

# *Visual Studio 2008*



## LẬP TRÌNH C# 2008 CƠ BẢN

## MỤC LỤC

<b>Chương 1. Cơ bản ngôn ngữ C# .....</b>	<b>1</b>
I. Giới thiệu ngôn ngữ C# 2008 .....	1
II. Môi trường lập trình.....	2
III. Biến, hằng, toán tử .....	3
IV. Quy ước lập trình, ứng dụng Console Application trong Visual Studio 2008.....	5
V. Kiểu dữ liệu .....	9
VI. Cấu trúc điều khiển .....	10
VII.Cấu trúc lặp .....	12
<b>Chương 2. Xây dựng Windows Forms Application.....</b>	<b>15</b>
I. Sử dụng Visual Studio 2008 .....	15
II. Các control cơ bản: Label, TextBox, Button, CheckBox, ... .....	18
III. Menu và ToolBar .....	30
IV. Common Dialog.....	30
<b>Chương 3. Array – String – Exception .....</b>	<b>34</b>
I. Mảng 1 chiều .....	34
II. Mảng nhiều chiều.....	37
III. String.....	40
IV. Exception .....	45
<b>Chương 4. Class – Object - Method .....</b>	<b>47</b>
I. Khái niệm.....	47
II. Định nghĩa lớp (Class) .....	47
III. Phương thức (Method).....	50
<b>Chương 5. SQL Server 2008 .....</b>	<b>54</b>
I. Tổng quan về SQL .....	54
II. Tổng quan về CSDL quan hệ.....	55
III. Table (Bảng) .....	58
IV. Câu lệnh truy vấn .....	66
V. Một số hàm thường dùng trong SQL Server .....	70
<b>Chương 6. Lập trình kết nối CSDL SQL Server 2008 .....</b>	<b>72</b>
I. Tạo kết nối – Vận chuyển dữ liệu.....	72
II. Sử dụng control.....	73
III. Các thao tác trên dữ liệu: Thêm - Sửa - Xóa với ADO.NET.....	78
<b>Chương 7. Xây dựng ứng dụng.....</b>	<b>85</b>
I. Chuẩn bị .....	85
II. Sử dụng control.....	85
III. Sử dụng database .....	88

## Chương 1: CƠ BẢN NGÔN NGỮ C#

### Bài 1: GIỚI THIỆU C# 2008

#### I. Giới thiệu C#

Ngôn ngữ C# khá đơn giản, chỉ khoảng hơn 80 từ khóa và hơn mươi mấy kiểu dữ liệu được dựng sẵn. Tuy nhiên, ngôn ngữ C# có ý nghĩa to lớn khi nó thực thi những khái niệm lập trình hiện đại. C# bao gồm tất cả những hỗ trợ cho cấu trúc, thành phần component, lập trình hướng đối tượng. Những tính chất đó hiện diện trong một ngôn ngữ lập trình hiện đại. Hơn nữa ngôn ngữ C# được xây dựng trên nền tảng hai ngôn ngữ mạnh nhất là C++ và Java.

Tóm lại, C# có các đặc trưng sau đây:

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

#### 1. C# là ngôn ngữ đơn giản

- C# loại bỏ được một vài sự phức tạp và rối rắm của các ngôn ngữ C++ và Java.  
- C# khá giống C / C++ về diện mạo, cú pháp, biểu thức, toán tử.  
- Các chức năng của C# được lấy trực tiếp từ ngôn ngữ C / C++ nhưng được cải tiến để làm cho ngôn ngữ đơn giản hơn.

#### 2. C# là ngôn ngữ hiện đại

C# có được những đặc tính của ngôn ngữ hiện đại như:

- Xử lý ngoại lệ
- Thu gom bộ nhớ tự động
- Có những kiểu dữ liệu mở rộng
- Bảo mật mã nguồn

#### 3. C# là ngôn ngữ hướng đối tượng

C# hỗ trợ tất cả những đặc tính của ngôn ngữ hướng đối tượng là:

- Sự đóng gói (encapsulation)
- Sự kế thừa (inheritance)
- Đa hình (polymorphism)

#### 4. C# là ngôn ngữ mạnh mẽ và mềm dẻo

- Với ngôn ngữ C#, chúng ta chỉ bị giới hạn ở chính bản thân của chúng ta. Ngôn ngữ này không đặt ra những ràng buộc lên những việc có thể làm.  
- C# được sử dụng cho nhiều dự án khác nhau như: tạo ra ứng dụng xử lý văn bản, ứng dụng đồ họa, xử lý bảng tính; thậm chí tạo ra những trình biên dịch cho các ngôn ngữ khác.  
- C# là ngôn ngữ sử dụng giới hạn những từ khóa. Phần lớn các từ khóa dùng để mô tả thông tin, nhưng không gì thế mà C# kém phần mạnh mẽ. Chúng ta có thể tìm thấy rằng ngôn ngữ này có thể được sử dụng để làm bất cứ nhiệm vụ nào.

