = 11$, $B = 11$

Chúng ta trở về với phát biểu if. Giả sử chúng ta viết chương trình nhập vào ba số nguyên a , b và c , nhập kết quả của tổng ba số đó. Nếu nhập đúng, trên màn hình sẽ báo ra câu "Hoan hô bé đã tính đúng.". Nếu nhập sai, chương trình sẽ báo ra câu "Rất tiếc bé đã tính sai!!!".

Chúng ta sẽ thực hiện như sau:

◆ Vào **Microsoft Visual Studio .NET**

◆ Tạo một **Project** mới có tên **CAUTRUCDIEUKHIEN1** bằng cách vào menu **File**, chọn **New**, chọn **Project**. Bạn thấy như hình sau:



H12

◆ Click nút **OK** trên hộp thoại.

Vào cửa sổ viết **code**, viết các câu lệnh ngay phía dưới câu ghi chú
`// TODO: Add code to start application here như sau:`

int a, b, c;

long kq;

```
Console.WriteLine("Ban nhap vao so a = ");
```

```
a=int.Parse(Console.ReadLine());
```

```
Console.WriteLine("Ban nhap vao so b = ");
```

```
b=int.Parse(Console.ReadLine());
```

```
Console.WriteLine("Ban nhap vao so c = ");
```

```
c=int.Parse(Console.ReadLine());
```

```
Console.WriteLine("Tong cua ba so tren = ");
```

```
kq=long.Parse(Console.ReadLine());
```

```
if (kq==a+b+c)
```

```
    Console.WriteLine("Hoan ho be da tinh dung.");
```

```
if (kq!=a+b+c)
```

```
Console.WriteLine("Rat tiếc be đã tính sai!!!");  
Console.Read();
```

Bạn có thể thấy một phần của cửa sổ viết code như hình dưới, còn các câu lệnh phía trên và các dấu móc bên dưới của chương trình, chúng có sẵn, đoạn code trên thêm vào vị trí như đã nêu ở trên.

```
CAUTRUCDIENKHIEN1.Class1 Main(string[] args)  
1 static void Main(string[] args)  
2 {  
3     //  
4     // TODO: Add code to start application here  
5     //  
6     int a, b, c;  
7     long kq;  
8     Console.Write("Ban nhap vao so a = ");  
9     a=int.Parse(Console.ReadLine());  
10    Console.Write("Ban nhap vao so b = ");  
11    b=int.Parse(Console.ReadLine());  
12    Console.Write("Ban nhap vao so c = ");  
13    c=int.Parse(Console.ReadLine());  
14    Console.Write("Tong cua ba so tren = ");  
15    kq=long.Parse(Console.ReadLine());  
16    if (kq==a+b+c)  
17        Console.WriteLine("Hoan ho be đã tính đúng.");  
18    if (kq!=a+b+c)  
19        Console.WriteLine("Rat tiếc be đã tính sai!!!");  
20    Console.Read();  
H13
```

Giải thích:

- Khai báo biến a, b và c có kiểu nguyên (int).
- Biến kq có kiểu long.
- Hiện ra câu "Ban nhap vao so a = "

Bạn đã biết là việc nhập xuất dữ liệu thông qua thiết bị nhập xuất chuẩn (console) được thực hiện chủ yếu bởi các phương thức của lớp **Console: Read()/ReadLine()** và **Console: Write()/WriteLine()**. Phương

thức `ReadLine()` được sử dụng để nhập vào một chuỗi ký tự, phương thức `Read()` chỉ cho phép nhập vào một ký tự.

Để đổi thành các dạng dữ liệu khác ta có thể dùng phương thức `Parse()` của kiểu dữ liệu mong muốn.

- Với câu lệnh `a=int.Parse(Console.ReadLine());` sẽ đổi chuỗi (kí tự số nhập vào) thành số nguyên kiểu `int`.
- Tương tự như trên với các câu lệnh nhập số `b`, số `c` và kết quả tổng của ba số bên dưới.
- Phát biểu `if` thứ nhất với điều kiện `(kq==a+b+c)`. Nếu đúng thì sẽ thực hiện câu lệnh theo sau `if`, tiếp tục xét câu lệnh tiếp theo của chương trình là phát biểu `if` kế tiếp, với điều kiện `(kq!=a+b+c)`. Nếu thỏa sẽ thực hiện câu lệnh theo sau phát biểu `if` này.

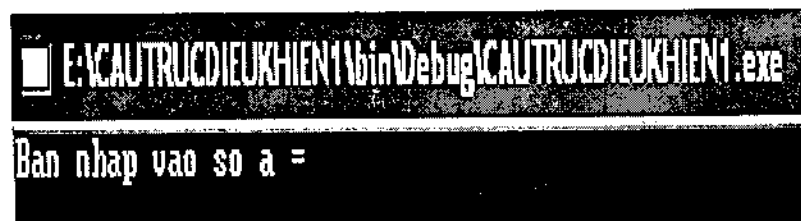
Biên dịch và chạy thử chương trình:

Biên dịch bằng cách:

◆ Vào menu **Build**, chọn **Build CAUTRUCDIEUKHIEN1**

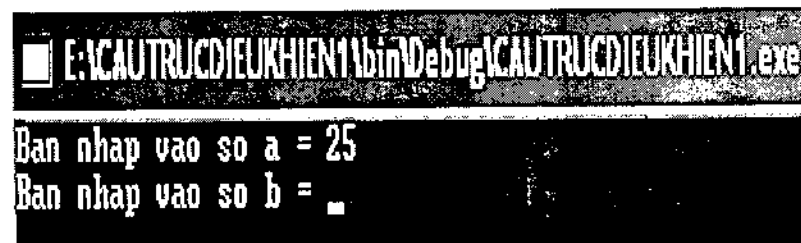
◆ Nhấn phím **F5** để chạy thử chương trình, bạn sẽ thấy như hình

sau:



H14

◆ Bạn nhập số `a`, giả sử chúng ta nhập `25` sau đó gõ phím **Enter**, bạn sẽ thấy hình như sau:



H15

◆ Giả sử chúng ta nhập số b bằng 525, gõ phím Enter, hình sau hiện ra:

```
E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 25
Ban nhap vao so b = 525
Ban nhap vao so c = _
```

H16

◆ Giả sử chúng ta nhập số c bằng 125, gõ phím Enter, chúng ta có hình như sau:

```
E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 25
Ban nhap vao so b = 525
Ban nhap vao so c = 125
Tong cua ba so tren =
```

H17

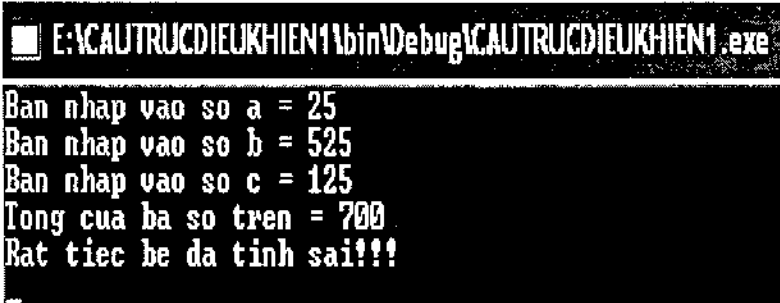
◆ Bạn nhập tổng của ba số trên, giả sử chúng ta nhập 675, gõ phím Enter, chúng ta thấy kết quả như hình sau:

```
E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 25
Ban nhap vao so b = 525
Ban nhap vao so c = 125
Tong cua ba so tren = 675
Hoan ho be da tinh dung.
```

H18

◆ Vì $a + b + c = 675$, nên kết quả báo ra "Hoan ho be đa tinh dung"

◆ Nhấn phím **F5** chạy lại chương trình, nhập lại a, b, c như trên, nhưng nhập kết quả tổng của ba số trên là một số khác 675, giả sử chúng ta nhập kết quả là 700. Bạn sẽ thấy kết quả như hình sau:



```
E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 25
Ban nhap vao so b = 525
Ban nhap vao so c = 125
Tong cua ba so tren = 700
Rat tien be đa tinh sai!!!
```

H19

Bài trên bạn đã dùng phát biểu if với cấu trúc như sau:

if (điều kiện)

S;

Phát biểu trên sẽ xét điều kiện theo sau if. Nếu đúng thì thực hiện lệnh S. Nếu sai thì coi như phát biểu này đã thực hiện xong.

Ghi chú:

■ Nếu sau phát biểu if có một câu lệnh, bạn không dùng cặp dấu móc. Nếu có từ hai câu lệnh trở lên, bạn phải dùng cặp dấu móc bao quanh các câu lệnh này.

Ví dụ:

Cũng bài tập trên, nếu bạn sửa lại code ở phần cuối như sau:

if (kq==a+b+c)

Console.WriteLine("Hoan ho be đa tinh dung.");

if (kq!=a+b+c)

Console.WriteLine("Rat tien be đa tinh sai!!!");

Console.WriteLine("Be can co gang nhiều hơn nua");

Console.Read();

Bạn thêm câu lệnh tô đậm ở trên.

Nhấn phím **F5** chạy thử chương trình, giả sử nhập $a=1$, $b=1$, $c=1$. Nhập kết quả là 3. Đúng. Thay vì trên màn hình chỉ xuất hiện câu

"Hoan ho be da tinh dung", nếu viết như trên nó sẽ hiện thêm câu "Be can co gang nhieu hon nua". Bạn sẽ thấy như hình sau:

```
E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 1
Ban nhap vao so b = 1
Ban nhap vao so c = 1
Tong cua ba so tren = 3
Hoan ho be da tinh dung.
Be can co gang hon nua
```

H20

Kết quả như vậy là sai, thực hiện lệch ý của chúng ta. Để không hiện câu "Be can co gang nhieu hon nua", Bạn phải thêm cặp dấu móc { } bao quanh hai câu lệnh của phát biểu if thứ hai như sau:

```
if (kq==a+b+c)
    Console.WriteLine("Hoan ho be da tinh dung.");
if (kq!=a+b+c)
{
    Console.WriteLine("Rat tiếc be da tinh sai!!!");
    Console.WriteLine("Be can co gang nhieu hon nua");
}
Console.Read();
```

Chạy thử chương trình, nhập kết quả như trên, bạn sẽ thấy như sau:

```
E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 1
Ban nhap vao so b = 1
Ban nhap vao so c = 1
Tong cua ba so tren = 3
Hoan ho be da tinh dung.
```

H21

Bạn cần chú ý thực hiện đúng như trên để tránh sai sót khi lập trình.

BÀI 3

CẤU TRÚC IF CÓ ELSE

Cấu trúc if else sẽ có tác dụng chuyển hướng tùy theo biểu thức điều kiện nhận giá trị đúng (True) hay sai (False).

Ví dụ:

```
if (kq==a+b+c)
    Console.WriteLine("Hoan ho be da tinh dung.");
if (kq!=a+b+c)
{
    Console.WriteLine("Rat tiec be da tinh sai!!!");
    Console.WriteLine("Be can co gang nhieu hon nua");
}
```

Chúng ta có thể sửa lại như sau:

```
if (kq==a+b+c)
    Console.WriteLine("Hoan ho be da tinh dung.");
else
{
    Console.WriteLine("Rat tiec be da tinh sai!!!");
    Console.WriteLine("Be can co gang nhieu hon nua");
}
```

Bạn thử vào cửa sổ viết code, sửa lại phát biểu có else như trên, lúc này đoạn code sẽ như sau:

```
int a, b, c;
long kq;
Console.Write("Ban nhap vao so a = ");
a=int.Parse(Console.ReadLine());
Console.Write("Ban nhap vao so b = ");
b=int.Parse(Console.ReadLine());
Console.Write("Ban nhap vao so c = ");
c=int.Parse(Console.ReadLine());
Console.Write("Tong cua ba so tren = ");
```

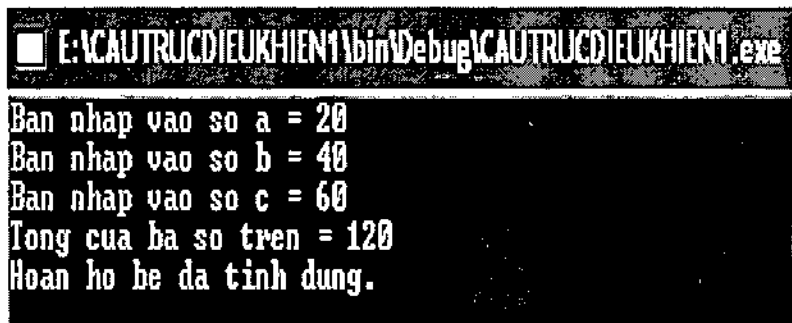
```

kq=long.Parse(Console.ReadLine());
if (kq==a+b+c)
    Console.WriteLine("Hoan ho be da tinh dung.");
else
{
    Console.WriteLine("Rat tiec be da tinh sai!!!");
    Console.WriteLine("Be can co gang hon nua");
}
Console.Read();

```

☐ Biên dịch và nhấn phím F5 để chạy thử chương trình, bạn sẽ có kết quả giống như dùng hai phát biểu if ở trên.

Bạn xem thử kết quả qua hai hình sau với các số mà chúng tôi nhập.

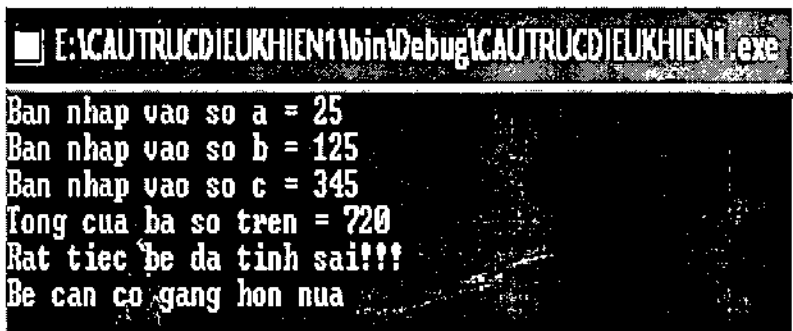


```

E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 20
Ban nhap vao so b = 40
Ban nhap vao so c = 60
Tong cua ba so tren = 120
Hoan ho be da tinh dung.

```

H22 Khi nhập a=20, b=40, c=60 và kết quả là 120



```

E:\CAUTRUCDIEUKHIEN1\bin\Debug\CAUTRUCDIEUKHIEN1.exe
Ban nhap vao so a = 25
Ban nhap vao so b = 125
Ban nhap vao so c = 345
Tong cua ba so tren = 720
Rat tiec be da tinh sai!!!
Be can co gang hon nua

```

H23 Khi nhập kết quả tính tổng sai

Phát biểu if có else. Nếu điều kiện sau if đúng thì sẽ thực hiện câu lệnh sau if. Ngược lại sẽ thực hiện câu lệnh sau else.

if (điều kiện)

s1;

else

s2;

Ghi chú:

Bạn ngầm hiểu s1, s2 có thể là một hoặc nhiều phát biểu, nếu từ hai trở lên thì ta phải dùng cặp dấu móc.

BÀI 4

CẤU TRÚC IF ELSE LỒNG NHAU

Viết chương trình nhập vào điểm thi đại học của ba môn Toán, Lý và Hóa. Nếu tổng số điểm của ba môn lớn hơn hoặc 25 thì sẽ báo là đậu vào đại học. Ngược lại, xét tiếp điều kiện, nếu tổng số điểm của 3 môn chỉ cần lớn hơn hoặc bằng 21 và mỗi môn phải lớn hơn hoặc bằng 7 thì cũng đậu. Các trường hợp còn lại sẽ không đủ tiêu chuẩn.

* Vào **Microsoft Visual studio .NET**

* Click chuột vào menu **File**, chọn **New**, chọn **Project**

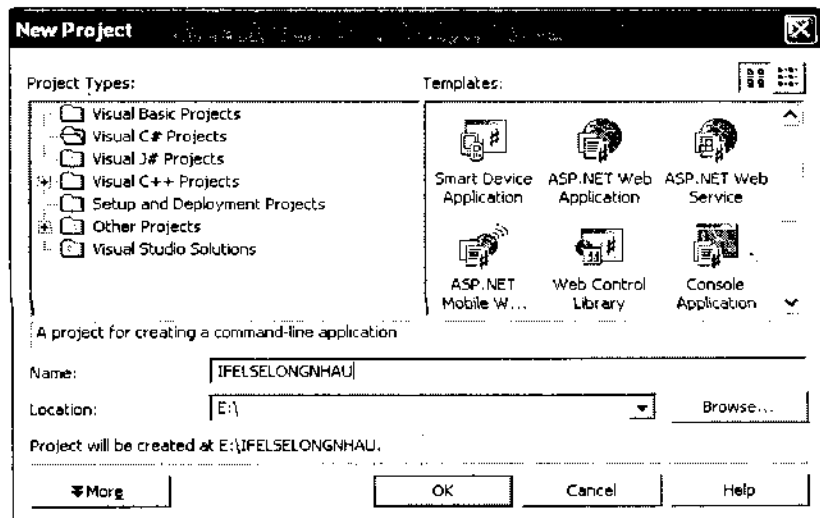
* Trong cửa sổ **New Project** hiện ra, bạn chọn:

* **Visual C# Project** trong khung bên trái có tên **Project Types**.

* Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.

Application.

* Đặt tên trong khung **Name**, giả sử là tên **IFELSELONGNHAU** như hình sau:



* Click nút OK trên hộp thoại.

Viết code như sau vào sau dòng ghi chú (như các bài tập trước)

```
// TODO: Add code to start application here
float toan, ly, hoa, ketqua;
Console.WriteLine("Nhap diem Toan = ");
toan=float.Parse(Console.ReadLine());
Console.WriteLine("Nhap diem Ly = ");
ly=float.Parse(Console.ReadLine());
Console.WriteLine("Nhap diem hoa = ");
hoa=float.Parse(Console.ReadLine());
ketqua=toan+ly+hoa;
if (ketqua>=25)
{
    Console.WriteLine("Thi sinh nay duoc vao dai hoc.");
    Console.WriteLine("Khong can xet diem khong che");
}
else
{
    if ((ketqua>=21)&&(toan>=7)&&(ly>=7)&&(hoa>=7))
        Console.WriteLine("Duoc vao dai hoc vi cac mon tu 7 tro len");
    else
        Console.WriteLine("Khong du tieu chuan vao dai hoc");
}
Console.ReadLine();
```

Giải thích:

- Khai báo các biến toan, ly, hoa và ketqua có kiểu số thực (float)
- Nhập điểm Toán, Lý và Hóa, các phương thức nhập xuất và chuyển đổi kiểu xem giải thích ở các bài trước.
- Gán tổng số điểm của ba môn cho biến có tên ketqua.
- Nếu ketqua>=25. Báo hai phát biểu theo sau if.
- Ngược lại (tức ketqua<25), xét trường hợp nếu đồng thời ketqua>=21 và điểm của từng môn đều lớn hơn hoặc bằng 7 thì báo câu lệnh theo sau phát biểu if tương ứng. Ngược lại báo câu lệnh theo sau else cuối cùng.