#### 5. C# là ngôn ngữ hướng module

- Mã nguồn của C# được viết trong Class (lớp). Những Class này chứa các Method (phương thức) thành viên của nó.  
- Class (lớp) và các Method (phương thức) thành viên của nó có thể được sử dụng lại trong những ứng dụng hay chương trình khác.

#### 6. C# sẽ trở nên phổ biến

C# mang đến sức mạnh của C++ cùng với sự dễ dàng của ngôn ngữ Visual Basic.

## II. Môi trường lập trình

### 1. Sử dụng Notepad soạn thảo

- **Bước 1:** Soạn thảo tập tin và lưu với tên **C:\ChaoMung.cs** có nội dung như sau

```
class ChaoMung
{
    static void Main()
    {
        // Xuat ra man hinh chuoi thong bao 'Chao mung ban den voi C# 2008 '
        System.Console.WriteLine("Chao mung ban den voi C# 2008 ");
        System.Console.ReadLine();
    }
}
```

- **Bước 2:** Vào menu Start | All Programs | Microsoft Visual Studio 2008 | Visual Studio Tools | Visual Studio 2008 Command Prompt

- **Bước 3:**

- Gõ lệnh biên dịch tập tin **ChaoMung.cs** sang tập tin **ChaoMung.exe**

**C:\> csc /t:exe /out:chaomung.exe chaomung.cs**

- Chạy tập tin **ChaoMung.exe** và được kết quả như sau :

**C:\> chaomung.exe**

Chao mung ban den voi C# 2008

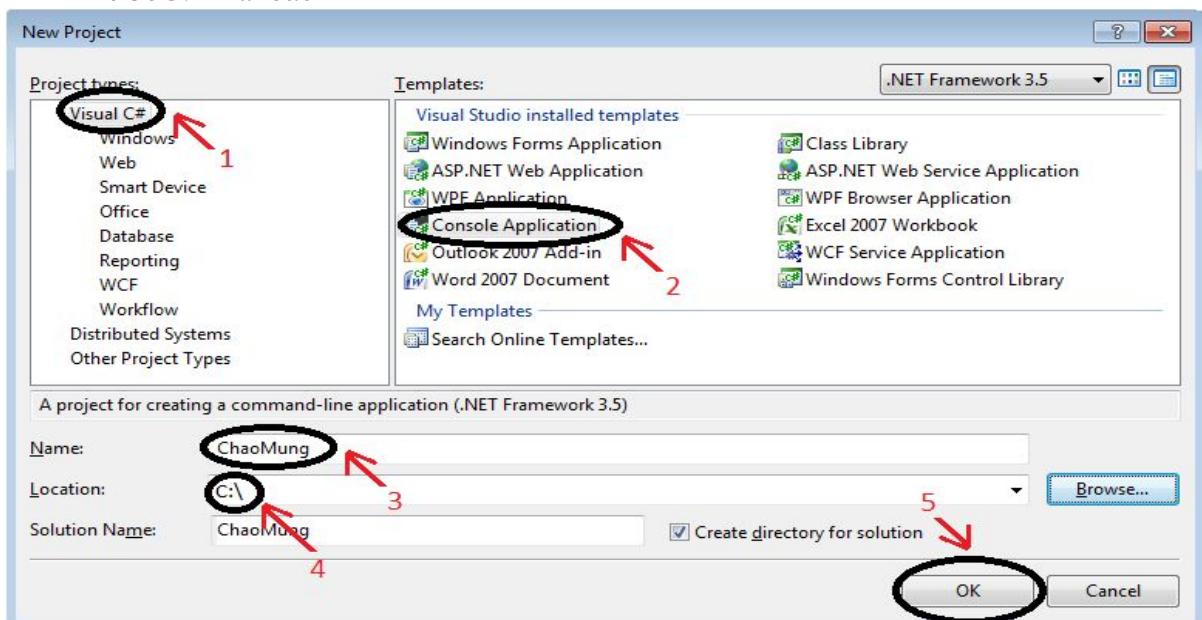
### 2. Sử dụng Microsoft Visual Studio 2008 để tạo chương trình

- **Bước 1:** Khởi động Visual Studio 2008

Start | All Programs | Microsoft Visual Studio 2008 | Microsoft Visual Studio 2008

- **Bước 2:** Vào menu File | New | Project

- **Bước 3:** Khai báo



\* Mặc định: Visual Studio 2008 (Visual Studio .NET) sẽ tạo ra tập tin Program.cs chứa một **namespace** tên ChaoMung và trong namespace này chứa một **class** tên Program.

▪ **Bước 4:** trong phương thức Main, gõ đoạn mã lệnh sau

\* Ví dụ:

```
// Xuat ra man hinh chuoi thong bao 'Chao mung ban den voi C# 2008 '
System.Console.WriteLine("Chao mung ban den voi C# 2008 ");
System.Console.ReadLine();
```

▪ **Bước 5:** Để chạy chương trình, nhấn F5 hoặc nhấp vào nút

### III. Biến, hằng, toán tử

#### 1. Biến

a) **Khái niệm:**

- Biến là một vùng lưu trữ ứng với một kiểu dữ liệu.
- Biến có thể được gán giá trị và cũng có thể thay đổi giá trị trong khi thực hiện các lệnh của chương trình.
- b) **Khai báo biến:** Sau khi khai báo biến phải gán giá trị cho biến  
`<Kiểu_Dữ_Liệu> <tên_biến> [ = <giá_trị> ] ;`
- c) **Ví dụ 1.1:** Khởi tạo và gán giá trị một biến

```
class Bien
{
    static void Main()
    {
        // Khai bao va khoi tao bien
        int bien = 9 ;
        System.Console.WriteLine("Sau khi khoi tao: bien = {0}", bien) ;
        // Gan gia tri cho bien
        bien = 5 ;
        // Xuat ra man hinh
        System.Console.WriteLine("Sau khi gan: bien = {0}", bien) ;
    }
}
```

#### 2. Hằng

a) **Khái niệm:**

- Hằng cũng là một biến nhưng giá trị của hằng không thay đổi trong khi thực hiện các lệnh của chương trình.

- Hằng được phân làm 3 loại:

- + Giá trị hằng (literal)
- + Biểu tượng hằng (symbolic constants)
- + Kiểu liệt kê (enumerations)

b) **Giá trị hằng:**

Ví dụ: `x = 100;` // 100 được gọi là giá trị hằng

c) **Biểu tượng hằng:** gán một tên hằng cho một giá trị hằng.

Khai báo:

`<const> <Kiểu_Dữ_Liệu> <tên_hằng> = <giá_trị> ;`

**Ví dụ 1.2:** Nhập vào bán kính, in ra chu vi và diện tích hình tròn.

```
class HinhTron
{
    static void Main()
    {
        // Khai bao bien tuong hang
        const double PI = 3.14159 ;
        // Khai bao bien
        int bankinh ;
        double chuvi , dientich ;
        string chuoi ;
        // Nhap gia tri cho bien chuoi
        System.Console.WriteLine("Nhập bán kính hình tròn: ") ;
        chuoi = System.Console.ReadLine() ;
        // Đổi chuỗi thành số và gán vào biến số
        bankinh = System.Convert.ToInt32(chuoi) ;
        // Gán giá trị cho biến
        chuvi = 2 * bankinh * PI ;
        dientich = bankinh * bankinh * PI ;
        // Xuất ra màn hình
        System.Console.WriteLine("Chu vi hình tròn = {0:0.00}", chuvi) ;
        System.Console.WriteLine("Diện tích hình tròn = {0:0.00}", dientich) ;
    }
}
```

- d) **Kiểu liệt kê:** là tập hợp các tên hằng có giá trị số không thay đổi (danh sách liệt kê)  
Khai báo:

```
<enum> <Tên_Kiểu_Liệt_Kê>
{
    <tên_hằng_1> = <giá_trị_số_1>,
    <tên_hằng_2> = <giá_trị_số_2>,
    ... ,
}
```

Ví dụ:

```
enum NheitDoNuoc
{
    DoDong = 0, DoNguoi = 20,
    DoAm = 40, DoNong = 60,
    DoSoi = 100,
}
```

### 3. Toán tử

- a) **Toán tử toán học:** +, -, \*, /, %
- b) **Toán tử tăng / giảm:** +=, -=, \*=, /=, %=
- c) **Toán tử tăng / giảm 1 đơn vị:** ++, --
- d) **Toán tử gán:** =
- e) **Toán tử quan hệ:** ==, !=, >, >=, <, <=
- f) **Toán tử logic:** !, &&, ||
- g) **Toán tử 3 ngôi:** (Điều\_Kiện) ? (Biểu\_Thúc\_1) : (Biểu\_Thúc\_2) ;

## IV. Quy tắc lập trình, ứng dụng Console Application

### 1. Quy tắc lập trình

Khi tạo một chương trình trong C#, chúng ta nên thực hiện theo các bước sau:

- **Bước 1:** Xác định mục tiêu của chương trình
- **Bước 2:** Xác định những phương pháp giải quyết vấn đề
- **Bước 3:** Tạo một chương trình để giải quyết vấn đề
- **Bước 4:** Thực thi chương trình để xem kết quả

### 2. Ứng dụng Console Application

Là ứng dụng giao tiếp với người dùng thông qua bàn phím và không có giao diện người dùng (UI).