Code đầy đủ của bài này như sau: (phần code còn lại là do phần mềm tạo sẵn)

```
using System;
namespace IFELSELONGNHAU
{
    /// <summary>
    /// Summary description for Class1.
    /// </summary>
    class Class1
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main(string[] args)
        {
            //
            // TODO: Add code to start application here
            float toan, ly, hoa, ketqua;
            Console.Write("Nhap diem Toan = ");
            toan=float.Parse(Console.ReadLine());
            Console.Write("Nhap diem Ly = ");
            ly=float.Parse(Console.ReadLine());
            Console.Write("Nhap diem hoa = ");
            hoa=float.Parse(Console.ReadLine());
            ketqua=toan+ly+hoa;
            if (ketqua>=25)
            {
                Console.WriteLine("Thi sinh nay duoc vao dai hoc.");
                Console.WriteLine("Khong can xet diem khong che");
            }
            else
            {
                if ((ketqua>=21)&&(toan>=7)&&(ly>=7)&&(hoa>=7))
```

```

        Console.WriteLine("Duoc vao dai hoc vi cac mon tu 7 tro
                           len");
    else
        Console.WriteLine("Khong du tieu chuan vao dai hoc");
    }
Console.ReadLine();
//
}
}
}
}

```

* Biên dịch và nhấn phím F5 để chạy thử chương trình, nhập vào điểm của ba môn Toán, Lý và Hóa. Giả sử chúng ta nhập điểm của ba môn lần lượt như sau:

```

E:\IFELSELONGNHAU\bin\Debug\IFELSELONGNHAU.exe
Nhap diem Toan = 8.5
Nhap diem Ly = 9.5
Nhap diem hoa = 7.0
Thi sinh nay duoc vao dai hoc.
Khong can xet diem khong che

```

H25 Tổng điểm của ba môn là 25

Thỏa điều kiện của phát biểu if đầu tiên

```

E:\IFELSELONGNHAU\bin\Debug\IFELSELONGNHAU.exe
Nhap diem Toan = 7.0
Nhap diem Ly = 7.5
Nhap diem hoa = 7.5
Duoc vao dai hoc vi cac mon tu 7 tro len

```

H26 Tổng điểm của ba môn là 22, các môn đều >=7

Không thỏa điều kiện của phát biểu if thứ nhất, thỏa điều kiện if thứ hai.

```
E:\IFELSELONGNHAU\bin\Debug\IFELSELONGNHAU.exe
Nhap diem Toan = 8.5
Nhap diem Ly = 9.0
Nhap diem hoa = 6.5
Khong du tieu chuan vao dai hoc
```

H27 Tổng điểm của ba môn là 24

Không thỏa điều kiện của phát biểu if thứ nhất và thứ hai, mặc dầu tổng số điểm của ba môn ≥ 21 , nhưng có môn Hóa là 6.5, nên không đủ tiêu chuẩn vào đại học.

```
E:\IFELSELONGNHAU\bin\Debug\IFELSELONGNHAU.exe
Nhap diem Toan = 10
Nhap diem Ly = 9
Nhap diem hoa = 6
Thi sinh nay duoc vao dai hoc.
Khong can xet diem khong che
```

H28 Tổng điểm của ba môn là 25

Mặc dầu một môn có điểm 6, nhưng vì tổng số điểm của ba môn bằng 25, thỏa điều kiện của đầu bài, nên thí sinh này đậu vào đại học.

BÀI 5

CẤU TRÚC CHỌN SWITCH CASE

Cấu trúc switch case có dạng như sau:

```
switch (biểu thức)
{
    case giá trị 1 : s1; break;
    case giá trị 2 : s2; break;
    .....
    .....
    case giá trị n : sn; break;
    [default : sm; break; ]
}
```

▪ Đầu tiên các giá trị được xem xét và đối chiếu với trị của biểu thức, nếu trị đó nằm trong case nào, thì câu lệnh (*hoặc nhiều lệnh*) sau case đó sẽ được thực hiện, sau đó gặp lệnh break sẽ thoát khỏi phát biểu switch. Nó sẽ kiểm tra từ trên xuống, nếu không có giá trị của case nào thoả, nó sẽ thực hiện câu lệnh (*hoặc nhiều lệnh*) theo sau default (có thể có hoặc không), và kết thúc, nếu không có default, nó sẽ kết thúc và không làm gì cả (vì không có giá trị nào sau case thoả).

Ghi chú:

- Phải có lệnh break sau mục default.
- Biểu thức và các giá trị 1, giá trị 2, ..., giá trị n có thể là dữ liệu kiểu rời rạc, kiểu nguyên, kiểu kí tự, kiểu luận lý. Đặc biệt đối với C# thì những biểu thức này có thể là dữ liệu kiểu chuỗi (string).
- Chúng ta có thể dùng lệnh goto case để chuyển đến một nhánh khác trong cấu trúc này.

Nhận xét:

- Phát biểu **switch** là một dạng làm gọn của phát biểu if. Mọi phát biểu switch đều có thể viết lại thành phát biểu if. Bạn nên dùng

switch khi lựa chọn nhiều trường hợp, ví dụ như khi xếp loại học lực của học sinh... Bạn có thể xem ví dụ sau:

Viết chương trình nhập vào một số nguyên là số của tháng, chương trình sẽ báo ra bằng chữ tương ứng với số vừa nhập. Nếu nhập một số khác 1 đến 12, không thực hiện gì hết.

* Vào **Microsoft Visual studio .NET**

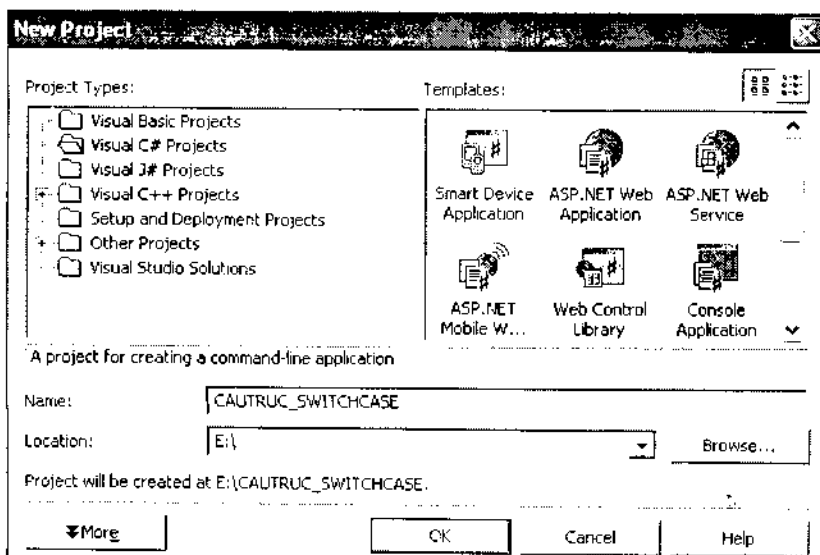
* Click chuột vào menu **File**, chọn **New**, chọn **Project**

* Trong cửa sổ **New Project** hiện ra, bạn chọn:

* **Visual C# Project** trong khung bên trái có tên **Project Types**.

* Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.

* Đặt tên **CAUTRUC_SWITCHCASE** trong khung **Name** như hình sau:



H29

* Click nút **OK** trên hộp thoại.

Bạn viết đoạn code này vào sau câu ghi chú của chương trình.

```
// TODO: Add code to start application here  
byte so;
```

```
Console.Write("Nhap so cua thang : ");
so=byte.Parse(Console.ReadLine());
switch (so)
{
    case 1:
        Console.Write("Thang Gieng");
        break;
    case 2:
        Console.Write("Thang hai");
        break;
    case 3:
        Console.Write("Thang ba");
        break;
    case 4:
        Console.Write("Thang tu");
        break;
    case 5:
        Console.Write("Thang nam");
        break;
    case 6:
        Console.Write("Thang sau");
        break;
    case 7:
        Console.Write("Thang bay");
        break;
    case 8:
        Console.Write("Thang tam");
        break;
    case 9:
        Console.Write("Thang chin");
        break;
    case 10:
        Console.Write("Thang muoi");
        break;
    case 11:
```

```

        Console.WriteLine("Thang muoi mot");
        break;
    case 12:
        Console.WriteLine("Thang muoi hai");
        break;
}
Console.ReadLine();

```

Giải thích:

- Khai báo biến số có kiểu nguyên (byte)
- Nhập vào một số, dùng phương thức Parse để đổi thành kiểu byte
- Biểu thức số được so sánh với các giá trị sau case, nếu giống, xuất ra màn hình câu lệnh sau case đó, lệnh break sẽ thoát khỏi cấu trúc switch case.

◆ Biên dịch và nhấn phím F5 chạy thử chương trình, giả sử nhập số 9. gõ phím Enter, bạn có kết quả như sau:

- Chạy lại chương trình, giả sử nhập vào số 12, bạn sẽ thấy kết quả như hình sau:

- Chạy lại chương trình, nhập vào số 15, bạn sẽ thấy kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
```

```
Nhap so cua thang : 15
```

H32

Không thực hiện.

Chúng ta thêm mục default như sau vào sau case cuối cùng:

default:

```
Console.WriteLine("Khong co thang nay");  
break;
```

Cuối cùng ta có đoạn code như sau:

```
// TODO: Add code to start application here
```

```
byte so;
```

```
Console.WriteLine("Nhap so cua thang : ");
```

```
so=byte.Parse(Console.ReadLine());
```

```
switch (so)
```

```
{
```

```
case 1:
```

```
    Console.WriteLine("Thang Gieng");
```

```
    break;
```

```
case 2:
```

```
    Console.WriteLine("Thang hai");
```

```
    break;
```

```
case 3:
```

```
    Console.WriteLine("Thang ba");
```

```
    break;
```

```
case 4:
```

```
    Console.WriteLine("Thang tu");
```

```
    break;
```

```
case 5:
```

```
    Console.WriteLine("Thang nam");
```

```
    break;
```

```
case 6:
```

```
    Console.WriteLine("Thang sau");
```

```

        break;
    case 7:
        Console.WriteLine("Thang bay");
        break;
    case 8:
        Console.WriteLine("Thang tam");
        break;
    case 9:
        Console.WriteLine("Thang chin");
        break;
    case 10:
        Console.WriteLine("Thang muoi");
        break;
    case 11:
        Console.WriteLine("Thang muoi mot");
        break;
    case 12:
        Console.WriteLine("Thang muoi hai");
        break;
    default:
        Console.WriteLine("Khong co thang nay");
        break;
}
Console.ReadLine();

```

- Nhấn phím F5 để chạy thử chương trình, nhập số 24, chúng ta sẽ có kết quả như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
Nhap so cua thang : 24
Khong co thang nay

```

H33

Vì không có giá trị nào sau case là 24, nên sẽ thực hiện các câu lệnh nằm sau default. Do đó sẽ hiện câu "Khong co thang nay"

Nếu bạn quy ước nếu bạn nào nhập từ 1 đến 12, thì sẽ in ra tháng tương ứng. Nếu nhập số 15 thì sẽ quay lại nhánh case có giá trị là 12. còn lại sẽ thông báo câu lệnh sau default.

Bạn thử sửa code lại như sau:

```
// TODO: Add code to start application here
```

```
byte so;
```

```
Console.Write("Nhap so cua thang. : ");
```

```
so=byte.Parse(Console.ReadLine());
```

```
switch (so)
```

```
{
```

```
    case 1:
```

```
        Console.Write("Thang Gieng");
```

```
        break;
```

```
    case 2:
```

```
        Console.Write("Thang hai");
```

```
        break;
```

```
    case 3:
```

```
        Console.Write("Thang ba");
```

```
        break;
```

```
    case 4:
```

```
        Console.Write("Thang tu");
```

```
        break;
```

```
    case 5:
```

```
        Console.Write("Thang nam");
```

```
        break;
```

```
    case 6:
```

```
        Console.Write("Thang sau");
```

```
        break;
```

```
    case 7:
```

```
        Console.Write("Thang bay");
```

```
        break;
```

```
    case 8:
```

```
        Console.Write("Thang tam");
```

```
        break;
```

```
    case 9:
```

```

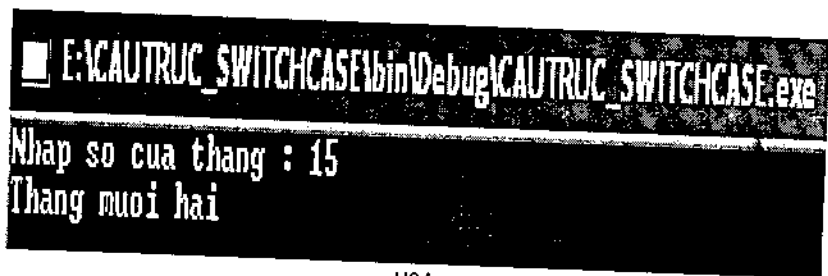
        Console.WriteLine("Thang chin");
        break;
    case 10:
        Console.WriteLine("Thang muoi");
        break;
    case 11:
        Console.WriteLine("Thang muoi mot");
        break;
    case 12:
        Console.WriteLine("Thang muoi hai");
        break;
    case 15:
        goto case 12;
    default:
        Console.WriteLine("Khong co thang nay");
        break;
}

Console.ReadLine();

```

Giải thích:

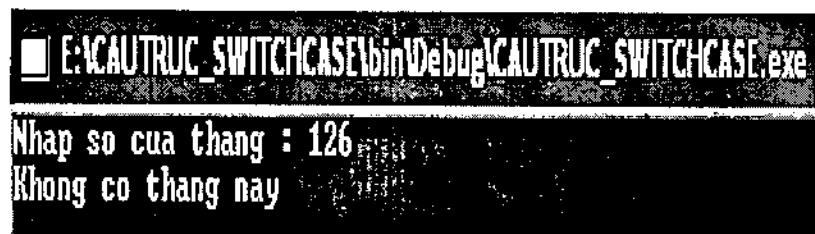
- Nếu nhập số **15**, thỏa với giá trị theo sau **case 15**: nên sẽ thực hiện câu lệnh **goto case 12**; sẽ chuyển đến nhánh **case** có giá trị **12**, do đó sẽ thực hiện câu lệnh nằm sau **case 12**.
- Nhấn phím **F5** để chạy thử chương trình, nhập số **15**, chúng ta sẽ có kết quả như hình sau:



H34

Bạn thấy sẽ thực hiện câu lệnh của case 12.

- Chạy lại chương trình, giả sử nhập một số khác 1 đến 12 và số 15 thì sẽ thực hiện câu lệnh sau default. Giả sử nhập số 126, chúng ta sẽ có kết quả như hình sau:



H35

Như đã đề cập ở trên, trong C# những biểu thức điều kiện có thể là kiểu chuỗi (string)

Sửa lại chương trình nhập vào một tháng trong năm có kiểu chuỗi, nếu là:

Tháng giêng thì nhập **thang1**

Tháng hai thì nhập là **thang2**

Tháng ba thì nhập là **thang3**

Tháng tư thì nhập là **thang4**

Tháng năm thì nhập là **thang5**

Tháng sáu thì nhập là **thang6**

Tháng bảy thì nhập là **thang7**

Tháng tám thì nhập là **thang8**

Tháng chín thì nhập là **thang9**

Tháng mười thì nhập là **thang10**

Tháng mười một thì nhập là **thang11**

Tháng mười hai thì nhập là **thang12**

Chương trình sẽ báo ra một chuỗi là tên của tháng đó.

◆ Bạn vào cửa sổ viết code và sửa lại như sau:

```
string thang;
Console.Write("Nhập số của tháng : ");
thang=Console.ReadLine();
switch (thang)
{
```

```
case "thang1":  
    Console.WriteLine("Thang Gieng");  
    break;  
case "thang2":  
    Console.WriteLine("Thang hai");  
    break;  
case "thang3":  
    Console.WriteLine("Thang ba");  
    break;  
case "thang4":  
    Console.WriteLine("Thang tu");  
    break;  
case "thang5":  
    Console.WriteLine("Thang nam");  
    break;  
case "thang6":  
    Console.WriteLine("Thang sau");  
    break;  
case "thang7":  
    Console.WriteLine("Thang bay");  
    break;  
case "thang8":  
    Console.WriteLine("Thang tam");  
    break;  
case "thang9":  
    Console.WriteLine("Thang chin");  
    break;  
case "thang10":  
    Console.WriteLine("Thang muoi");  
    break;  
case "thang11":  
    Console.WriteLine("Thang muoi mot");  
    break;  
case "thang12":  
    Console.WriteLine("Thang muoi hai");
```

```

        break;
    default:
        Console.WriteLine("Khong co thang nay");
        break;
}
Console.ReadLine();

```

Giải thích:

- Khai báo biến thang có kiểu string
- Hiện ra câu "Nhap so cua thang :"
- Nhập vào tên của tháng, đó là (thang1, thang2, ..., thang12)
- Cấu trúc switch sẽ kiểm tra biến thang với các giá trị theo sau case. Nếu thỏa sẽ xuất hiện ra màn hình tên của tháng tương ứng.

Ghi chú:

- Giá trị sau case là chuỗi, các bạn nhớ có cặp dấu nháy kép.
- Biên dịch và chạy thử chương trình, giả sử nhập vào thang5. Bạn sẽ có kết quả như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
Nhap so cua thang : thang5
Thang nam_

```

H36

- Nếu nhập thang10, chúng ta sẽ có kết quả thông báo như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
Nhap so cua thang : thang10
Thang muoi_

```

H37

Viết chương trình nhập vào hai số a và b, nhập các phép toán cộng (+), trừ (-), nhân (*), chia (/) và tính số dư (%). Chương trình sẽ xuất hiện kết quả.