#### Ví dụ 1.3:

```
using System;
class ChaoMung
{
    static void Main()
    {
        // Xuat ra man hinh chuoi thong bao 'Chao mung ban den voi C# 2008 '
        Console.WriteLine("Chao mung ban den voi C# 2008 ");
        Console.ReadLine();
    }
}
```

#### \* Phản Bổ sung

##### 1. Namespace

- .NET cung cấp một thư viện các class rất đồ sộ, trong đó **Console** là một class nhỏ trong thư viện các class này.

- Mỗi class có một tên riêng, vì vậy người lập trình không thể nào nhớ hết tên các class trong .NET. Để giải quyết vấn đề này là việc tạo ra một **namespace**, **namespace** sẽ hạn chế phạm vi của một tên, làm cho tên này chỉ có ý nghĩa trong vùng đã định nghĩa.

##### 2. Từ khóa using

- Để không phải viết namespace cho từng đối tượng, ta dùng từ khóa using.

- Ta có thể dùng dòng lệnh sau ở đầu chương trình:

```
using System;
```

Khi đó, thay vì viết đầy đủ **System.Console**, ta chỉ cần viết **Console**.

##### 3. Từ khóa static

Từ khóa static chỉ ra rằng hàm Main() có thể được gọi mà không cần phải tạo đối tượng ChaoMung.

##### 4. Từ khóa this

Từ khóa this dùng để tham chiếu đến thể hiện hiện hành của đối tượng.

##### 5. Chú thích (Comment)

- Một chương trình được viết tốt thì cần phải có chú thích các đoạn mã lệnh được viết.

- Mục đích chính là làm cho đoạn mã lệnh nguồn rõ ràng và dễ hiểu.

- Có 2 loại chú thích:

+ Chú thích một dòng: //

+ Chú thích nhiều dòng: /\* \*/

##### 6. Phân biệt chữ thường và chữ hoa

C# là ngôn ngữ phân biệt chữ thường với chữ hoa.

## 7. Toán tử '.'

Toán tử '.' được sử dụng để truy cập đến phương thức hay dữ liệu trong một class và ngăn cách giữa tên class đến một namespace.

Ví dụ: **System.Console.WriteLine()**



Một chỉ dẫn lập trình đầy đủ được gọi là một câu lệnh.

Ví dụ: int bankinh = 5 ; // một câu lệnh

chuvi = 2 \* bankinh \* PI ; // một câu lệnh khác

## 9. Kiểu chuỗi ký tự

Kiểu chuỗi ký tự là một mảng các ký tự.

### a) Khai báo chuỗi hằng:

string <Tên\_chuỗi\_hằng> = <"Nội dung chuỗi hang"> ;

Ví dụ: string tentuong = "Nhat Nghe" ;

### b) Khai báo biến kiểu chuỗi:

string <Biến\_chuỗi> [= "Nội dung chuỗi hang"] ;

Ví dụ: string hoten = "Nguyen Van Teo" ;

### c) Nhập chuỗi:

<Biến\_chuỗi> = System.Console.ReadLine() ;

Ví dụ: hoten = System.Console.ReadLine() ;

### d) Xuất chuỗi:

System.Console.WriteLine("Chuoi") ;

Ví dụ: System.Console.WriteLine("Do dai cua chuoi la:") ;

### e) Một số thao tác trên chuỗi:

Phương thức	Ý nghĩa
Length	Chiều dài của chuỗi
Substring()	Lấy chuỗi con
ToLower()	Trả về bản sao của chuỗi ở kiểu chữ thường
ToUpper()	Trả về bản sao của chuỗi ở kiểu chữ IN HOA

**Ví dụ 1.4:** Nhập vào họ và tên, in ra màn hình họ tên bằng chữ IN HOA, chữ thường, độ dài của họ và tên.

```
using System ;
class HoTen
{
    static void Main()
    {
        // Khai bao bien
        string hoten ;
        // Nhap gia tri cho bien chuoi
        Console.Write("Nhap Ho va Ten: ") ;
        hoten = Console.ReadLine() ;
        // Thao tac tren chuoi
        string HT = hoten.ToUpper() ;
        string ht = hoten.ToLower() ;
        int dodai = hoten.Length ;
        // Xuat ra man hinh
        Console.WriteLine("Ho va Ten (chu IN HOA): {0}", HT) ;
        Console.WriteLine("Ho va Ten (chu thuong): {0}", ht) ;
    }
}
```

```

        }
    }

    Console.WriteLine("Do dai Ho va Ten la: {0}",dodai);
}
}

```

### 10. Bảng liệt kê các từ khóa của ngôn ngữ C# 2008

abstract	event	new	struct
as	explicit	null	switch
base	extern	object	this
bool	false	operator	throw
break	finally	out	true
byte	fixed	override	try
case	float	params	typeof
catch	for	private	unit
Char	foreach	protected	ulong
checked	goto	public	unchecked
Class	if	readonly	unsafe
Const	implicit	ref	ushort
continue	in	return	using
decimal	interface	sbyte	virtual
default	internal	sealed	volatile
delegate	is	short	void
do	lock	sizeof	while
double	long	stackalloc	
else	namespace	static	
enum		string	
from		get	group
into		join	let
orderby		partial (type)	partial (method)
select		set	value
where (generic type constraint)		where (query clause)	yield

### Bài tập

- Viết chương trình nhập vào 1 số nguyên n. Cho biết:  
 a) n là số chẵn hay số lẻ?  
 b) n là số âm hay số không âm?
- Viết chương trình nhập vào 2 số thực dương chỉ chiều dài và chiều rộng của hình chữ nhật. In ra màn hình chu vi và diện tích của hình chữ nhật đó.

3. Viết chương trình nhập vào một số thực dương chỉ cạnh của một hình vuông. Tính diện tích và chu vi của hình vuông đó.
4. Viết chương trình nhập vào họ tên (HoTen), điểm toán (Toan), điểm lý (Ly), điểm hoá (Hoa) của một học sinh. In ra màn hình họ tên của học sinh dưới dạng chữ IN HOA và điểm trung bình (Dtb) của học sinh này theo công thức:  $Dtb = (Toan + Ly + Hoa) / 3$
5. Viết chương trình nhập bậc lương (BacLuong), ngày công (NgayCong), phụ cấp (PhuCap). Tính tiền lanh (TienLanh) = BacLuong \* 650000 \* NCTL + PhuCap  
Với: NCTL = NgayCong nếu  $NgayCong < 25$   
=  $(NgayCong - 25) * 2 + 25$  nếu  $NgayCong \geq 25$   
--- oOo ---

## Bài 2: (tiếp theo)

### KIẾU DỮ LIỆU – CẤU TRÚC ĐIỀU KHIỂN – CẤU TRÚC LẮP

#### V. Kiểu dữ liệu

C# chia kiểu dữ liệu thành hai tập hợp kiểu dữ liệu chính:

- Kiểu xây dựng sẵn (built-in): do ngôn ngữ cung cấp cho người lập trình.
- Kiểu do người dùng định nghĩa (user-defined): do người lập trình tạo ra.

#### 1. Kiểu dữ liệu dựng sẵn

Kiểu C#	Số byte	Kiểu .NET	Mô tả
byte	1	Byte	Số nguyên dương không dấu từ 0 đến 255
char	2	Char	Ký tự Unicode
bool	1	Boolean	Giá trị logic true / false
sbyte	1	Sbyte	Số nguyên có dấu từ -128 đến 127
short	2	Int16	Số nguyên có dấu từ -32768 đến 32767
ushort	2	UInt16	Số nguyên dương không dấu từ 0 đến 65535
int	4	Int32	Số nguyên có dấu từ -2.147.483.647 đến 2.147.483.647
uint	4	UInt32	Số nguyên không dấu từ 0 đến 4.294.967.295
float	4	Single	Kiểu dấu chấm động, giá trị xấp xỉ từ -3.4E-38 đến 3.4E+38, với 7 chữ số có nghĩa
double	8	Double	Kiểu dấu chấm động có độ chính xác gấp đôi, giá trị xấp xỉ từ -1.7E-308 đến 1.7E+308, với 15, 16 chữ số có nghĩa
decimal	8	Decimal	Có độ chính xác đến 28 con số và giá trị thập phân, được dùng trong tính toán tài chính, kiểu này đòi hỏi phải có hậu tố "m" hay "M"
long	8	Int64	Kiểu số nguyên có dấu có giá trị trong khoảng -9.223.370.036.854.775.808 đến 9.223.372.036.854.775.807
ulong	8	UInt64	Số nguyên không dấu từ 0 đến 0xffffffffffffffff

#### \* Bảng trình bày các ký tự đặc biệt

Ký tự	Ý nghĩa
\'	Dấu nháy đơn
\\"	Dấu nháy kép
\\"	Dấu chéo
\0	Ký tự null
\a	Alert
\b	Backspace
\f	Sang trang form feed
\n	Dòng mới
\r	Đầu dòng
\t	Tab ngang
\v	Tab dọc

## 2. Chuyển đổi kiểu dữ liệu

Ví dụ a:

```
short x = 10 ;  
int y = x ; // chuyển đổi ngầm định
```

Ví dụ b:

```
short x ;  
int y = 100 ;  
x = (short) y ; // ép kiểu tường minh, trình biên dịch không báo lỗi
```