Giả sử bạn đã vào Microsoft Visual Studio .NET. Bạn sẽ viết code như sau vào sau câu ghi chú:

```
float a, b, ketqua;
Console.Write("Nhap so a = ");
a=float.Parse(Console.ReadLine());
Console.Write("Nhap so b = ");
b=float.Parse(Console.ReadLine());
string toantu;
Console.Write("Nhap toan tu : ");
toantu=Console.ReadLine();
switch (toantu)
{
    case "+":
        Console.Write("a + b = ");
        ketqua = a + b;
        Console.Write(ketqua);
        break;
    case "-":
        Console.Write("a - b = ");
        ketqua = a - b;
        Console.Write(ketqua);
        break;
    case "*":
        Console.Write("a * b = ");
        ketqua = a * b;
        Console.Write(ketqua);
        break;
    case "/":
        Console.Write("a / b = ");
        ketqua = a / b;
        Console.Write(ketqua);
```

```

        break;
    case "%":
        Console.WriteLine("So du cua a / b = ");
        ketqua = a % b;
        Console.WriteLine(ketqua);
        break;
    default:
        Console.WriteLine("Ban da nhap sai toan tu");
        break;
}
Console.ReadLine();

```

Giải thích:

- Khai báo các biến a, b và ketqua có kiểu số thực (float)
- Nhập vào số a, số b
- Khai báo biến toantu có kiểu chuỗi (string)
- Nhập vào toán tử: (+, -, *, / hoặc %)
- So sánh toán tử vừa nhập với các giá trị theo sau các case, nếu thỏa sẽ thực hiện các câu lệnh của case đó.
- Gán biến ketqua = a + b
- In ra biến ketqua "Console.WriteLine(ketqua)"

Biên dịch và chạy thử chương trình, giả sử nhập a = 2, b = 4. Nhập dấu +. Ta được kết quả như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
Nhap so a = 2
Nhap so b = 4
Nhap toan tu : +
a + b = 6_

```

H38

- Chạy chương trình, giả sử nhập a = 2, b = 4. Nhập dấu -. Ta được kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
```

```
Nhap so a = 2  
Nhap so b = 4  
Nhap toan tu : -  
a - b = -2_
```

H39

■ Chạy chương trình, giả sử nhập $a = 2$, $b = 4$. Nhập dấu $-$. Ta được kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
```

```
Nhap so a = 2  
Nhap so b = 4  
Nhap toan tu : *  
a * b = 8
```

H40

■ Chạy chương trình, giả sử nhập $a = 2$, $b = 4$. Nhập dấu $/$. Ta được kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
```

```
Nhap so a = 2  
Nhap so b = 4  
Nhap toan tu : /  
a / b = 0.5_
```

H41

■ Chạy chương trình, giả sử nhập $a = 5$, $b = 2$. Nhập dấu $\%$. Ta được kết quả như hình sau:

```
Nhap so a = 5
Nhap so b = 2
Nhap toan tu : %
So du cua a / b = 1
```

H42

Đôi khi người nhập thường nhầm phép nhân trong máy tính là x hoặc X (thay vì đúng là *). Bạn có thể sửa lại chương trình như sau:

```
float a, b, ketqua;
Console.Write("Nhap so a = ");
a=float.Parse(Console.ReadLine());
Console.Write("Nhap so b = ");
b=float.Parse(Console.ReadLine());
string toantu;
Console.Write("Nhap toan tu : ");
toantu=Console.ReadLine();
switch (toantu)
{
    case "+":
        Console.Write("a + b = ");
        ketqua = a + b;
        Console.Write(ketqua);
        break;
    case "-":
        Console.Write("a - b = ");
        ketqua = a - b;
        Console.Write(ketqua);
        break;
    case "**":
        Console.Write("a * b = ");
        ketqua = a * b;
```

```

    Console.WriteLine(ketqua);
    break;
case "/":
    Console.WriteLine("a / b = ");
    ketqua = a / b;
    Console.WriteLine(ketqua);
    break;
case "%":
    Console.WriteLine("So du cua a / b = ");
    ketqua = a % b;
    Console.WriteLine(ketqua);
    break;
case "x":
case "X":
    goto case "**";
default:
    Console.WriteLine("Ban da nhap sai toan tu");
    break;
}
Console.ReadLine();

```

■ Bạn thêm các lệnh in đậm vào vị trí như trên, nếu toán tử nhập vào là "x" hoặc "X" thì sẽ thực hiện lệnh goto case "**"; trở về nhánh có case là "**" và thực hiện phép nhân.

Viết hai cách sau đều đúng. Bạn thử suy nghĩ tại sao chúng ta không viết:

```

case "x":
    goto case "**";
case "X":
    goto case "**";

```

mà viết:

```

case "x":
case "X":
    goto case "**";

```

Bạn thử suy nghĩ và tự giải thích!!!

BÀI 6

CẤU TRÚC LẬP FOR

Cấu trúc này thường được dùng khi xác định được số lần lập.
for ([khởi động giá trị đầu]; [điều kiện]; [các lệnh trước khi lập])

Ví dụ:

Viết chương trình xuất hiện ra 12 tháng trong năm.

* Vào Microsoft Visual Studio .NET

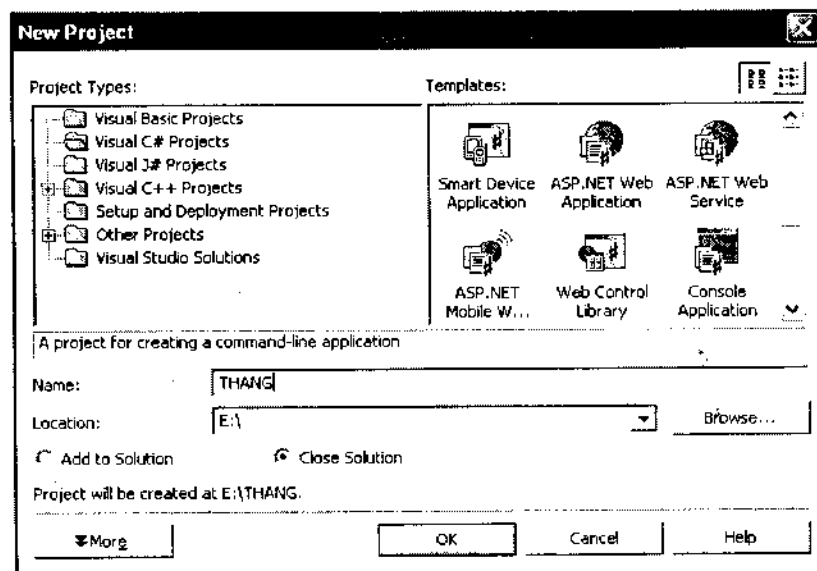
* Click chuột vào menu **File**, chọn **New**, chọn **Project**

* Trong cửa sổ **New Project** hiện ra, bạn chọn:

* **Visual C# Project** trong khung bên trái có tên **Project Types**.

* Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.

* Đặt tên **THANG** trong khung **Name** như hình sau:



H43

* Click nút **OK**.

Viết code của chương trình sau câu ghi chú như sau:

```
// TODO: Add code to start application here
```

```
byte x;
```

```
for (x=1; x<=12; x++)
```

```
{
```

```
    Console.WriteLine("DAY LA THANG: ");
```

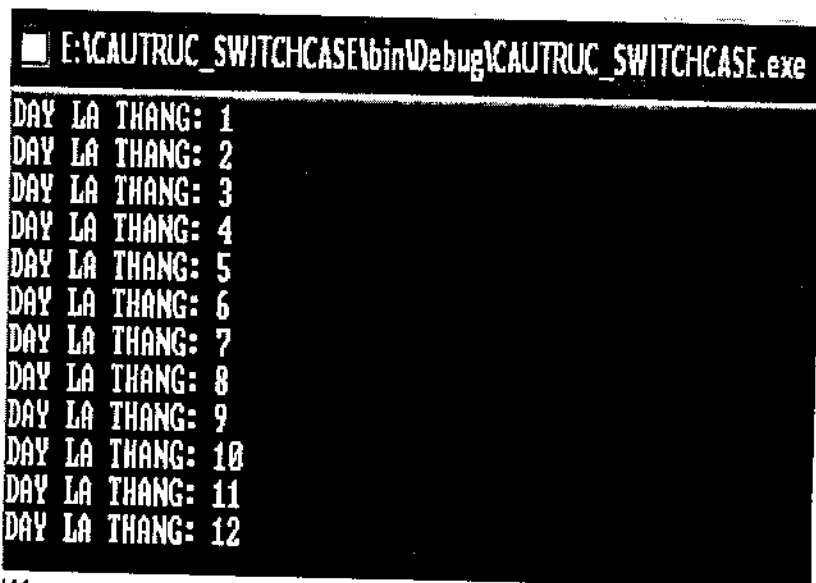
```
    Console.WriteLine(x);
```

```
}
```

```
Console.ReadLine();
```

Giải thích:

- Khai báo biến x có kiểu nguyên (byte).
- Vòng lặp for với giá trị ban đầu của x=1, thực hiện khi x còn nhỏ hơn hoặc bằng 12, tăng biến x lên 1 đơn vị sau mỗi lần lặp (x++ sẽ tăng lên 1 đơn vị so với giá trị của x trước đó)
- Biên dịch và chạy thử chương trình, bạn sẽ thấy kết quả như hình sau:



```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
DAY LA THANG: 1
DAY LA THANG: 2
DAY LA THANG: 3
DAY LA THANG: 4
DAY LA THANG: 5
DAY LA THANG: 6
DAY LA THANG: 7
DAY LA THANG: 8
DAY LA THANG: 9
DAY LA THANG: 10
DAY LA THANG: 11
DAY LA THANG: 12
```

H44

- Bạn có thể dùng vòng lặp for lặp từ số lớn đến số nhỏ.
- Vào cửa sổ viết code, sửa lại cấu trúc lặp for như sau:

```

byte x;
for (x=12; x>=1; x--)
{
    Console.Write("DAY LA THANG: ");
    Console.WriteLine(x);
}
Console.ReadLine();

```

Giải thích:

- Vòng lặp for với giá trị ban đầu của x là 12, sau mỗi lần lặp sẽ giảm 1 đơn vị so với giá trị của x trước đó. (x -- sẽ giảm 1 đơn vị sau mỗi lần lặp)
- Nhấn F5 chạy thử chương trình, bạn sẽ thấy kết quả như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
DAY LA THANG: 12
DAY LA THANG: 11
DAY LA THANG: 10
DAY LA THANG: 9
DAY LA THANG: 8
DAY LA THANG: 7
DAY LA THANG: 6
DAY LA THANG: 5
DAY LA THANG: 4
DAY LA THANG: 3
DAY LA THANG: 2
DAY LA THANG: 1

```

H45

Ví dụ:

Viết chương trình tính tổng các số lẻ từ 1 đến 2006

Bạn tạo một Project mới với một tên gì đó, viết code như sau:

```

int x;
long tong=0;

```

```
for (x=1; x<2006; x=x+2)
```

```
tong=tong+x;
```

```
Console.Write("TONG CAC SO LE TU 1 DEN 2006 LA: ");
```

```
Console.WriteLine(tong);
```

```
Console.ReadLine();
```

Giải thích:

- Khai báo biến `x` có kiểu nguyên (`int`)
- Khai báo biến `tong` có kiểu số nguyên (`long`) và gán cho biến này có giá trị là 0.
- Vòng lặp `for` có giá trị ban đầu `x = 1`, thực hiện khi `x` còn nhỏ hơn 2006, tăng lên 2 đơn vị sau mỗi lần lặp so với giá trị trước đó của `x`. Đó chính là các số lẻ từ 1 đến 2005 (1, 3, 5, 7, ..., 2003, 2005).
- Cứ mỗi lần lặp của vòng lặp `for` sẽ lấy biến `tong` cộng với giá trị của `x`.
- Báo ra màn hình lời ghi chú và giá trị của biến `tong` tương ứng với lần lặp cuối cùng tức là `x = 2005` (tăng thêm 2 đơn vị nữa, lúc này `x` sẽ là 2007. Không thỏa điều kiện `x < 2006`. Vòng lặp kết thúc).
- Biên dịch và chạy thử chương trình, chúng ta sẽ thấy kết quả như sau:



H46

Ví dụ:

Nhập và tính tổng của 5 số thực được nhập vào.

Bạn viết code như sau:

```
int x;
```

```
float so,tong=0;
```

```
for (x=1; x<=5; x++)
```

```
{  
    Console.Write("BAN NHAP SO THU ");  
    Console.Write(x);
```

```

Console.Write(" = ");
so=float.Parse(Console.ReadLine());
tong=tong+so;
}
Console.Write("TONG 5 SO THUC LA: ");
Console.WriteLine(tong);
Console.ReadLine();

```

Giải thích:

- Khai báo biến x có kiểu số nguyên (int)
- Khai báo biến so và tong có kiểu số thực (float)
- Vòng lặp for chạy từ 1 đến 5 (tức là lặp 5 lần)
- Hiện ra câu nhắc để nhập số thực
- Đổi biến so có kiểu số thực
- Cộng số thực vừa nhập vào với biến tong là số trước đó
- Báo ra tổng của 5 số thực vừa nhập vào

Biên dịch và chạy thử chương trình, giả sử nhập vào 5 số thực lần lượt như trong hình, đó là 12.5, 20.4, 10.1, 34.6, 235.8. Chúng ta sẽ có kết quả như hình dưới

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe

```

```

BAN NHAP SO THU 1 = 12.5
BAN NHAP SO THU 2 = 20.4
BAN NHAP SO THU 3 = 10.1
BAN NHAP SO THU 4 = 34.6
BAN NHAP SO THU 5 = 235.8
TONG 5 SO THUC LA: 313.4

```

H47

BÀI 7

CẤU TRÚC LẶP WHILE/DO WHILE

Các vòng lặp này dừng khi biểu-thức [Điều kiện] nhận giá trị sai.

while (Điều kiện)

```
{  
    S1;  
    S2;  
    .....  
    Sn  
}
```

Ví dụ:

Chúng tôi lấy bài tập vừa thực hiện ở vòng lặp for ở trên, đó là nhập vào 5 số thực, tính tổng của 5 số thực đó. Nhưng yêu cầu các bạn phải dùng vòng lặp while.

- * Vào Microsoft Visual Studio .NET
- * Click chuột vào menu **File**, chọn **New**, chọn **Project**
- * Trong cửa sổ **New Project** hiện ra, bạn chọn:
- * **Visual C# Project** trong khung bên trái có tên **Project Types**.
- * Di chuyển con chạy bên khung bên phải, tìm chọn **Console**

Application.

- * Đặt tên cho project trong khung **Name**

Viết code vào sau dòng ghi chú của chương trình như sau:

```
int x=1;  
float so,tong=0;  
while (x<6)  
{  
    Console.Write("BAN NHAP SO THU ");  
    Console.Write(x);  
    Console.Write(" = ");
```

```

so=float.Parse(Console.ReadLine());
tong=tong+so;
x=x+1;
}
Console.Write("TONG 5 SO THUC LA: ");
Console.WriteLine(tong);
Console.ReadLine();

```

Giải thích:

- Khai báo biến x có kiểu số nguyên (int)
- Khai báo biến so và tong có kiểu số thực (float)
- Vòng lặp while thực hiện từ từ x=1 đến khi x< 6 (tức là đến x=5)
- Hiện ra câu nhắc để nhập số thực
- Đổi biến so có kiểu số thực
- Cộng số thực vừa nhập vào với biến tong là số trước đó
- Tăng biến x lên một đơn vị
- Báo ra tổng của 5 số thực vừa nhập vào

Biên dịch và chạy thử chương trình, giả sử nhập vào 5 số thực lần lượt là: 5.1, 5.2, 5.3, 5.4, 5.5. Bạn sẽ thấy kết quả như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP SO THU 1 = 5.1
BAN NHAP SO THU 2 = 5.2
BAN NHAP SO THU 3 = 5.3
BAN NHAP SO THU 4 = 5.4
BAN NHAP SO THU 5 = 5.5
TONG 5 SO THUC LA: 26.5

```

H48

Bạn thấy kết quả cũng như vòng lặp for ở bài trước.

đo

```

{
    S1;
    S2;
    .....
    Sn;
} while (Điều kiện);

```

Cũng thực hiện nội dung của bài tập trên, nhưng dùng phát biểu lặp do ... while vừa nêu.

```

int x=1;
float so,tong=0;
do
{
    Console.Write("BAN NHAP SO THU ");
    Console.Write(x);
    Console.Write(" = ");
    so=float.Parse(Console.ReadLine());
    tong=tong+so;
    x=x+1;
} while (x<6);
Console.Write("TONG 5 SO THUC LA: ");
Console.WriteLine(tong);
Console.ReadLine();

```

Giải thích:

Bạn tự tìm hiểu đoạn code trên.

☐ Phát biểu **break**

Phát biểu này dùng để ngưng các vòng lặp, mặc dù điều kiện của vòng lặp còn đúng. Các bạn đã biết phát biểu này trong cấu trúc switch case.

Bạn thử thêm phát biểu break trong code của bài tập trên như sau:

```

int x=1;
float so,tong=0;
do
{

```



```

Console.Write("BAN NHAP SO THU ");
Console.Write(x);
Console.Write(" = ");
so=float.Parse(Console.ReadLine());
tong=tong+so;
x=x+1;
if (x==3)
    break;
} while (x<6);
Console.Write("TONG 5 SO THUC LA: ");
Console.WriteLine(tong);
Console.ReadLine();

```

▪ Bạn thêm phát biểu if, nếu $x = 3$ thì sẽ thực hiện phát biểu break. Lúc này $x = 3$ vẫn còn thỏa điều kiện trong vòng lặp do ... while ở trên, nhưng nó vẫn thoát ra khỏi vòng lặp.

▪ Chạy thử chương trình nhập 5 số thực. Bạn chỉ nhập được hai số thì vòng lặp kết thúc, vì lúc này biến $x = 3$. Các câu lệnh phía sau vòng lặp sẽ được thực hiện. Bạn xem hình sau:

```

BAN NHAP SO THU 1 = 2.5
BAN NHAP SO THU 2 = 7.5
TONG 5 SO THUC LA: 10

```

H49

Chúng tôi nhập số thứ nhất là 2.5, số thứ hai là 7.5, lúc này $x = 3$ nên thực hiện phát biểu break thoát ra khỏi vòng lặp và thực hiện các câu lệnh sau vòng lặp.

☐ Phát biểu goto

Bạn đã biết phát biểu này trong cấu trúc switch case. Bạn xem ví dụ sau:

```

char nhayden;
int x=1;

```

```

float so,tong=0;
do
{
    Console.Write("BAN NHAP SO THU ");
    Console.Write(x);
    Console.Write(" = ");
    so=float.Parse(Console.ReadLine());
    tong=tong+so;
    x=x+1;
    if (x==3)
        goto nhayden;
} while (x<6);
Console.Write("TONG 5 SO THUC LA: ");
Console.WriteLine(tong);
nhayden::
if (x==3)
    Console.Write("TAI SAO KHONG BAO RA TONG CUA 5 SO THUC!!!");
Console.ReadLine();

```

Giải thích:

- Khai báo thêm biến nhayden có kiểu char
- Khi biến x = 3 thì thực hiện phát biểu goto nhayden; lệnh này nhảy đến nhãn nhayden; vì x = 3 nên thực hiện câu lệnh của phát biểu if đó là xuất hiện câu "TAI SAO KHONG BAO RA TONG CUA 5 SO THUC!!!"
- Chạy thử chương trình, bạn sẽ thấy kết quả như sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe

```

```

BAN NHAP SO THU 1 = 123.7

```

```

BAN NHAP SO THU 2 = 3456.5

```

```

BAN BIET TAI SAO KHONG BAO RA TONG CUA 5 SO THUC!!!

```

BÀI 8

MẢNG TRONG C#

Để khai báo nhiều biến có cùng tên nhưng phân biệt nhau bởi chỉ số, chúng ta dùng dữ liệu mảng. Mảng trong C# quy ước khởi đầu với chỉ số 0.

☞ Mảng một chiều

Kiểu của mảng[] tên mảng;

Ví dụ:

Int [] cam;

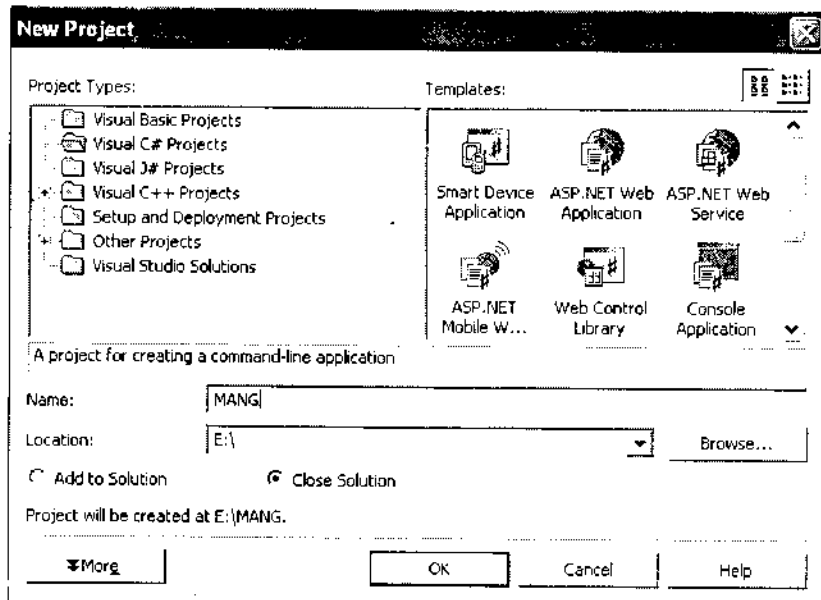
Float [] so;

Mảng trong C# được cấp phát bộ nhớ bằng toán tử **new**. Bạn sẽ thấy trong bài tập thực hành.

Một đặc điểm của C# không giống với các ngôn ngữ truyền thống C và C++ đó là người lập trình không cần quan tâm đến việc giải phóng bộ nhớ, hệ thống sẽ thực hiện việc này nhờ cơ chế nhặt rác. Bạn xem một bài tập về mảng như sau:

Viết chương trình nhập vào chỉ số mảng bất kỳ, nhập các phần tử đó vào, xuất các phần tử của mảng vừa nhập ra màn hình.

- * Vào Microsoft Visual Studio .NET
- * Click chuột vào menu File, chọn **New**, chọn **Project**
- * Trong cửa sổ **New Project** hiện ra, bạn chọn:
 - * **Visual C# Project** trong khung bên trái có tên **Project Types**.
 - * Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.
- * Đặt tên **MANG** trong khung **Name** như hình sau:



H51

* Click nút OK

Viết code tại vị trí ghi chú của chương trình như sau:

```
float[] mang;
int i, sophantu;
Console.WriteLine("BAN NHAP SO PHAN TU CUA MANG: ");
sophantu=int.Parse(Console.ReadLine());
mang=new float[sophantu];
// Nhập các phần tử của mảng
for (i=0; i<sophantu; i++)
{
    Console.WriteLine("GIA TRI PHAN TU THU ");
    Console.WriteLine(i);
    Console.WriteLine(" LA : ");
    mang[i]=float.Parse(Console.ReadLine());
}
// Xuất các phần tử của mảng ra màn hình
for (i=0; i<sophantu; i++)
```

```

{
    Console.Write("PHAN TU THU ");
    Console.Write(i);
    Console.Write(" LA : ");
    Console.WriteLine(mang[i]);
}
Console.ReadLine();

```

Giải thích:

- Khai báo một mảng kiểu số thực (float) có tên là mang.
- Khai báo biến i và biến sophantu có kiểu nguyên. Biến i dùng trong vòng lặp for, biến sophantu là số phần tử của mảng.
- Nhập vào số phần tử của mảng.
- Cấp phát bộ nhớ cho mảng
- Vòng lặp for thứ nhất dùng để nhập các phần tử của mảng
- Vòng lặp thứ hai để xuất ra các phần tử của mảng
- Biên dịch và chạy thử chương trình, giả sử bạn nhập mảng có 6 phần tử, sau đó nhập giá trị của 6 phần tử này. Nhập xong, bạn sẽ thấy kết quả các số vừa nhập hiện ra.

Nhìn vào hình sau để đối chiếu với những lời chúng tôi vừa nói ở trên.

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP SO PHAN TU CUA MANG: 6
GIA TRI PHAN TU THU 0 LA : 123.6
GIA TRI PHAN TU THU 1 LA : 564.5
GIA TRI PHAN TU THU 2 LA : 12.3
GIA TRI PHAN TU THU 3 LA : 45.9
GIA TRI PHAN TU THU 4 LA : 78.9
GIA TRI PHAN TU THU 5 LA : 456.8

```

■ Khi nhập số 6 chỉ số phần tử có trong mảng và giá trị của 6 phần tử như hình trên. Nếu gõ phím Enter, bạn sẽ thấy kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP SO PHAN TU CUA MANG: 6
GIA TRI PHAN TU THU 0 LA : 123.6
GIA TRI PHAN TU THU 1 LA : 564.5
GIA TRI PHAN TU THU 2 LA : 12.3
GIA TRI PHAN TU THU 3 LA : 45.9
GIA TRI PHAN TU THU 4 LA : 78.9
GIA TRI PHAN TU THU 5 LA : 456.8
PHAN TU THU 0 LA : 123.6
PHAN TU THU 1 LA : 564.5
PHAN TU THU 2 LA : 12.3
PHAN TU THU 3 LA : 45.9
PHAN TU THU 4 LA : 78.9
PHAN TU THU 5 LA : 456.8
```

H53

Bạn xem hình sau và tự tìm hiểu:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP SO PHAN TU CUA MANG: 10
GIA TRI PHAN TU THU 0 LA : 1
GIA TRI PHAN TU THU 1 LA : 2
GIA TRI PHAN TU THU 2 LA : 3
GIA TRI PHAN TU THU 3 LA : 4
GIA TRI PHAN TU THU 4 LA : 5
GIA TRI PHAN TU THU 5 LA : 6
GIA TRI PHAN TU THU 6 LA : 7
GIA TRI PHAN TU THU 7 LA : 8
GIA TRI PHAN TU THU 8 LA : 9
GIA TRI PHAN TU THU 9 LA : 10
PHAN TU THU 0 LA : 1
PHAN TU THU 1 LA : 2
PHAN TU THU 2 LA : 3
PHAN TU THU 3 LA : 4
PHAN TU THU 4 LA : 5
PHAN TU THU 5 LA : 6
PHAN TU THU 6 LA : 7
PHAN TU THU 7 LA : 8
PHAN TU THU 8 LA : 9
PHAN TU THU 9 LA : 10
```

H54

Các bạn có thể khai báo một mảng với các phần tử có sẵn bên trong, giả sử khai báo một mảng có 5 phần tử là các số nguyên như sau:

```
int[] so = new int[]{5, 25, 46, 57, 12};
```

Ví dụ:

Bạn viết chương trình xuất ra màn hình 5 phần tử của một mảng chứa 5 phần tử là các số nguyên lần lượt là: 2, 5, 89, 567, 123.

Bạn viết code như sau:

```
int[] mang=new int[]{2, 5, 89, 567, 123};
int i;
for (i=0; i<5; i++)
{
    Console.Write("PHAN TU THU ");
    Console.Write(i+1);
    Console.Write(" LA : ");
    Console.WriteLine(mang[i]);
}
Console.ReadLine();
```

Giải thích:

- Khai báo mảng một chiều tên mang có kiểu là int và có 5 phần tử là 2, 5, 89, 567 và 123
- Khai báo biến i để dùng trong vòng lặp for
- Vòng lặp thực hiện 5 lần
- Biến i + 1 để xuất ra phần tử đầu tiên là phần tử 1
- Biên dịch và chạy thử chương trình, bạn sẽ có kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
PHAN TU THU 1 LA : 2
PHAN TU THU 2 LA : 5
PHAN TU THU 3 LA : 89
PHAN TU THU 4 LA : 567
PHAN TU THU 5 LA : 123
```

H55

Qua bài tập này, các bạn thấy ví khai báo các phần tử có trong mảng, nên chúng ta bỏ qua vòng lặp for để nhập chúng.

➤ Mảng hai chiều

Kiểu của mảng[,] tên mảng;

Ví dụ:

Int [,] so;

float [,] cacso;

Tương tự như mảng một chiều, ở mảng hai chiều, các bạn cũng có thể gán hoặc truy xuất các phần tử của mảng. Bạn có thể tham khảo các ví dụ sau:

Ví dụ:

Viết chương trình khai báo mảng hai chiều mang[3,3]. Nhập các phần tử của mảng vào, in ra màn hình các phần tử của mảng hai chiều này. Bạn thực hiện như sau:

* Vào Microsoft Visual Studio .NET

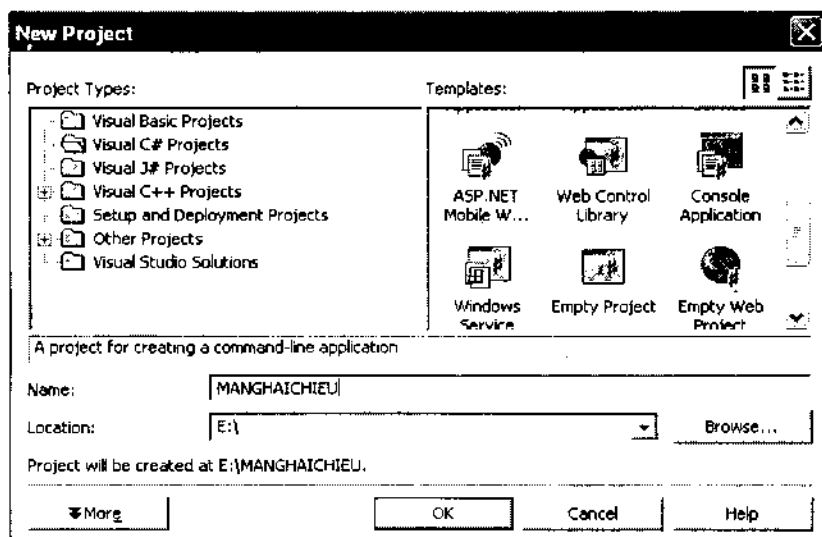
* Click chuột vào menu **File**, chọn **New**, chọn **Project**

* Trong cửa sổ **New Project** hiện ra, bạn chọn:

* **Visual C# Project** trong khung bên trái có tên **Project Types**.

* Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.

* Đặt tên **MANGHAICHIEU** trong khung **Name** như hình sau:



* Click nút OK.

Viết code cho chương trình sau câu ghi chú là:

```
int[ , ] mang;  
int i, j;  
mang=new int[3,3];  
for (i=0; i<3; i++)  
    for (j=0; j<3; j++)  
    {  
        Console.WriteLine("BAN NHAP PHAN TU THU MANG[");  
        Console.WriteLine(i+1);  
        Console.WriteLine(",");  
        Console.WriteLine(j+1);  
        Console.WriteLine("] LA :");  
        mang[i,j]=int.Parse(Console.ReadLine());  
    }  
Console.WriteLine("CAC PHAN TU CUA MANG LA: ");  
for (i=0; i<3; i++)  
    for (j=0; j<3; j++)  
    {  
        Console.WriteLine(mang[i,j]);  
        Console.WriteLine(" ");  
    }  
Console.ReadLine();
```

Giải thích:

- Khai báo mảng hai chiều kiểu nguyên int có tên là mang
- Khai báo biến i và j có kiểu nguyên
- Cấp phát bộ nhớ cho mảng hai chiều có 3 hàng 3 cột
- Hai vòng lặp for lồng nhau dùng để nhập vào 9 phần tử của mảng
 - $i + 1$ và $j + 1$ trong các câu lệnh của hai vòng lặp for để xuất hiện đúng phần tử thứ 1, 1, phần tử thứ 1, 2, phần tử thứ 1, 3, phần tử thứ 2, 1... cho đến phần tử thứ 3, 3.
 - Trong hai vòng lặp for lồng nhau ở trên, đầu tiên biến $i = 0$, các biến j của vòng for bên trong sẽ chạy từ 0 đến 2. Sau đó biến i sẽ

tăng lên một tức là $i = 1$, các biến j của vòng for bên trong sẽ chạy từ 0 đến 2, sau đó biến i tăng lên một tức là $i = 2$, biến j sẽ chạy từ 0 đến 2. Biến i tăng lên một đơn vị, tức là $i = 3$, không thỏa điều kiện của ($i < 3$) nên kết thúc vòng lặp. Ta nhập tất cả là 9 phần tử.

- Xuất hiện trên màn hình câu "CAC PHAN TU CUA MANG LA: ". Nếu bạn để câu này trong vòng lặp for, nó sẽ xuất hiện lặp đi lặp lại nhiều lần.

- Hai vòng lặp for tiếp theo sẽ xuất 9 phần tử của mảng hai chiều ra màn hình.

- Biên dịch và chạy thử chương trình, bạn nhập 9 phần tử vào, giả sử chúng tôi nhập 9 phần tử bất kỳ, các bạn sẽ có kết quả như hình sau:

```
E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP PHAN TU THU MANG[1,1] LA :67
BAN NHAP PHAN TU THU MANG[1,2] LA :123
BAN NHAP PHAN TU THU MANG[1,3] LA :345
BAN NHAP PHAN TU THU MANG[2,1] LA :642
BAN NHAP PHAN TU THU MANG[2,2] LA :12356
BAN NHAP PHAN TU THU MANG[2,3] LA :34567
BAN NHAP PHAN TU THU MANG[3,1] LA :6788
BAN NHAP PHAN TU THU MANG[3,2] LA :89765
BAN NHAP PHAN TU THU MANG[3,3] LA :34567
CAC PHAN TU CUA MANG LA:
67 123 345 642 12356 34567 6788 89765 34567
```

H57

Khi chạy chương trình, bạn sẽ thấy trên màn hình hiện câu:

```
BAN NHAP PHAN TU THU MANG[1, 1] LA :
```

Chúng ta nhập 67 và gõ phím Enter, câu thứ hai hiện ra, chúng ta nhập 123, gõ phím Enter nhập tiếp tục số thứ ba, thực hiện tương tự cho đến khi nhập đến phần tử cuối cùng. Gõ phím Enter, bạn sẽ thấy xuất hiện

CAC PHAN TU CUA MANG LA:

67 123 345 642 12356 34567 6788 89765 34567

Gõ phím Enter, nhấn F5 chạy lại chương trình, bạn nhập thử một kết quả khác để kiểm chứng lại.

Bài trên chúng ta tự cho mảng hai chiều [3, 3] ngay từ trong chương trình. Nếu người sử dụng yêu cầu nhập các số này sẽ được nhập từ bàn phím khi chạy chương trình, bạn sửa code lại như sau:

```
int[ , ] mang;  
int i,j,hang,cot;  
Console.Write("BAN NHAP SO HANG CUA MANG: ");  
hang=int.Parse(Console.ReadLine());  
Console.Write("BAN NHAP SO COT CUA MANG: ");  
cot=int.Parse(Console.ReadLine());  
mang=new int[hang,cot];  
for (i=0; i<hang; i++)  
    for (j=0; j<cot; j++)  
        {  
            Console.Write("BAN NHAP PHAN TU THU MANG[");  
            Console.Write(i+1);  
            Console.Write(",");  
            Console.Write(j+1);  
            Console.Write("] LA :");  
            mang[i,j]=int.Parse(Console.ReadLine());  
        }  
Console.WriteLine("CAC PHAN TU CUA MANG LA: ");  
for (i=0; i<hang; i++)  
    for (j=0; j<cot; j++)  
        {  
            Console.Write(mang[i,j]);  
            Console.Write(" ");  
        }  
Console.ReadLine();
```

Giải thích:

- Khai báo mảng hai chiều kiểu nguyên int có tên là mang

■ Khai báo biến `i`, `j`, `hang`, `cot` có kiểu nguyên `int`. Biến `i` và `j` dùng cho vòng lặp `for`. Biến `hang` là số hàng của ma trận, biến `cot` là số cột của ma trận. Hai biến `hang` và `cot` sẽ được nhập vào.

- Cấp phát bộ nhớ cho mảng hai chiều này.
- Phần tiếp theo giống như bài trên.

Ghi chú:

Bạn không được cấp phát bộ nhớ cho mảng hai chiều trên khi chưa có giá trị của biến `hang` và biến `cot`.

Nếu bạn cấp phát trước, chương trình sẽ báo lỗi. Bạn xem đoạn code sau:

```
int[ , ] mang;  
int i,j,hang,cot;  
mang=new int[hang,cot];  
Console.Write("BAN NHAP SO HANG CUA MANG: ");  
hang=int.Parse(Console.ReadLine());  
Console.Write("BAN NHAP SO COT CUA MANG: ");  
cot=int.Parse(Console.ReadLine());  
for (i=0; i<hang; i++)  
    for (j=0; j<cot; j++)  
        {  
            Console.Write("BAN NHAP PHAN TU THU MANG[");  
            Console.Write(i+1);  
            Console.Write(",");  
            Console.Write(j+1);  
            Console.Write("] LA :");  
            mang[i,j]=int.Parse(Console.ReadLine());  
        }  
Console.WriteLine("CAC PHAN TU CUA MANG LA: ");  
for (i=0; i<hang; i++)  
    for (j=0; j<cot; j++)  
        {  
            Console.Write(mang[i,j]);  
            Console.Write(" ");  
        }  
Console.ReadLine();
```

- Nếu bạn thay đổi vị trí câu lệnh cấp phát bộ nhớ cho mảng hai chiều lên bên trên (câu lệnh đậm). Bạn sẽ bị chương trình báo lỗi.
- Biên dịch và chạy thử chương trình, giả sử nhập hang=2, cot=4, bạn sẽ thấy như hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP SO HANG CUA MANG: 2
BAN NHAP SO COT CUA MANG: 4

```

H58

Gõ phím Enter, bạn thấy hình sau:

```

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe
BAN NHAP SO HANG CUA MANG: 2
BAN NHAP SO COT CUA MANG: 4
BAN NHAP PHAN TU THU MANG[1,1] LA :

```

H59

- Bạn nhập giá trị thứ nhất của mảng vào, gõ phím Enter, và tiếp tục như vậy cho đến khi nhập xong 8 phần tử. Giả sử chúng tôi nhập 8 số nguyên bất kỳ, sau đó gõ phím Enter, các bạn xem kết quả như hình sau:

E:\CAUTRUC_SWITCHCASE\bin\Debug\CAUTRUC_SWITCHCASE.exe

```
BAN NHAP SO HANG CUA MANG: 2
BAN NHAP SO COT CUA MANG: 4
BAN NHAP PHAN TU THU MANG[1,1] LA :1
BAN NHAP PHAN TU THU MANG[1,2] LA :3
BAN NHAP PHAN TU THU MANG[1,3] LA :5
BAN NHAP PHAN TU THU MANG[1,4] LA :7
BAN NHAP PHAN TU THU MANG[2,1] LA :9
BAN NHAP PHAN TU THU MANG[2,2] LA :11
BAN NHAP PHAN TU THU MANG[2,3] LA :13
BAN NHAP PHAN TU THU MANG[2,4] LA :15
CAC PHAN TU CUA MANG LA:
1 3 5 7 9 11 13 15
```

H60

BÀI 9

LỚP VÀ ĐỐI TƯỢNG TRONG C#

Khái niệm lớp (**class**) trong các ngôn ngữ lập trình hướng đối tượng dùng để mô tả dữ liệu và các thao tác của các đối tượng trong thế giới thực. Các thành phần dữ liệu cũng được gọi các trường hay các thuộc tính. Các thao tác còn được gọi là hàm hay các phương thức. Chẳng hạn, để mô tả phân số thì thuộc tính sẽ là tử số và mẫu số còn các phương thức là các phép toán về phân số. Để mô tả một hình học thì các thuộc tính là các số đo kích thước còn các phương thức sẽ là các hàm tính diện tích, chu vi ...

Bạn xem các ví dụ sau để hiểu rõ hơn về lớp. Tiêu chí của chúng tôi là dùng những bài tập giản đơn nhất, với mục đích cuối cùng là giúp cho người đọc cảm thấy nhẹ nhàng, thoải mái và dễ thực hiện, dễ tiếp thu từ đó tạo sự thích thú của người đọc với ngôn ngữ lập trình, yêu thích lập trình và tìm tòi nghiên cứu thêm khi cần giải quyết những sự việc.

Bạn xem một lớp thể hiện cách tính diện tích và chu vi của một hình tròn.