Ví dụ c:

```
short x ;  
int y = 100 ;  
x = y ; // không biên dịch, lỗi
```

## VI. Cấu trúc điều khiển

### 1. Câu lệnh if ... else

#### a) Cú pháp:

```
if (Điều_Kiện)  
    <Khối lệnh Điều_Kiện đúng>  
[else  
    <Khối lệnh Điều_Kiện sai>]
```

#### b) Ví dụ 2.1: Dùng câu lệnh điều kiện if ... else

```
using System;  
class Chan_Le  
{  
    static void Main()  
    {  
        // Khai bao va khai tao bien  
        int bienDem = 9 ;  
        // Xuat ra man hinh  
        if (bienDem % 2 == 0)  
            Console.WriteLine("{0} la so chan", bienDem) ;  
        else      Console.WriteLine("{0} la so le", bienDem) ;  
    }  
}
```

### 2. Câu lệnh if lồng nhau

#### a) Cú pháp:

```
if (Điều_Kiện_1)  
    <Khối lệnh 1>  
else if (Điều_Kiện_2)  
    <Khối lệnh 2.1>  
else  
    <Khối lệnh 2.2>
```

#### b) Ví dụ 2.2:

```
using System;  
class Thu_Trong_Tuan
```

```
{
    static void Main()
    {
        // Khai bao va khoi tao bien
        int thu = 5; // 0: Chu nhat, 1: Thu hai, 2: Thu ba, 3: Thu tu,
                    // 4: Thu nam, 5: Thu sau, 6: Thu bay
        // Xuat ra man hinh
        if ((thu == 1) || (thu == 3) || (thu == 5))
            Console.WriteLine("Day la ngay 2-4-6");
        else if ((thu == 2) || (thu == 4) || (thu == 6))
            Console.WriteLine("Day la ngay 3-5-7");
        else    Console.WriteLine("Day la ngay chu nhat");
    }
}
```

### 3. Câu lệnh switch

#### a. Cú pháp:

```
switch (Biểu_Thức)
{
    case <giá_trị_1>:
        < Khối lệnh 1>
        <Lệnh Nhảy>
    case <giá_trị_2>:
        < Khối lệnh 2>
        <Lệnh Nhảy>
    ....
    [default:
        < Khối lệnh khác>]
}
```

#### b. Ví dụ 2.3:

```
using System;
class Thu
{
    static void Main()
    {
        // Khai bao va khoi tao bien
        int thu = 5; // 0: Chu nhat, 1: Thu hai, 2: Thu ba, 3: Thu tu,
                    // 4: Thu nam, 5: Thu sau, 6: Thu bay
        // Xuat ra man hinh
        switch (thu)
        {
            case 0:
                Console.WriteLine("Chu nhat");
                break;
            case 1:
                Console.WriteLine("Thu hai");
                break;
            case 2:
                Console.WriteLine("Thu ba");
                break;
            case 3:
                Console.WriteLine("Thu tu");
                break;
            case 4:
                Console.WriteLine("Thu nam");
                break;
            case 5:
                Console.WriteLine("Thu sau");
                break;
            case 6:
                Console.WriteLine("Thu bay");
                break;
        }
    }
}
```

```
        Console.WriteLine("Thu ba") ;
        break;
    case 3:
        Console.WriteLine("Thu tu") ;
        break;
    case 4:
        Console.WriteLine("Thu nam") ;
        break;
    case 5:
        Console.WriteLine("Thu sau") ;
        break;
    case 6:
        Console.WriteLine("Thu bay") ;
        break;
    default:
        Console.WriteLine("Khong phai la thu trong tuan") ;
        break;
    }
}
```

## VII. Cấu trúc lặp

### 1. Lệnh lặp while

#### a. Cú pháp:

**while** (Điều\_Kiện)  
    < Khối lệnh>

#### b. Ví dụ 2.4:

```
using System;
class UsingWhile
{
    static void Main()
    {
        // Khai bao va khai tao bien dem
        int i = 1 ;
        // Xuat ra man hinh
        while (i<=10) {
            Console.WriteLine("i = {0}",i) ;
            i++ ; // tang bien dem,
        }
    }
}
```

### 2. Lệnh lặp do ... while

#### a. Cú pháp:

**do**  
    < Khối lệnh>  
**while** (Điều\_Kiện) ;

**b. Ví dụ 2.5:**

```
using System;
class UsingDoWhile
{
    static void Main()
    {
        // Khai bao va khai tao bien dem
        int i = 1 ;
        // Xuat ra man hinh
        do {
            Console.WriteLine("i = {0}",i);
            i++; // tang bien dem
        } while (i<= 10);
    }
}
```

**3. Lệnh lặp for****a. Cú pháp:**

```
for ([Khởi_tạo] ; [Điều_kiện] ; [Bước_lặp])
    < Khối_lệnh>
```

**b. Ví dụ 2.6:**

```
using System;
class UsingFor
{
    static void Main()
    {
        for (int i=1 ; i<=30 ; i++)
            if (i % 10 ==0)
                Console.WriteLine("{0} \n",i);
            else    Console.WriteLine("{0} ",i);
    }
}
```

**4. Lệnh lặp foreach****a. Cú pháp:**

```
foreach (<Kiểu_tập_hợp> <Tên_truy_cập_thành_phần> in <Tên_tập_hợp>)
    < Khối_lệnh>
```

**b. Ví dụ 2.7:**

```
using System;
public class UsingForeach
{
    public static void Main()
    {
        int[] intArray = {1,2,3,4,5,6,7,8,9,10};
        foreach (int item in intArray)
            Console.WriteLine("i = {0} ",item);
    }
}
```

## Bài tập

1. Viết chương trình nhập vào 3 số nguyên. In ra màn hình số nguyên nhỏ nhất trong 3 số đó.
2. Viết chương trình nhập vào họ tên, điểm thi cuối kỳ của một học sinh. In ra họ tên học sinh bằng chữ IN HOA, và kết quả xếp loại của học sinh theo tiêu chuẩn sau:
  - Giỏi: Nếu Điểm kết quả  $\geq 8$
  - Khá: Nếu  $8 > Điểm \geq 6.5$
  - Trung bình: Nếu  $6.5 > Điểm \geq 5$
  - Yếu: Nếu Điểm  $< 5$
3. Viết chương trình giải phương trình bậc 1:  $bx + c = 0$
4. Viết chương trình giải phương trình bậc 2:  $ax^2 + bx + c = 0$
5. Viết chương trình nhập vào một số nguyên cho đến khi nhận được số nguyên dương thì dừng.
6. Viết chương trình nhập vào một số nguyên n. Cho biết số nguyên n có phải là số nguyên tố không ?
7. Viết chương trình nhập vào một số nguyên dương n chỉ năm dương lịch. Cho biết n có phải là năm nhuận không ?
8. Viết chương trình nhập vào số nguyên dương n. In ra màn hình kết quả của các tổng sau:
  - a)  $S_1 = 1 + 2 + 3 + \dots + n$
  - b)  $S_2 = 1 + \frac{1}{2} + \frac{1}{3} + \dots + \frac{1}{n}$
9. Viết chương trình nhập vào số nguyên dương n. In ra màn hình:
  - a) Các số nguyên dương từ 1 đến n
  - b) Tổng và trung bình cộng của n số nguyên dương này.

--- oOo ---

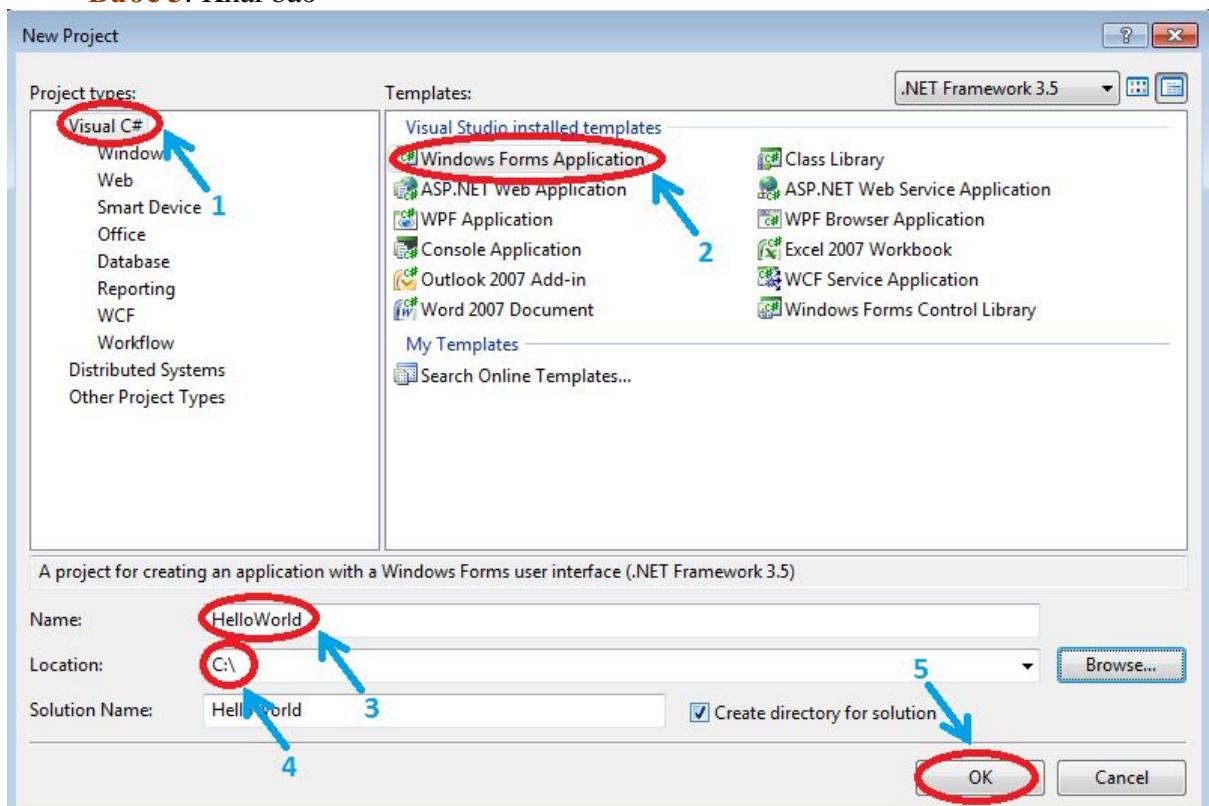
## Chương 2: XÂY DỰNG WINDOWS FORMS APPLICATION