```
class vongtron
{
    static double PI=3.1416;
    double R;
    public vongtron(double bk)
    {
        R=bk;
    }
    public double dientich()
    {
        return PI*R*R;
    }
    public double chuvi()

```

```

    {
        return 2*PI*R;
    }

```

};

Giải thích:

- Để tạo lớp, chúng ta dùng từ khóa **class** kế đến là tên của lớp cần tạo. Trong đoạn code trên, tên lớp là vongtron.

- Vì trong lớp sẽ có nhiều câu lệnh, chúng ta dùng dấu móc "{" để làm dấu mở bao các câu lệnh, sau các câu lệnh sẽ là dấu móc và dấu chấm phẩy ";" để đóng các câu lệnh, bạn hiểu một cách nôm na như vậy.

- Trong vòng tròn, bạn cần khai báo các thuộc tính cần thiết để có thể tính được diện tích và chu vi của vòng tròn. Bạn thấy có số Pi và bán kính R. Vậy chúng ta phải khai báo PI có kiểu số thực, R có kiểu số thực.

```
static double PI=3.1416;
```

```
double R;
```

- Trong lớp vongtron, khai báo một phương thức (ham) có tên vongtron với một tham số đó là bán kính. Từ khóa Public để có thể gọi được từ bên ngoài lớp. Gán R bằng giá trị trong phương thức. **Đây là phương thức khởi tạo dữ liệu** (constructor).

```
public vongtron(double bk)
```

```
{
    R=bk;
```

```
}
```

- Trong lớp vongtron, khai báo một phương thức (ham) có tên dientich(). Từ khóa Public để có thể gọi được từ bên ngoài lớp. Trả về giá trị là diện tích của hình tròn là $PI * R * R$.

```
public double dientich()
```

```
{
    return PI*R*R;
```

```
}
```

- Trong lớp vongtron, khai báo một phương thức (ham) có tên chuvi(). Từ khóa Public để có thể gọi được từ bên ngoài lớp. Trả về giá trị là chu vi của hình tròn là $2 * PI * R$.

Ghi chú:

Trong các phương thức ta thấy có phương thức **vongtron** trùng với tên của lớp (phương thức **vongtron**(double bk) ứng với lớp **vongtron**). Đây là phương thức tạo lập (**constructor**) có mục đích khởi tạo dữ liệu, những phương thức này tự động chạy ngay khi đối tượng của lớp được tạo lập. Cơ chế này nhằm khắc phục tình trạng các biến không được khởi tạo giá trị mặc nhiên (nhận giá trị rác). Sau khi xây dựng các lớp như trên, có thể sử dụng bằng cách khai báo các đối tượng của chúng như sau:

Bạn sẽ viết đoạn chương trình chính, trong đó sẽ gọi đến các phương thức của lớp vừa tạo ở trên.

```
class MainProgram
{
    static void Main()
    {
        vongtron vt=new vongtron(3);
        Console.WriteLine("Hình tron :");
        Console.WriteLine(" DT={0},CV={1}",vt.dientich(), vt.chuvi());
        Console.ReadLine();
    }
}
```

Giải thích:

- Đầu tiên là một lớp chương trình chính với hàm void Main().
- Khai báo biến vt có kiểu vongtron và cấp phát bộ nhớ với bán kính bằng 3.
- Xuất hiện ra màn hình diện tích và chu vi của hình tròn có bán kính là 3.

```
Console.WriteLine(" DT={0},CV={1}",vt.dientich(), vt.chuvi());
```

- Câu lệnh trên trong phương thức WriteLine, Giá trị của biến dientich và chuvi được nhúng vào chuỗi dữ liệu định dạng, hai đối số vt.dientich(), vt.chuvi() được đánh chỉ số từ 0 đến 1. Trong dòng lệnh trên là kiểu định dạng mặc nhiên đơn giản nhất, ta chỉ việc ghi chỉ số biến kẹp trong hai dấu móc { và }, đó là {0}, {1}, {0} sẽ được thay thế bởi vt.dientich() và {1} được thay thế bởi vt.chuvi(). Vì vt.dientich() có

chỉ số là 0 trong danh sách các đối số sau chuỗi định dạng và vt.chuvi() có chỉ số là 1 trong danh sách các đối số sau chuỗi định dạng.

■ Sau đây là những trường hợp định dạng thường dùng trong chuỗi định dạng của phương thức Console.WriteLine()

C (hay c, currency): định dạng tiền, số in ra có dấu \$, số âm để trong ngoặc đơn.

Ví dụ:

Console.WriteLine("{0:C4}", x) cho kết quả \$1.2000 nếu x là 1,2 và (\$1.2000) nếu x có giá trị là -1,2 (số 4 nghĩa là in 4 chữ số lẻ).

E (hay e): định dạng kiểu số mũ

Ví dụ:

Console.WriteLine("{0:E}", 250) cho kết quả là 2.500000E+002

F (hay f): dạng số dấu chấm động

Ví dụ:

Console.WriteLine("{0:F2}", 25) cho kết quả là 25.00

N (hay n): định dạng số nói chung, có dấu ngăn cách hàng nghìn và dấu ngăn cách phần nguyên và phần lẻ (dấu phẩy, dấu chấm)

X (hay x): dạng thập lục phân

Ngoài ra Ký tự 0 (1 chữ số hay là chữ số 0 nếu không có chữ số tại vị trí tương ứng), ký tự # (1 chữ số hay là không có gì nếu không có chữ số tại vị trí tương ứng), dấu - (dấu ngắt khi viết viết số điện thoại), dấu phẩy và dấu chấm.

Ví dụ:

Console.WriteLine("{0(0)(##)###.###}", 564567341) cho kết quả là (0)(56)4-567-341

Ghi chú:

Phương thức string.Format() cũng tương tự như các hàm in, nhưng kết quả được đặt vào biến chuỗi thay vì in ra màn hình.

Ví dụ:

Một lệnh khai báo như sau:

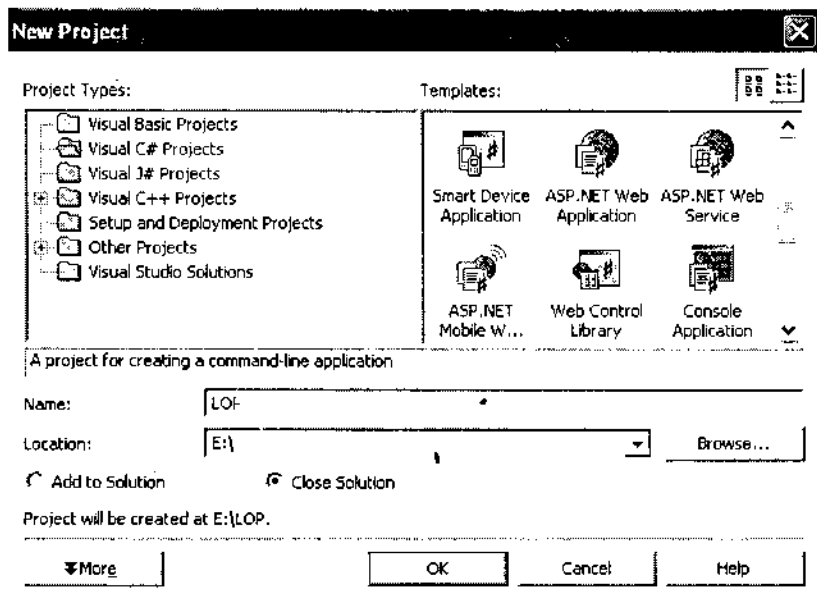
```
String chuoi = string.Format("0x{0:X}", 27);
```

Thì biến chuoi sẽ nhận giá trị là "0x1B".

Để thử chương trình trên, bạn thực hiện như sau:

✱ Vào Microsoft Visual Studio .NET

- * Click chuột vào menu **File**, chọn **New**, chọn **Project**
- * Trong cửa sổ **New Project** hiện ra, bạn chọn:
- * **Visual C# Project** trong khung bên trái có tên **Project Types**.
- * Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.
- * Đặt tên LOP trong khung **Name** như hình sau:



H61

- * Click nút **OK**

Vào cửa sổ viết code, xóa hết các dòng lệnh có sẵn trong chương trình. Viết lại các câu lệnh như sau:

```
using System;
class vongtron
{
    static double PI=3.1416;
    double R;
    public vongtron(double bk)
    {
```

```

        R=bk;
    }
    public double dientich()
    {
        return PI*R*R;
    }
    public double chuvi()
    {
        return 2*PI*R;
    }
};
class MainProgram
{
    static void Main()
    {
        vongtron vt=new vongtron(3);
        Console.WriteLine("Hinh tron :");
        Console.WriteLine(" DT={0},CV={1}",vt.dientich(), vt.chuvi());
        Console.ReadLine();
    }
}

```

Giải thích:

- Câu lệnh using System dùng cho phương thức Console
- Viết một lớp có tên vongtron, với các khai báo và các phương thức bên trong để tính diện tích và chu vi của hình tròn, chúng tôi đã giải thích ở phần trên.
- Viết một lớp chương trình chính để in ra diện tích và chu vi của hình tròn với bán kính R=3.
- Biên dịch và chạy thử chương trình, chúng ta sẽ thấy kết quả như hình sau:

```

E:\VLOP\bin\Debug\VLOP.exe
Hinh tron :
DT=28.2744, CV=18.8496

```

Thử lại bạn thấy:

$$DT = PI * R * R = 3.1416 * 3 * 3 = 28.2744$$

$$CV = 2 * PI * R = 2 * 3.1416 * 3 = 18.8496$$

Bạn vào code của chương trình sửa số 3 trong cấp phát bộ nhớ cho phương thức vongtron(3) phần lớp chính của chương trình, thành số 10 (nghĩa là bán kính bằng 10). Bạn sẽ thấy đoạn code như sau:

```
using System;
class vongtron
{
    static double Pi=3.1416;
    double R;
    public vongtron(double bk)
    {
        R=bk;
    }
    public double dientich()
    {
        return PI*R*R;
    }
    public double chuvi()
    {
        return 2*PI*R;
    }
};
class MainProgram
{
    static void Main()
    {
        vongtron vt=new vongtron(10);
        Console.WriteLine("Hinh tron :");
        Console.WriteLine(" DT={0},CV={1}",vt.dientich(), vt.chuvi());
        Console.ReadLine();
    }
}
```

Chạy thử chương trình, bạn sẽ thấy kết quả như hình sau:

```
E:\LOP\bin\Debug\LOP.exe
```

```
Hình tròn :  
DT=314.16 , CU=62.832
```

H63

Bạn tự kiểm tra lại kết quả khi bán kính $R=10$

Nhận xét:

▪ Nếu viết chương trình như trên, muốn tính diện tích và chu vi của vòng tròn với giá trị bán kính là bao nhiêu thì chúng ta phải vào code của chương trình để sửa lại giá trị của bán kính, vậy rất là bất tiện, chúng ta sẽ viết sao để người sử dụng sẽ nhập giá trị của R khi chạy chương trình. Bạn viết thêm code như sau:

```
using System;  
class vongtron  
{  
    static double PI=3.1416;  
    double R;  
    public vongtron(double bk)  
    {  
        R=bk;  
    }  
    public double dientich()  
    {  
        return PI*R*R;  
    }  
    public double chuvi()  
    {  
        return 2*PI*R;  
    }  
};  
class MainProgram  
{  
    static void Main()  
{
```

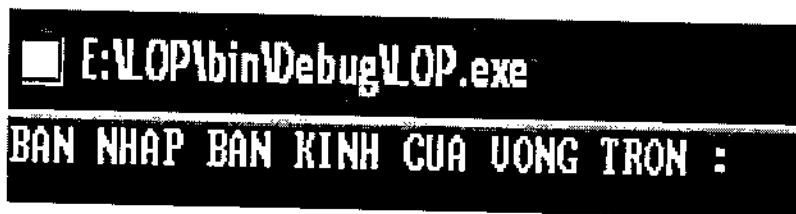
```

float bankinh;
Console.WriteLine("BAN NHAP BAN KINH CUA VONG TRON :");
bankinh=float.Parse(Console.ReadLine());
vongtron vt=new vongtron(bankinh);
Console.WriteLine("Hinh tron :");
Console.WriteLine(" DT={0},CV={1}",vt.dientich(), vt.chuvi());
Console.ReadLine();
}
}

```

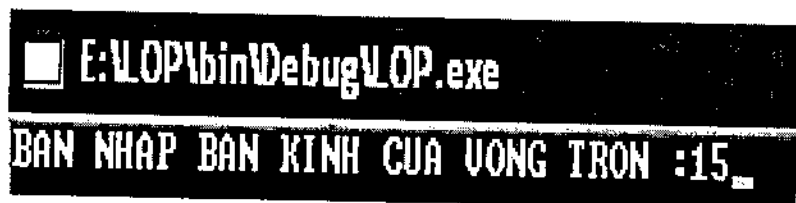
Giải thích:

- Bạn thêm vào 3 dòng lệnh đậm tại vị trí ở trên.
- Xóa số 10 trong phương thức vongtron(10), nhập vào biến bankinh thành vongtron(bankinh).
- Khai báo biến bankinh có kiểu số thực float.
- In ra màn hình câu hỏi "BAN NHAP BAN KINH CUA VONG TRON".
- Nhập vào giá trị của bán kính.
- Biên dịch và chạy thử chương trình, bạn sẽ có thấy như hình sau:



H64

- Giả sử bạn nhập R=15 như hình sau:



H65

Nhấn phím Enter, chúng ta có kết quả

E:\LOP\bin\Debug\LOP.exe

BAN NHAP BAN KINH CUA VONG TRON :15
Hinh tron :
DT=706.86,CU=94.248

H66

- Bạn tự kiểm tra kết quả.

Nếu bạn sửa một chút như sau:

```
using System;
```

```
class vongtron
```

```
{
```

```
    static double PI=3.1416;
```

```
    double R;
```

```
    public vongtron(double bk)
```

```
    {
```

```
        R=bk;
```

```
    }
```

```
    public double dientich()
```

```
    {
```

```
        return PI*R*R;
```

```
    }
```

```
    public double chuvi()
```

```
    {
```

```
        return 2*PI*R;
```

```
    }
```

```
};
```

```
class MainProgram
```

```
{
```

```
    static void Main()
```

```
{
```

```
        float bankinh;
```

```
        Console.Write("BAN NHAP BAN KINH CUA VONG TRON :");
```



```

bankinh=float.Parse(Console.ReadLine());
vongtron vt=new vongtron(bankinh);
Console.WriteLine("Hình tron :");
Console.WriteLine(" Diện tích hình tron={0},Chu vi hình
tron={1}",vt.dientich(), vt.chuvi());
Console.ReadLine();
}
}

```

■ Thay đổi chuỗi thể hiện trong câu lệnh đậm ở trên, bạn sẽ thấy kết quả như hình sau, nếu bạn nhập giá trị bán kính là 12.

```

E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA VONG TRON :12
Hình tron :
Dien tích hình tron=452.3904,Chu vi hình tron=75.3984

```

H67

Để hiểu rõ hơn về lớp, bạn xem ví dụ sau:

Viết một lớp thực hiện tính diện tích và chu vi của một hình chữ nhật.

Phân tích:

Trong hình chữ nhật sẽ có các thuộc tính chiều dài và chiều rộng, có các phương thức (hàm) để tính diện tích và chu vi qua các thuộc tính. Bạn sẽ viết một lớp để tính như sau:

```

class hinhchunhat
{
    double dai, rong;
    public hinhchunhat(double d, double r)
    {
        dai=d;
        rong=r;
    }
}

```

```

public double dientich()
{
    return dai*rong;
}
public double chuvi()
{
    return 2*(dai+rong);
}

```

};

Giải thích:

- Nếu đọc kỹ bài trên, bạn thấy bài này rất là dễ hiểu. Khai báo lớp bằng từ khóa class có tên là hinhchunhat. Khai báo biến dai, rong có kiểu số thực double.

- Viết phương thức hinhchunhat với hai đối có kiểu thực là d và r (trong vongtron thì chỉ có một đối đó là bán kính)

- Gán giá trị của hai đối cho các biến tương ứng.

- Viết phương thức tính diện tích

- Viết phương thức tính chu vi

- Để hoàn thành chương trình, bạn viết code như sau:

```

using System;
class hinhchunhat
{
    double dai, rong;
    public hinhchunhat(double d, double r)
    {
        dai=d;
        rong=r;
    }
    public double dientich()
    {
        return dai*rong;
    }
    public double chuvi()
    {

```

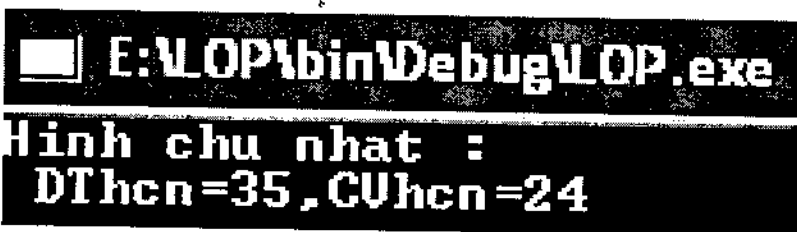
```

        return 2*(dai+rong);
    }
};
class MainProgram
{
    static void Main()
    {
        hinhchunhat hcn=new hinhchunhat(7,5);
        Console.WriteLine("Hinh chu nhat :");
        Console.WriteLine(" DThcn={0},CVhcn={1}",hcn.dientich(),
            hcn.chuvi());
        Console.ReadLine();
    }
}

```

Giải thích:

- Trong lớp chính chúng ta khai báo biến hcn có kiểu hinhchunhat và cấp phát bộ nhớ cho nó với chiều dài bằng 7cm và chiều rộng bằng 5cm. Hiện ra kết quả diện tích và chu vi của hình chữ nhật với chiều dài và rộng như trên.
- Biên dịch và chạy thử chương trình, bạn thấy kết quả như hình sau:



H68

Thử lại kết quả:

$$DThcn = dai * rong = 7 * 5 = 35$$

$$CVhcn = 2 (dai +rong) = 2(7+5) = 24$$

Bạn thấy kết quả trên không có đơn vị sau giá trị của diện tích và chu vi. Để xuất hiện, bạn định dạng trong code như sau:

```

using System;
class hinhchunhat

```

```

{
    double dai, rong;
    public hinhchunhat(double d, double r)
    {
        dai=d;
        rong=r;
    }
    public double dientich()
    {
        return dai*rong;
    }
    public double chuvi()
    {
        return 2*(dai+rong);
    }
};
class MainProgram
{
    static void Main()
    {
        hinhchunhat hcn=new hinhchunhat(7.5);
        Console.WriteLine("Hinh chu nhat :");
        Console.WriteLine(" DThcn={0:#cm2},CVhcn={1:#cm}",
            hcn.dientich(), hcn.chuvi());
        Console.ReadLine();
    }
}

```

Giải thích:

- Bạn sửa lại chuỗi định dạng trong câu lệnh tô đậm ở trên. Dấu # thay thế cho số, cm^2 , cm điền đơn vị sau số.
- Biên dịch và chạy thử chương trình, chúng ta sẽ thấy kết quả như hình sau:

E:\LOP\bin\Debug\LOP.exe

Hình chữ nhật :
DT_{hcn}=35cm², CV_{hcn}=24cm

H69

Để nhập vào giá trị của chiều dài và chiều rộng bất kỳ khi chạy chương trình, bạn thêm vào đoạn code như sau:

```
using System;
class hìnhchữnhật
{
    double dài, rộng;
    public hìnhchữnhật(double d, double r)
    {
        dài=d;
        rộng=r;
    }
    public double diện tích()
    {
        return dài*rộng;
    }
    public double chu vi()
    {
        return 2*(dài+rộng);
    }
};
class MainProgram
{
    static void Main()
    {
        float chieudài, chieurộng;
        Console.Write("BAN NHAP CHIEU DAI :");
        chieudài=float.Parse(Console.ReadLine());
```

```

Console.Write("BAN NHAP CHIEU RONG :");
chieurong=float.Parse(Console.ReadLine());
hinhchunhat hcn=new hinhchunhat(chieudai,chieurong);
Console.WriteLine("Hinh chu nhhat :");
Console.WriteLine(" DThcn={0:#cm2},CVhcn={1:#cm}",
    hcn.dientich(), hcn.chuvi());
Console.ReadLine();
}
)

```

Giải thích:

- Bạn nhập và sửa lại các dòng lệnh đậm ở trên.
- Khai báo hai biến chieudai và chieurong có kiểu thực
- Nhập vào chiều dài, chiều rộng
- Thay hai đối của phương thức hinhchunhat là chieudai và chieurong, hinhchunhat(chieudai, chieurong)
 - Xuất ra màn hình diện tích và chu vi của hình chữ nhật.
 - Biên dịch và chạy thử chương trình, giả sử chúng ta nhập chiều dài là 10 và chiều rộng bằng 8, ta có kết quả như hình sau:

```

E:\LOP\bin\Debug\LOP.exe
BAN NHAP CHIEU DAI :10
BAN NHAP CHIEU RONG :8
Hinh chu nhhat :
DThcn=80cm2, CVhcn=36cm

```

H70

- Chạy lại chương trình, giả sử chúng ta nhập sai (chiều dài có giá trị nhỏ hơn chiều rộng) như sau:
 - Chiều dài bằng 10, chiều rộng bằng 15. Bạn sẽ thấy:

```

E:\LOP\bin\Debug\LOP.exe
BAN NHAP CHIEU DAI :15
BAN NHAP CHIEU RONG :10
Hinh chu nhhat :
DThcn=150cm2, CVhcn=50cm

```

H71

Chương trình vẫn tính, vì trong phương thức hìnhchunhat có hai đối là hai số thực, nhận được giá trị là nó sẽ thực hiện, không xét điều kiện chiều dài phải lớn hơn chiều rộng.

Để chương trình không thực hiện khi người sử dụng nhập chiều dài có giá trị nhỏ hơn chiều rộng, có nhiều cách, giả sử chúng ta thực hiện một cách là dùng lệnh goto như sau:

```
using System;
class hìnhchunhat
{
    double dai, rong;
    public hìnhchunhat(double d, double r)
    {
        dai=d;
        rong=r;
    }
    public double dientich()
    {
        return dai*rong;
    }
    public double chuvi()
    {
        return 2*(dai+rong);
    }
};
class MainProgram
{
    static void Main()
    {
        char hih;
        float chieudai, chieurong;
        Console.Write("BAN NHAP CHIEU DAI :");
        chieudai=float.Parse(Console.ReadLine());
        Console.Write("BAN NHAP CHIEU RONG :");
        chieurong=float.Parse(Console.ReadLine());
        if (chieurong>=chieudai)
```

```
goto hihi;
```

```
hinhchunhat hcn=new hinhchunhat(chieudai,chieurong);
```

```
Console.WriteLine("Hinh chu nhhat :");
```

```
Console.WriteLine(" DThcn={0:#cm2},CVhcn={1:#cm}",  
hcn.dientich(), hcn.chuvi());
```

```
hihi::
```

```
if (chieurong>=chieudai)
```

```
Console.WriteLine("BAN DA NHAP SAI, CHIEU DAI PHAI  
LON HON CHIEU RONG");
```

```
Console.ReadLine();
```

```
}
```

```
}
```

Giải thích:

- Khai báo biến hihi có kiểu char
- Thêm phát biểu if với điều kiện khi chiều rộng lớn hơn hoặc bằng chiều dài thì sẽ thực hiện lệnh goto hihi để nhảy đến nhãn có tên hihi để bỏ qua các câu lệnh tính toán, và báo ra lỗi sai cho người sử dụng biết.

Qua hai bài tập trên, các bạn đã hiểu rõ khá nhiều về lớp. Bây giờ các bạn có thể gộp hai lớp trên vào chung một chương trình. Người sử dụng sẽ nhập bán kính của vòng tròn để tính diện tích và chu vi. Sau đó nhập tiếp chiều dài và chiều rộng để tính diện tích và chu vi của hình chữ nhật. Bạn viết code lại như sau:

```
using System;
```

```
class vongtron
```

```
{
```

```
static double PI=3.1416;
```

```
double R;
```

```
public vongtron(double bk)
```

```
{
```

```
    R=bk;
```

```
}
```

```
public double dientich()
```

```
{
```

```
    return PI*R*R;
```



```

    }
    public double chuvi()
    {
        return 2*PI*R;
    }
};
class hinhchunhat
{
    double dai, rong;
    public hinhchunhat(double d, double r)
    {
        dai=d;
        rong=r;
    }
    public double dientich()
    {
        return dai*rong;
    }
    public double chuvi()
    {
        return 2*(dai+rong);
    }
};
class MainProgram
{
    static void Main()
    {
        float bankinh;
        Console.Write("BAN NHAP BAN KINH CUA VONG TRON :");
        bankinh=float.Parse(Console.ReadLine());
        vongtron vt=new vongtron(bankinh);
        Console.WriteLine("Hinh tron :");
        Console.WriteLine(" Diên tích hình tron={0},Chu vi hình
            tron={1}",vt.dientich(), vt.chuvi());
        Console.ReadLine();
    }
}

```

```

char hihi;
float chieudai, chieurong;
Console.WriteLine("BAN NHAP CHIEU DAI :");
chieudai=float.Parse(Console.ReadLine());
Console.WriteLine("BAN NHAP CHIEU RONG :");
chieurong=float.Parse(Console.ReadLine());
if (chieurong>=chieudai)
    goto hihi;
hinhchunhat hcn=new hinhchunhat(chieudai,chieurong);
Console.WriteLine("Hinh chu nhât :");
Console.WriteLine(" DThcn={0:#cm2},CVhcn={1:#cm}",
    hcn.dientich(), hcn.chuvi());
hihi;
if (chieurong<chieudai)
    Console.WriteLine("BAN DA NHAP SAI, CHIEU DAI PHAI
        LON HON CHIEU RONG");
Console.ReadLine();
}
}

```

Giải thích:

- Code của chương trình không có gì là thay đổi, chúng ta chỉ sắp xếp lại: trước tiên là lớp vongtron, sau đó là lớp hinhchunhat, sau đó là lớp chính cho chúng ta nhập vào giá trị bán kính và xuất hiện kết quả diện tích và chu vi hình tròn, sau đó nhập kết quả chiều dài, chiều rộng của hình chữ nhật, sẽ xuất diện tích và chu vi của hình chữ nhật.

- Biên dịch và chạy thử chương trình, bạn thấy tuần tự thực hiện của chương trình như sau:

 E:\LOP\bin\Debug\LOP.exe

BAN NHAP BAN KINH CUA VONG TRON :

- Bạn nhập vào giá trị của bán kính. Giả sử nhập 7, gõ Enter ta được

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA VONG TRON :7
Hinh tron :
Dien tich hinh tron=153.9384,Chu vi hinh tron=43.9824
```

H73

Ghi chú:

- Nhờ có lệnh `Console.ReadLine()` phần này (lệnh đậm), chương trình sẽ dừng lại. Gõ phím Enter, bạn thấy hình như sau:

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA VONG TRON :7
Hinh tron :
Dien tich hinh tron=153.9384,Chu vi hinh tron=43.9824
BAN NHAP CHIEU DAI :_
```

H74

Giả sử bạn nhập chiều dài là 10, gõ Enter

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA VONG TRON :7
Hinh tron :
Dien tich hinh tron=153.9384,Chu vi hinh tron=43.9824
BAN NHAP CHIEU DAI :10
BAN NHAP CHIEU RONG :
```

H75

■ Giả sử bạn nhập chiều rộng là 8. Gõ Enter bạn có kết quả như sau:

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA UONG TRON :7
Hinh tron :
  Dien tích hình tron=153.9384,Chu vi hình tron=43.9824

BAN NHAP CHIEU DAI :10
BAN NHAP CHIEU RONG :8
Hinh chu nhật :
  DThcn=80cm2, CVhcn=36cm
```

H76

Nếu nhập sai về kích thước của hình chữ nhật, tức chiều dài nhỏ hơn hoặc bằng chiều rộng, bạn sẽ thấy kết quả tính toán của hình chữ nhật không thực hiện và báo ra lỗi sai. Bạn tham khảo hai hình sau:

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA UONG TRON :4
Hinh tron :
  Dien tích hình tron=50.2656,Chu vi hình tron=25.1328

BAN NHAP CHIEU DAI :4
BAN NHAP CHIEU RONG :4
BAN DA NHAP SAI, CHIEU DAI PHAI LON HON CHIEU RONG
```

H77 Khi chiều dài bằng chiều rộng

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA UONG TRON :4
Hinh tron :
  Dien tích hình tron=50.2656,Chu vi hình tron=25.1328

BAN NHAP CHIEU DAI :6
BAN NHAP CHIEU RONG :9
BAN DA NHAP SAI, CHIEU DAI PHAI LON HON CHIEU RONG
```

H78 Khi chiều rộng lớn hơn chiều dài

Bạn thử làm thêm một bài tập nữa về lớp như sau:

Viết chương trình nhập vào hai số a và b. Tính a+b, a-b, a*b, a/b và a%b dùng lớp.

* Vào Microsoft Visual Studio .NET

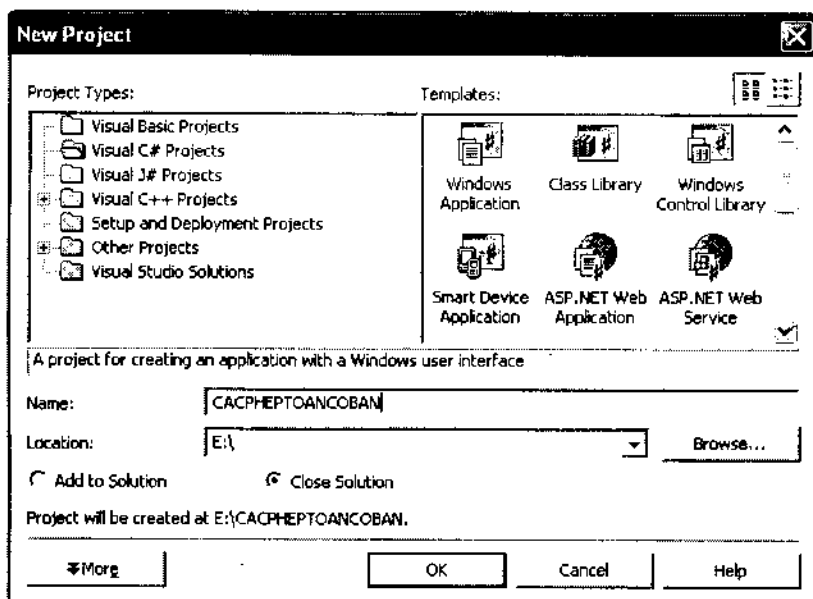
* Click chuột vào menu **File**, chọn **New**, chọn **Project**

* Trong cửa sổ **New Project** hiện ra, bạn chọn:

* **Visual C# Project** trong khung bên trái có tên **Project Types**.

* Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.

* Đặt tên **CACPHEPTOANCOBAN** trong khung **Name** như hình sau:



H79

* Click nút **OK**

Vào cửa sổ viết code, xóa hết các dòng lệnh có sẵn, viết code như sau:

```
using System;
```

```
class tinhtoan
```

```
{
```

```
    double soa, sob;
```

```

public tinhtoan(double a, double b)
{
    soa=a;
    sob=b;
}
public double cong()
{
    return soa+sob;
}
public double tru()
{
    return soa-sob;
}
public double nhan()
{
    return soa*sob;
}
public double chia()
{
    return soa/sob;
}
public double sodu()
{
    return soa%sob;
}
};

class MainProgram
{
    static void Main()
    {
        double x, y;
        Console.Write("BAN NHAP SO a :");
        x=double.Parse(Console.ReadLine());
        Console.Write("BAN NHAP SO b :");
    }
}

```

```

y=double.Parse(Console.ReadLine());
tinhtoan tt=new tinhtoan(x,y);
Console.WriteLine("KET QUA TINH TOAN");
Console.WriteLine(" a+b={0},a-b={1},a*b={2}, a/b={3},a mod
    b={4} ",tt.cong(), tt.tru(), tt.nhan(), tt.chia(),tt.sodu());
Console.ReadLine();
}
}

```

Giải thích:

- Thiết lập một lớp có tên tinhtoan
- Để tính toán $a+b$, $a-b$, $a*b$, a/b và $a\%b$ ta thấy phải có hai số a và b thì mới tính được, vậy bạn phải khai báo hai biến này, ở đây chúng tôi báo hai biến có kiểu số thực.
 - Khai báo một phương thức có tên trùng với tên của lớp đó là tinhtoan, có hai đối là số thực. Gán biến soa cho đối thứ nhất, sob cho đối thứ hai. Phương thức này là public để có thể gọi được ở ngoài lớp. Phương thức này dùng để khởi tạo dữ liệu (constructor).
 - Khai báo phương thức cong có kiểu thực dạng **public** để có thể gọi ở ngoài lớp, với trị trả về là $a + b$.
 - Khai báo phương thức tru có kiểu thực dạng public để có thể gọi ở ngoài lớp, với trị trả về là $a - b$.
 - Khai báo phương thức nhan có kiểu thực dạng public để có thể gọi ở ngoài lớp, với trị trả về là $a * b$.
 - Khai báo phương thức chia có kiểu thực dạng public để có thể gọi ở ngoài lớp, với trị trả về là a / b .
 - Khai báo phương thức sodu có kiểu thực dạng public để có thể gọi ở ngoài lớp, với trị trả về là $a \% b$.
 - Thiết lập một lớp chính của chương trình, khai báo hai biến x, y có kiểu thực.
 - Nhập hai số thông qua hai biến này.
 - Khai báo biến tt có kiểu tinhtoan và cấp phát bộ nhớ cho biến này với 2 đối là hai số vừa nhập.
 - Xác lập các giá trị trong chuỗi của phương thức Write là: {0}, {1}, {2}, {3}, {4} tương ứng với các biến là các phương thức cong, tru, nhan, chia và sodu là con của lớp tinhtoan thông qua tên biến tt.

▪ Biên dịch và chạy thử chương trình. Giả sử bạn nhập số a là 10, số b là 5. Gõ phím Enter, bạn thấy kết quả như hình sau:

```
E:\LOP\bin\Debug\LOP.exe
BAN NHAP SO a :10
BAN NHAP SO b :5
KET QUA TINH TOAN
a+b=15,a-b=5,a*b=50, a/b=2,a mod b=0
```

H80

Bạn chạy lại chương trình, nhập giá trị a và b bất kỳ thử kiểm tra kết quả xem có đúng hay không.

BÀI 10

Kiểu dữ liệu của người lập trình

Ngoài những kiểu dữ liệu đơn có sẵn, người lập trình có thể định nghĩa thêm kiểu dữ liệu mới cùng với các phép toán trên kiểu dữ liệu này. Chẳng hạn, để tạo kiểu dữ liệu tinhtoaan cho hai số qua các phép toán cơ bản cộng, trừ, nhân, chia, tính số dư và in kết quả ra màn hình. Bạn có thể dùng kiểu cấu trúc struct

```
struct tenkieudulieu
```

```
{  
    .....  
    .....  
    .....  
};
```

Bạn xem đoạn chương trình sau:

```
using System;
```

```
struct tinhtoaan
```

```
{  
    double soa, sob;  
    public tinhtoaan(double a, double b)  
    {  
        soa=a;  
        sob=b;  
    }  
    public double cong()  
    {  
        return soa+sob;  
    }  
    public double tru()  
    {  
        return soa-sob;  
    }  
}
```

```

public double nhan()
{
    return soa*sob;
}
public double chia()
{
    return soa/sob;
}
public double sodu()
{
    return soa%sob;
}
};

class MainProgram
{
    static void Main()
    {
        double x, y;
        Console.Write("BAN NHAP SO a :");
        x=double.Parse(Console.ReadLine());
        Console.Write("BAN NHAP SO b :");
        y=double.Parse(Console.ReadLine());
        tinhtoan tt=new tinhtoan(x,y);
        Console.WriteLine("KET QUA TINH TOAN");
        Console.WriteLine(" a+b={0},a-b={1},a*b={2}, a/b={3},a mod
            b={4} ",tt.cong(), tt.tru(), tt.nhan(), tt.chia(),tt.sodu());
        Console.ReadLine();
    }
}

```

■ Bạn thấy thay vì thiết lập một lớp với từ khóa class, bạn có thể dùng cấu trúc struct. Đối chiếu với chương trình tính toán các phép toán đơn giản dùng lớp ở bài trước, bạn thấy không có gì thay đổi nhiều, chỉ có một thay đổi nhỏ duy nhất đó là từ khóa class được thay thế bằng từ khóa struct.

- Bạn vào lại chương trình cũ, sửa **class** thành **struct** và chạy thử chương trình, nhập vào hai số a và b. Giả sử chúng ta nhập a=34, b=3. Gõ phím Enter, bạn thấy kết quả như hình sau:

```

E:\LOP\bin\Debug\LOP.exe
BAN NHAP SO a :34
BAN NHAP SO b :3
KET QUẢ TINH TOAN
a+b=37,a-b=31,a*b=102, a/h=11.33333333333333,a mod b=1

```

H81

- Tương tự như vậy, bạn có thể vào lại project để tính diện tích và chu vi của vòng tròn và hình chữ nhật. Thay đổi từ khóa class của hai lớp vongtron và hinhchunhat thành từ khóa struct. Bạn thấy chương trình cũng thực hiện giống như khi dùng class. Vậy trong C# từ khóa class và từ khóa struct là tương đương với nhau (trừ một vài trường hợp đặc biệt, chúng tôi sẽ đề cập sau). Bạn xem lại đoạn code của chương trình trên.

```

using System;
struct vongtron
{
    static double PI=3.1416;
    double R;
    public vongtron(double bk)
    {
        R=bk;
    }
    public double dientich()
    {
        return PI*R*R;
    }
}

```

```

public double chuvi()
{
    return 2*PI*R;
}
};
struct hinhchunhat
{
    double dai, rong;
    public hinhchunhat(double d, double r)
    {
        dai=d;
        rong=r;
    }
    public double dientich()
    {
        return dai*rong;
    }
    public double chuvi()
    {
        return 2*(dai+rong);
    }
};
class MainProgram
{
    static void Main()
    {
        float bankinh;
        Console.WriteLine("BAN NHAP BAN KINH CUA VONG TRON :");
        bankinh=float.Parse(Console.ReadLine());
        vongtron vt=new vongtron(bankinh);
        Console.WriteLine("Hinh tron :");
        Console.WriteLine(" Diên tích hình tron={0},Chu vi hình
            tron={1}",vt.dientich(), vt.chuvi());
        Console.ReadLine();
        char hihi;
    }
}

```

```

float chieudai, chieurong;
Console.Write("BAN NHAP CHIEU DAI :");
chieudai=float.Parse(Console.ReadLine());
Console.Write("BAN NHAP CHIEU RONG :");
chieurong=float.Parse(Console.ReadLine());
if (chieurong>=chieudai)
    goto hihi;
hinhchunhat hcn=new hinhchunhat(chieudai,chieurong);
Console.WriteLine("Hinh chu nhât:");
Console.WriteLine(" DThcn={0:#cm2},CVhcn={1:#cm}",
    hcn.dientich(), hcn.chuvi());

hihi;
if (chieurong>=chieudai)
    Console.WriteLine("BAN DA NHAP SAI, CHIEU DAI PHAI
        LON HON CHIEU RONG");

Console.ReadLine();
}
}

```

- Biên dịch và chạy thử chương trình, bạn nhập bán kính R, bạn sẽ thấy kết quả diện tích và chu vi của vòng tròn ứng với giá trị của bán kính vừa nhập. Gõ phím Enter, nhập chiều dài, chiều rộng của hình chữ nhật, bạn sẽ thấy kết quả diện tích và chu vi của hình chữ nhật ứng với các giá trị vừa nhập.
- Bạn thử xem kết quả khi chúng tôi nhập R=20, chiều dài = 15 và chiều rộng = 10.

```

E:\LOP\bin\Debug\LOP.exe
BAN NHAP BAN KINH CUA VONG TRON :20
Hinh tron :
Dien tích hình tron=1256.64,Chu vi hình tron=125.664

BAN NHAP CHIEU DAI :15
BAN NHAP CHIEU RONG :10
Hinh chu nhât :
DThcn=150cm2,CVhcn=50cm

```

BÀI 11

CÁC PHƯƠNG THỨC TRÙNG TÊN (Chồng phương thức)

Một ưu điểm của các ngôn ngữ lập trình hướng đối tượng là việc cho phép các phương thức có trùng tên và phân biệt nhau nhờ vào số lượng các tham số, kiểu của tham số và giá trị trả về của phương thức. Khái niệm này được gọi là **method overloading** (chồng phương thức). Cơ chế này cho phép khai thác các phương thức một cách linh động hơn. Bạn xem lại chương trình dưới đây, chúng tôi lấy lại bài tạo lớp tinhtoan để tính các phép toán cơ bản, và thay đổi lại một chút.

```
using System;
class tinhtoan
{
    double soa, sob, soc;
    public tinhtoan(double a, double b)
    {
        soa=a;
        sob=b;
    }
    public tinhtoan(double a, double b, double c)
    {
        soa=a;
        sob=b;
        soc=c;
    }
    public double conghai()
    {
        return soa+sob;
    }
}
```

```
public double truhai()
{
    return soa-sob;
}
public double nhanhai()
{
    return soa*sob;
}
public double chiahai()
{
    return soa/sob;
}
public double sodu()
{
    return soa%sob;
}
public double congba()
{
    return soa+sob+soc;
}
public double truba()
{
    return soa-sob-soc;
}
public double nhanba()
{
    return soa*sob*soc;
}
public double chiaba()
{
    return soa/sob/soc;
}
};

class MainProgram
```

```

static void Main()
{
    double x, y, z;
    Console.Write("BAN NHAP SO a :");
    x=double.Parse(Console.ReadLine());
    Console.Write("BAN NHAP SO.b :");
    y=double.Parse(Console.ReadLine());
    Console.Write("BAN NHAP SO c :");
    z=double.Parse(Console.ReadLine());
    tinhtoan tt2=new tinhtoan(x,y);
    Console.WriteLine("KET QUA TINH TOAN TREN HAI SO a,b");
    Console.WriteLine(" a+b={0},a-b={1},a*b={2}, a/b={3},a mod
        b={4} ",tt2.conghai(), tt2.truhai(), tt2.nhanhai(),
        tt2.chiahai(),tt2.sodu());
    tinhtoan tt3=new tinhtoan(x,y,z);
    Console.WriteLine("KET QUA TINH TOAN TREN BA SO a,b,c");
    Console.WriteLine(" a+b+c={0},a-b-c={1},a*b*c={2},
        a/b/c={3}",tt3.congba(), tt3.truba(), tt3.nhanba(),
        tt3.chiaba());
    Console.ReadLine();
}
}

```

Giải thích:

▪ Bài này cũng giống như bài tập dùng lớp ở trên. Một cái đặc biệt là bạn thấy chúng tôi khai báo hai phương thức có cùng tên tinhtoan, nhưng hai phương thức này có số tham số khác nhau, phương thức tinhtoan thứ nhất có hai tham số, phương thức tinhtoan thứ hai có ba tham số. Điều này cho phép được trùng tên với nhau.

- Thiết lập một lớp có tên là tinhtoan
- Khai báo ba biến có kiểu số thực (double)
- Thiết lập phương thức có tên tinhtoan có hai đối, đây là phương thức khởi tạo dữ liệu
- Thiết lập phương thức có tên tinhtoan có ba đối, đây là phương thức khởi tạo dữ liệu

- Thiết lập các phương thức conghai, truhai, nhanhai, chiahai, sodu để thực hiện các phép toán tương ứng trên hai số.
- Thiết lập các phương thức congba, truba, nhanba, chiaaba để thực hiện các phép toán tương ứng trên ba số.
- Tạo một lớp chính để thể hiện việc tính toán ra màn hình.
- Khai báo ba biến là số thực để nhận ba số nhập vào
- Khai báo biến tt2 có kiểu tinhtoan có hai đối và cấp phát bộ nhớ cho nó.
 - Xuất kết quả các phép toán ra màn hình
 - Khai báo biến tt3 có kiểu tinhtoan có ba đối và cấp phát bộ nhớ cho nó.
 - Xuất kết quả tính toán ra màn hình.
 - Biên dịch và chạy thử chương trình, giả sử nhập $a=12$, $b=4$, $c=2$.
Gõ phím Enter, bạn thấy kết quả như hình sau:

```

E:\LOP\bin\Debug\LOP.exe
BAN NHAP SO a :12
BAN NHAP SO b :4
BAN NHAP SO c :2
KET QUA TINH TOAN TREN HAI SO a,b
a+b=16,a-b=8,a*b=48, a/b=3,a mod b=0
KET QUA TINH TOAN TREN BA SO a,b,c
a+b+c=18,a-b-c=6,a*b*c=96, a/b/c=1.5

```

H83

BÀI 12

TRUYỀN THAM SỐ

Một cải tiến đáng chú ý của C# so với các ngôn ngữ lập trình khác là việc truyền tham số của các phương thức được quy định rất chặt chẽ, tạo điều kiện để giảm bớt các lỗi lập trình xảy ra do việc truyền tham số. Tuy nhiên, người lập trình cũng cần phải nắm vững các quy định này thì mới vận dụng có hiệu quả.

Tham số giá trị

Giá trị của tham số được truyền cho hàm gọi, các thao tác trên tham số hình thức sẽ không ảnh hưởng đối với biến truyền đến.

Ví dụ:

Bạn xem đoạn chương trình sau:

```
using System;
struct sophuc
{
    public double thuc, ao;
    public sophuc(double a, double b)
    {
        thuc=a;
        ao=b;
    }
};
class MainProgram
{
    static void tang(int i)
    {
        i++;
    }
}
```

```

static void giam(int j)
{
    j--;
}
static void tang(sophuc c)
{
    c.thuc++;
}
static void giam(sophuc d)
{
    d.thuc--;
}
static void addString(string chuoi)
{
    chuoi=chuoi+" are you";
}
static void Main()
{
    int n=25;
    sophuc sp =new sophuc(4,9);
    string ch="How";
    tang(n);
    Console.WriteLine("1/ gia tri cua n la: 0",n);
    giam(n);
    Console.WriteLine("2/ gia tri cua n la: 0",n);
    tang(sp);
    Console.WriteLine("3/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
    giam(sp);
    Console.WriteLine("4/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
    addString(ch);
    Console.WriteLine("5/ {0}",ch);
    Console.ReadLine();
}
};

```

Giải thích:

- Khai báo một cấu trúc có tên sophuc.
 - Khai báo hai biến thực và ao có kiểu số thực double, dạng public để có thể gọi khi ở ngoài cấu trúc.
 - Thiết lập một phương thức khởi tạo có tên sophuc (cùng tên với cấu trúc). Phương thức này có hai đối a, b kiểu số thực.
 - Khởi tạo biến thực và biến ao-
 - Tạo một hàm tang có một đối i kiểu số nguyên
 - Tăng biến i lên 1
 - Tạo một hàm giam có một đối j kiểu số nguyên
 - Giảm biến j xuống 1
 - Tạo một hàm tang có một đối c kiểu sophuc
 - Tăng biến thực của số phức lên 1
 - Tạo một hàm giam có một đối d kiểu sophuc
 - Giảm biến thực của số phức xuống 1
 - Tạo một hàm addString có một đối chuỗi kiểu string
 - Gán chuỗi bằng chuỗi + " are you"
 - Vào hàm chính
 - Khai báo biến n có kiểu int và gán giá trị 25 cho nó
 - Khai báo biến sp có kiểu sophuc có giá trị thực là 4, ảo là 9 và cấp phát bộ nhớ cho nó
 - Khai báo biến ch có kiểu string và gán chuỗi "How" cho nó
 - Gọi hàm tang với đối là n (n=25)
 - In giá trị của biến n sau khi thực hiện hàm tang
 - Gọi hàm giam với đối là n (n=25)
 - In giá trị của biến n sau khi thực hiện hàm giam
 - Gọi hàm tang với đối là sp
 - In giá trị của số phức sau khi thực hiện hàm tang
 - Gọi hàm giam với đối là sp
 - In giá trị của số phức sau khi thực hiện hàm giam
 - Gọi hàm nối chuỗi addString
 - In ra chuỗi sau khi thực hiện hàm nối chuỗi addString
- Biên dịch và chạy thử chương trình, bạn thấy kết quả như hình sau:

```

1/ gia tri cua n la: 25
2/ gia tri cua n la: 25
3/ So phuc: 4+9i
4/ So phuc: 4+9i
5/ How

```

H84

Bạn thấy kết quả vẫn giữ nguyên giá trị cũ. $n=25$, thành phần thực của số phức là 4, ảo là 9 (đây là các giá trị được gán trong chương trình khi khai báo biến).

Trường hợp đặc biệt

Khi tham số là **đối tượng** thì tình huống xảy ra không giống như kiểu **struct** bởi vì các đối tượng của **C#** là kiểu dữ liệu **tham chiếu**.

Vẫn lấy ví dụ trên nhưng thay từ khóa **struct** bằng từ khóa **class** như sau (từ tô đậm):

```

using System;
class sophuc
{
    public double thuc, ao;
    public sophuc(double a, double b)
    {
        thuc=a;
        ao=b;
    }
};
class MainProgram
{
    static void tang(int i)
    {
        i++;
    }
    static void giam(int j)
    {
        j--;
    }
}

```

```

static void tang(sophuc c)
{
    c.thuc++;
}
static void giam(sophuc d)
{
    d.thuc--;
}
static void addString(string chuoai)
{
    chuoai=chuoai+" are you";
}
static void Main()
{
    int n=25;
    sophuc sp =new sophuc(4,9);
    string ch="How";
    tang(n);
    Console.WriteLine("1/ gia tri cua n la: {0}",n);
    giam(n);
    Console.WriteLine("2/ gia tri cua n la: {0}",n);
    tang(sp);
    Console.WriteLine("3/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
    giam(sp);
    Console.WriteLine("4/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
    addString(ch);
    Console.WriteLine("5/ {0}",ch);
    Console.ReadLine();
}
};

```

Chạy thử chương trình, bạn thấy kết quả như hình sau:

```

1/ gia tri cua n la: 25
2/ gia tri cua n la: 25
3/ So phuc: 5+9i
4/ So phuc: 4+9i
5/ How

```

H85

Hàm tang và giam viết cho kiểu sophuc trong (class) và (struct) là giống nhau, nhưng kết quả in ra cho trường hợp này là phần thực của số phức tăng lên 1 đơn vị (từ 4 lên 5) khi thực hiện hàm tang(sp). Sau đó sẽ giảm 1 đơn vị khi gặp hàm giam(sp) (từ 5 xuống 4). Số n và biến chuỗi không phải là đối tượng nên không thay đổi.

Nếu bạn sửa lại code của chương trình từ giam(sp) thành tang(sp) như sau:

```

using System;
class sophuc
{
    public double thuc, ao;
    public sophuc(double a, double b)
    {
        thuc=a;
        ao=b;
    }
};
class MainProgram
{
    static void tang(int i)
    {
        i++;
    }
    static void giam(int j)
    {
        j--;
    }
    static void tang(sophuc c)
    {
        c.thuc++;
    }
}

```

```

}
static void giam(sophuc d)
{
    d.thuc--;
}
static void addString(string chuoai)
{
    chuoai=chuoai+" are you";
}
static void Main()
{
    int n=25;
    sophuc sp =new sophuc(4,9);
    string ch="How";
    tang(n);
    Console.WriteLine("1/ gia tri cua n la: {0}",n);
    giam(n);
    Console.WriteLine("2/ gia tri cua n la: {0}",n);
    tang(sp);
    Console.WriteLine("3/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
    tang(sp);
    Console.WriteLine("4/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
    addString(ch);
    Console.WriteLine("5/ {0}",ch);
    Console.ReadLine();
}
};

```

Chạy thử chương trình, bạn sẽ thấy kết quả như sau:

```

E:\TRUYEN_THAM_SO\bin\Debug\TRUYEN_THAM_SO.exe
1/ gia tri cua n la: 25
2/ gia tri cua n la: 25
3/ So phuc: 5+9i
4/ So phuc: 6+9i
5/ How

```


Hàm tang(sp) thứ nhất tăng phần thực của số phức lên 1 (4 lên 5), hàm tang(sp) thứ hai sẽ tăng phần thực lên 1 đơn vị nữa (5 lên 6).

Bạn xem thêm một ví dụ nữa qua đoạn code sau:

```
using System;
struct phanso
{
    public int tuso, mauso;
    public phanso(int tu, int mau)
    {
        tuso=tu;
        mauso=mau;
    }
};
class MainProgram
{
    static void tang(phanso ps)
    {
        ps.tuso+=ps.mauso;
    }
    static void Main()
    {
        phanso huuti=new phanso(3,11);
        tang(huuti);
        Console.WriteLine("{0}/{1}",huuti.tuso,huuti.mauso);
        Console.ReadLine();
    }
};
```

Giải thích:

- Khai báo cấu trúc struct có tên phanso
- Khai báo hai biến tuso và mauso có kiểu nguyên int dạng public
- Thiết lập phương thức phanso có hai đối là số nguyên (phương thức khởi tạo dữ liệu)
- Khai báo hàm tang có một đối ps có kiểu phanso để tăng phần số (phần số + 1)

▪ Khai báo biến huuti có kiểu phanso có tử số là 3, mẫu số là 11 và cấp phát bộ nhớ cho nó.

- Thực hiện hàm tang(huuti)
- In ra màn hình phân số
- Chạy thử chương trình, bạn thấy kết quả như sau:

```
E:\TRUYEN_THAM_SO\bin\Debug\TRUYEN_THAM_SO.exe
```

```
3/11
```

H87

Kết quả in ra trên màn hình phân số 3/11 (giống như giá trị khai báo)

Sửa lại đoạn code trên như sau:

```
using System;
class phanso
{
    public int tuso, mauso;
    public phanso(int tu, int mau)
    {
        tuso=tu;
        mauso=mau;
    }
};
class MainProgram
{
    static void tang(phanso ps)
    {
        ps.tuso+=ps.mauso;
    }
    static void Main()
    {
        phanso huuti=new phanso(3,11);
        tang(huuti);
        Console.WriteLine("{0}/{1}",huuti.tuso,huuti.mauso);
    }
}
```

```
Console.ReadLine();
```

- Chạy lại chương trình, bạn thấy kết quả như sau:

```
E:\TRUYEN_THAM_SO\bin\Debug\TRUYEN_THAM_SO.exe
```

```
14/11
```

H88

Kết quả in ra là 14/11 tức là bằng phân số khai báo (3//11) cộng với 1 (3/11+11/11=14/11)

Bạn suy nghĩ kết quả của hai trường hợp trên.

Tham số tham chiếu

Các biến đưa tới khi gọi hàm sẽ thay đổi nếu có các thao tác trên tham số hình thức tương ứng.

```
using System;
```

```
struct sophuc
```

```
{  
    public double thuc, ao;  
    public sophuc(double a, double b)  
    {  
        thuc=a;  
        ao=b;  
    }  
};
```

```
class MainProgram
```

```
{  
    static void tang(ref int i)  
    {  
        i++;  
    }  
    static void giam(ref int j)  
    {
```

```

        j--;
    }
    static void tang(ref sophuc c)
    {
        c.thuc++;
    }
    static void giam(ref sophuc d)
    {
        d.thuc--;
    }
    static void addString(ref string chuoai)
    {
        chuoai=chuoai+" are you";
    }
    static void Main()
    {
        int n=25;
        sophuc sp =new sophuc(4,9);
        string ch="How";
        tang(ref n);
        Console.WriteLine("1/ gia tri cua n la: {0}",n);
        giam(ref n);
        Console.WriteLine("2/ gia tri cua n la: {0}",n);
        tang(ref sp);
        Console.WriteLine("3/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
        giam(ref sp);
        Console.WriteLine("4/ So phuc: {0}+{1}i",sp.thuc, sp.ao);
        addString(ref ch);
        Console.WriteLine("5/ {0}",ch);
        Console.ReadLine();
    }
};

```

Giải thích:

- Đoạn code này giống như chương trình trên dùng từ khóa struct chỉ thay đổi bằng cách thêm từ khóa ref trước các tham số.

▪ Các tham số tham chiếu được khai báo tường minh bằng từ khóa `ref` kể cả lúc viết hàm và gọi hàm.

▪ Biên dịch và chạy thử chương trình, chúng ta có kết quả như hình sau:

```
E:\TRUYEN_THAM_SO\bin\Debug\TRUYEN_THAM_SO.exe
```

```
1/ gia tri cua n la: 26  
2/ gia tri cua n la: 25  
3/ So phuc: 5+9i  
4/ So phuc: 4+9i  
5/ How are you
```

H89

Bây giờ thì giá trị các biến đưa đến bị thay đổi so với trước khi gọi hàm.

Tham số kết quả

C# hỗ trợ thêm kiểu truyền tham số kết quả để lưu kết quả tính toán sau khi gọi hàm. Tham số này được khai báo với từ khóa `out` có cú pháp lập trình hoàn toàn giống như từ khóa `ref` (lúc cài đặt và gọi hàm), ý nghĩa gần giống tham số tham chiếu nhưng cần chú ý các đặc điểm khác biệt như sau:

Không cần khởi tạo biến sẽ được gửi đến tham số kết quả của hàm (ngược lại đối với tham số giá trị và tham số tham chiếu thì C# bắt buộc phải khởi tạo các biến gửi đến)

Bên trong mỗi phương thức phải có ít nhất một lệnh gán giá trị cho mỗi tham số kết quả của hàm.

BÀI 13

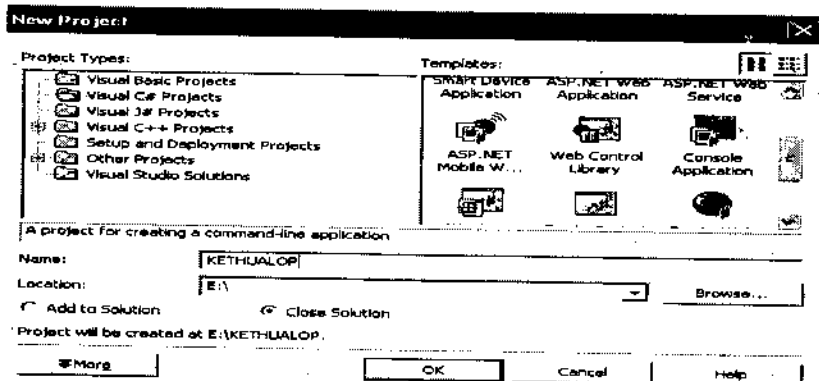
KẾ THỪ LỚP ĐỐI TƯỢNG

Một trong những đặc điểm nổi bật của ngôn ngữ lập trình hướng đối tượng là việc kế thừa mã nguồn đã có từ các lớp có sẵn. Ý tưởng kế thừa được dựa trên quan hệ rất tự nhiên của thế giới thực.

Ví dụ:

Bởi vì hình vuông là trường hợp đặc biệt của hình chữ nhật nên nó cũng có đầy đủ các tính chất của hình chữ nhật. Nên ta có thể tạo một lớp mới kế thừa lớp hình chữ nhật đã có. Khi lớp B kế thừa từ lớp A, ta nói A là lớp cơ sở (cũng gọi là lớp cha) còn B là lớp kế thừa (cũng gọi là lớp dẫn xuất hay là lớp con). Bạn thiết kế chương trình như sau:

- * Vào Microsoft Visual Studio .NET
- * Click chuột vào menu **File**, chọn **New**, chọn **Project**
- * Trong cửa sổ **New Project** hiện ra, bạn chọn:
- * **Visual C# Project** trong khung bên trái có tên **Project Types**.
- * Di chuyển con chuột bên khung bên phải, tìm chọn **Console Application**.
- * Đặt tên **KETHUALOP** trong khung **Name** như hình sau:



* Click nút OK

Xóa hết code có sẵn trong khung, nhập vào đoạn code như sau:

```
using System;
class hcn
{
    protected float dai, rong;
    public hcn(float d, float r)
    {
        dai=d;
        rong=r;
    }
    public float dientich()
    {
        return dai*rong;
    }
    public float chuvi()
    {
        return 2*(dai+rong);
    }
};
class hv:hcn
{
    public hv(float kt):base(kt,kt)
    {
    }
};
class MainProgram
{
    static void Main()
    {
        hcn cn=new hcn(8,6);
        hv v=new hv(8);
        Console.WriteLine("- Dien tich hinh chu nhat
                            ={0}",cn.dientich());
    }
}
```

```

Console.WriteLine("- Chu vi hình chữ nhật ={0}",cn.chuvi());
Console.WriteLine("- Diện tích hình vuông ={0}",v.dientich());
Console.WriteLine("- chu vi hình vuông ={0}",v.chuvi());
Console.ReadLine();
}
}

```

Giải thích:

- Thiết lập lớp hcn (hình chữ nhật) bao gồm một constructor và hai phương thức tính diện tích và chu vi.
- Khai báo hai biến dài và rộng có kiểu số thực float. Từ khóa **protected** (lớp hcn) là để che dấu hai trường dài và rộng trong lớp hcn nhưng vẫn cho truy xuất được trong lớp kế thừa hv (hình vuông) (khác với trường private là không được truy xuất ngay cả đối với lớp kế thừa).
 - Các kiểu truy xuất đối với thuộc tính và phương thức của lớp:
 - ◆ Khai báo **private**: được hiển thị mặc nhiên nếu không quy định từ khóa này. Chỉ dùng trong phạm vi lớp, không thể truy xuất từ bên ngoài hay từ lớp kế thừa.
 - ◆ Khai báo **protected**: được truy xuất từ lớp kế thừa
 - ◆ Khai báo **public**: không giới hạn phạm vi, được truy xuất từ bên ngoài lớp.
 - Thiết lập lớp hv (hình vuông) thừa kế lớp hcn (hình chữ nhật), bạn thấy khai báo **class hv:hcn**

class tên lớp con : tên lớp cha hoặc
class tên lớp cơ sở : tên lớp dẫn xuất

 - Từ khóa **base** trong câu lệnh public hv(float kt):base(kt,kt) dùng để gọi lại constructor của lớp cơ sở trong quan hệ kế thừa, tức là constructor của lớp hcn.
 - Trong đoạn chương trình chính khai báo biến cn có kiểu hcn có hai đối dài =8 và rộng = 6 đồng thời cấp bộ nhớ cho nó.
 - Khai báo biến v có kiểu hv với đối là 8.
 - In ra màn hình diện tích hình chữ nhật.
 - In ra màn hình chu vi hình chữ nhật.
 - In ra màn hình diện tích hình vuông, sử dụng phương thức dientich của hcn (kế thừa)

- In ra màn hình chu vi hình vuông, sử dụng phương thức chuvi của hcn (kế thừa)
- Biên dịch và chạy thử chương trình, bạn thấy kết quả như hình sau:

 E:\KETHUA\bin\Debug\KETHUA.exe

```
- Dien tich hinh chu nhat =48
- Chu vi hinh chu nhat =28
- Dien tich hinh vuong =64
- chu vi hinh vuong =32
```

H91

Nếu bạn thích nhập độ dài các cạnh của hình chữ nhật, hình vuông bằng một số bất kỳ, bạn sửa lại code của chương trình như sau:

```
using System;
class hcn
{
    protected float dai, rong;
    public hcn(float d, float r)
    {
        dai=d;
        rong=r;
    }
    public float dientich()
    {
        return dai*rong;
    }
    public float chuvi()
    {
```

```

        return 2*(dai+rong);
    }
};
class hv:hcn
{
    public hv(float kt):base(kt,kt)
    {
    }
};
class MainProgram
{
    static void Main()
    {
        float dai, rong, canh;
        Console.Write("Ban nhap chieu dai cua hinh chu nhac: ");
        dai=float.Parse(Console.ReadLine());
        Console.Write("Ban nhap chieu rong cua hinh chu nhac: ");
        rong=float.Parse(Console.ReadLine());
        Console.Write("Ban nhap canh cua hinh vuong: ");
        canh=float.Parse(Console.ReadLine());
        hcn cn=new hcn(dai,rong);
        hv v=new hv(canh);
        Console.WriteLine("- Dien tich hinh chu nhac
                           ={0}",cn.dientich());
        Console.WriteLine("- Chu vi hinh chu nhac ={0}",cn.chuvi());
        Console.WriteLine("- Dien tich hinh vuong ={0}",v.dientich());
        Console.WriteLine("- chu vi hinh vuong ={0}",v.chuvi());
        Console.ReadLine();
    }
}

```

Giải thích:

- Bạn thêm các dòng lệnh đậm ở phía trên. Đầu tiên là khai báo các biến dai, rong, canh có kiểu số thực float.
- Nhập giá trị chiều dài của hình chữ nhật
- Nhập giá trị chiều rộng của hình chữ nhật

- Nhập giá trị cạnh hình vuông
- Cấp phát bộ nhớ cho biến cn có kiểu hcn và đối là hai biến dai và rong vừa nhập
- Cấp phát bộ nhớ cho biến v có kiểu hv và đối là biến canh vừa nhập.
- Các lệnh khác đã giải thích ở phần trên
- Biên dịch và chạy thử chương trình, giả sử nhập chiều dài = 12, chiều rộng = 8, cạnh hình vuông = 15. Ta được kết quả như hình sau:

E:\KETHUA\bin\Debug\KETHUA.exe

```

Ban nhap chieu dai cua hinh chu nhat: 12
Ban nhap chieu rong cua hinh chu nhat: 8
Ban nhap canh cua hinh vuong: 15
- Dien tich hinh chu nhat =96
- Chu vi hinh chu nhat =40
- Dien tich hinh vuong =225
- chu vi hinh vuong =60
  
```

H92

Bạn tự định dạng để có đơn vị của từng lời giải, nếu quên, xem lại các bài tập ở trên.

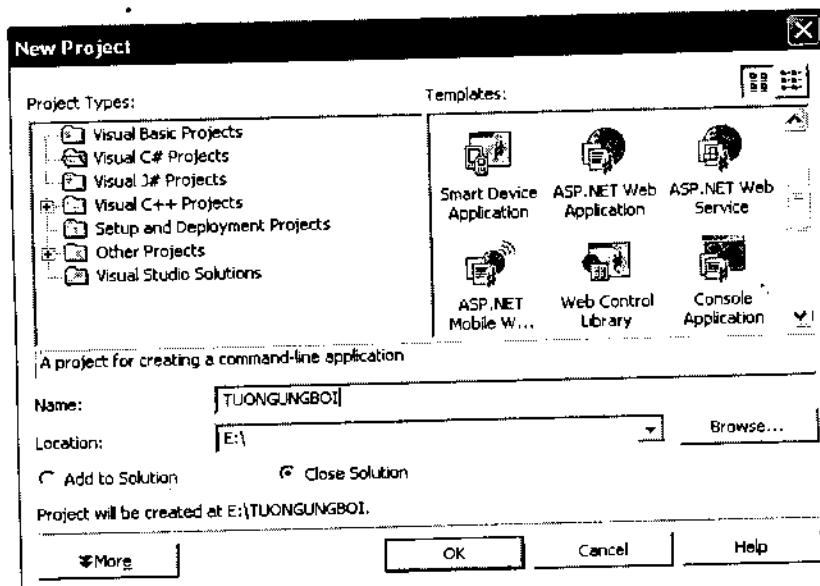
BÀI 14

TƯƠNG ỨNG BỘỊ (POLYMORPHISM)

Đây là cơ chế đặc sắc của các ngôn ngữ lập trình hướng đối tượng. Khi có nhiều đối tượng khác kiểu mà có phương thức trùng tên, cơ chế này cho phép một biến tham chiếu đến đối tượng gọi đúng phương thức cài đặt cho lớp của chính đối tượng đó.

Bạn xem ví dụ sau:

- * Vào Microsoft Visual Studio .NET
- * Click chuột vào menu **File**, chọn **New**, chọn **Project**
- * Trong cửa sổ **New Project** hiện ra, bạn chọn:
- * **Visual C# Project** trong khung bên trái có tên **Project Types**.
- * Di chuyển con chạy bên khung bên phải, tìm chọn **Console Application**.
- * Đặt tên **TUONGUNGBOI** trong khung **Name** như hình sau:



* Click nút OK

Chúng tôi viết code như sau:

```
using System;
class chung
{
    public virtual float dientich()
    {
        return 0;
    }
};
class vongtron:chung
{
    float pi=3.1416F;
    float bk;
    public vongtron (float r)
    {
        bk=r;
    }
    public override float dientich()
    {
        return pi*bk*bk;
    }
};
class hinhchunhat:chung
{
    float dai, rong;
    public hinhchunhat(float d, float r)
    {
        dai=d;
        rong=r;
    }
    public override float dientich()
    {
        return dai*rong;
    }
};
```

```

    }
};
class MainProgram
{
    static void Main()
    {
        chung goi;
        vongtron vt=new vongtron(5);
        hinhchunhat hcn=new hinhchunhat(4,3);
        goi = vt;
        Console.WriteLine(goi.dientich());
        goi = hcn;
        Console.WriteLine(goi.dientich());
        Console.ReadLine();
    }
}

```

Giải thích:

- Lớp có tên chung khai báo phương thức `dientich()` với từ khóa `virtual` (phương thức ảo). Hai phương thức `vongtron` và `hinhchunhat` kế thừa từ lớp chung.

- Cài đặt cụ thể phương thức `dientich()` để tính diện tích hình tròn và hình chữ nhật. Nhờ khai báo `virtual` nên có cơ chế tương ứng bội.

- Trong hàm `Main()`, khai báo biến `goi` có kiểu chung, khai báo biến `vt` có kiểu `vongtron` có bán kính bằng 5 và cấp phát bộ nhớ cho nó, khai báo biến `hcn` có kiểu `hinhchunhat` có hai đối chiếu dài bằng 4 và chiều rộng bằng 3.

- Khi biến `goi` trở đến `vt` (kiểu `vongtron`) thì lệnh `goi.dientich()` sẽ gọi phương thức `dientich()` của lớp `vongtron`, nếu biến `goi` trở đến `hcn` (kiểu `hinhchunhat`) thì cũng là lệnh `goi.dientich()` nhưng lại gọi đến phương thức của lớp `hinhchunhat`.

- Biên dịch và chạy thử chương trình, bạn thấy kết quả như hình sau:



78.54

12

H94

- Kiểm tra lại ta thấy:

Diện tích hình tròn: $3.1416 * 25 = 78.54$

Diện tích hình chữ nhật: $4 * 3 = 12$

Bây giờ nếu muốn tính thêm chu vi hình tròn và chu vi hình chữ nhật. Bạn phải thêm phương thức chuvi với từ khóa virtual, cài đặt thêm hai phương thức tính chu vi để tính chu vi hình tròn và chu vi hình chữ nhật. Bạn xem đoạn code sửa lại như sau:

```
using System;
class chung
{
    public virtual float dientich()
    {
        return 0;
    }
    public virtual float chuvi()
    {
        return 0;
    }
};
class vongtron:chung
{
    float pi=3.1416F;
    float bk;
    public vongtron (float r)
    {
        bk=r;
    }
}
```

```

public override float dientich()
{
    return pi*bk*bk;
}
public override float chuvi()
{
    return 2*pi*bk;
}
};
class hinhchunhat:chung
{
    float dai, rong;
    public hinhchunhat(float d, float r)
    {
        dai=d;
        rong=r;
    }
    public override float dientich()
    {
        return dai*rong;
    }
    public override float chuvi()
    {
        return 2*(dai+rong);
    }
};
class MainProgram
{
    static void Main()
    {
        chung goi;
        vongtron vt=new vongtron(5);
        hinhchunhat hcn=new hinhchunhat(4,3);
        goi = vt;
        Console.WriteLine(goi.dientich());
    }
}

```



```

Console.WriteLine(goi.chuvi());
goi = hcn;
Console.WriteLine(goi.dientich());
Console.WriteLine(goi.chuvi());
Console.ReadLine();
}
}

```

- Biên dịch và chạy thử chương trình, bạn có kết quả như hình sau:

```

E:\TUONGUNGBOI\bin\Debug\tUONGUNGBOI.exe
78.54
31.416
12
14

```

H95

Bạn tính và thấy kết quả chính xác phải không. Bạn thử thay đổi code để có thể nhập vào bán kính, chiều dài, chiều rộng và khi kết quả in ra sẽ có đơn vị nằm ở phía sau (giả sử đơn vị là cm). Bạn viết code như sau:

```

using System;
class chung
{
    public virtual float dientich()
    {
        return 0;
    }
    public virtual float chuvi()
    {
        return 0;
    }
};
class vongtron:chung

```

```

{
float pi=3.1416F;
float bk;
public vongtron (float r)
{
    bk=r;
}
public override float dientich()
{
    return pi*bk*bk;
}
public override float chuvi()
{
    return 2*pi*bk;
}
};
class hinhchunhat:chung
{
float dai, rong;
public hinhchunhat(float d, float r)
{
    dai=d;
    rong=r;
}
public override float dientich()
{
    return dai*rong;
}
public override float chuvi()
{
    return 2*(dai+rong);
}
};
class MainProgram
{

```

```

static void Main()
{
    chung goi;
    float bankinh, cd, cr;
    Console.Write("Ban nhap vao gia tri cua ban kinh: ");
    bankinh=float.Parse(Console.ReadLine());
    Console.Write("Ban nhap vao chieu dai cua hinh chu nhac: ");
    cd=float.Parse(Console.ReadLine());
    Console.Write("Ban nhap vao chieu rong cua hinh chu nhac: ");
    cr=float.Parse(Console.ReadLine());
    vongtron vt=new vongtron(bankinh);
    hinhchunhat hcn=new hinhchunhat(cd,cr);
    goi = vt;
    Console.WriteLine("Dien tich vong tron ={0}cm2",goi.dientich());
    Console.WriteLine("Chu vi cua vong tron ={0}cm",goi.chuvi());
    goi = hcn;
    Console.WriteLine("Dien tich hinh chu nhac
                        ={0}cm2",goi.dientich());
    Console.WriteLine("Chu vi hinh chu nhac ={0}cm",goi.chuvi());
    Console.ReadLine();
}
}

```

Giải thích:

■ Phần này chúng tôi đã giải thích ở trên, nếu quên các bạn có thể xem lại ở những bài trước.