### Bài 3: XÂY DỰNG WINDOWS FORMS APPLICATION

#### I. Sử dụng Microsoft Visual Studio 2008

##### 1. Khởi động

- **Bước 1:** Khởi động Visual Studio 2008  
Start | All Programs | Microsoft Visual Studio 2008 | Microsoft Visual Studio 2008
- **Bước 2:** Vào menu File | New | Project
- **Bước 3:** Khai báo



- Mở hộp ToolBox: Menu View | ToolBox → chứa các control
- Mở cửa sổ Properties: Menu View | Properties → chứa thuộc tính
- Mở cửa sổ Solution Explorer: Menu View | Solution Explorer → cửa sổ Project xuất hiện.
- **Bước 4:** Thiết kế Form – Viết code
  - Thiết kế form: Nhấp vào View Designer (trong cửa sổ Solution Explorer)
  - Viết code: Nhấp vào View Code (trong cửa sổ Solution Explorer)
- **Bước 5:** Để chạy chương trình, nhấn F5 hoặc nhấp vào nút Để dừng chương trình, nhấn Shift + F5 hoặc nhấp vào nút

#### \* Các thao tác với Project / Solution

##### a. Tạo Project

C1. Vào menu File | New | Project

C2. Ctrl + Shift + N

C3. Chọn công cụ **New Project** trên thanh Standart

**b. Mở Project / Solution:**

C1. Vào menu File | Open | Project / Solution

C2. Ctrl + Shift + O

**c. Lưu Project / Solution**

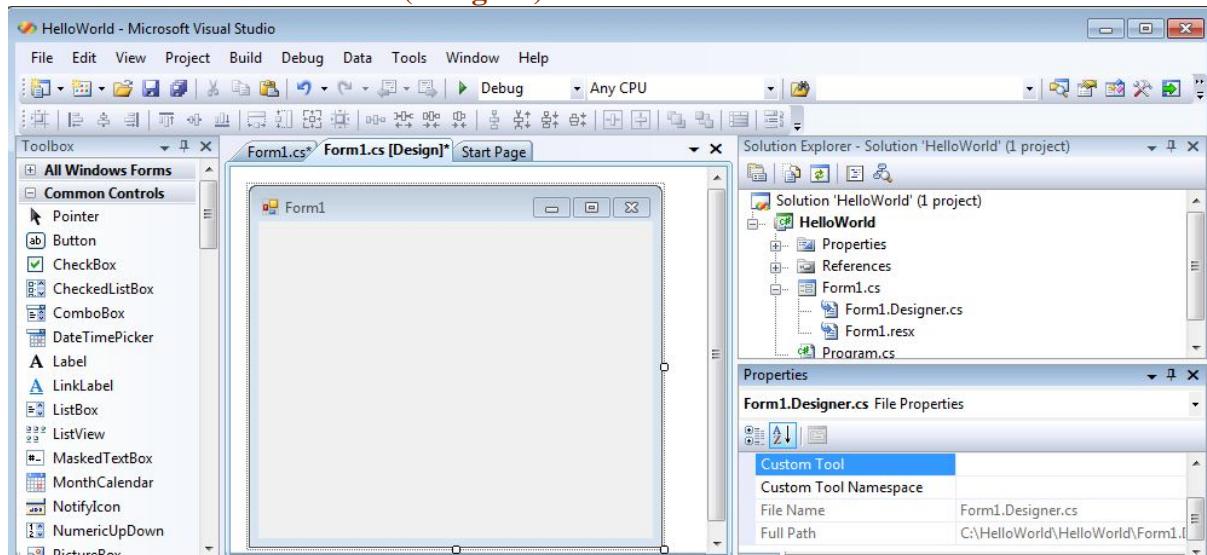
C1. Vào menu File | Save All

C2. Chọn công cụ **Save All** trên thanh Standart

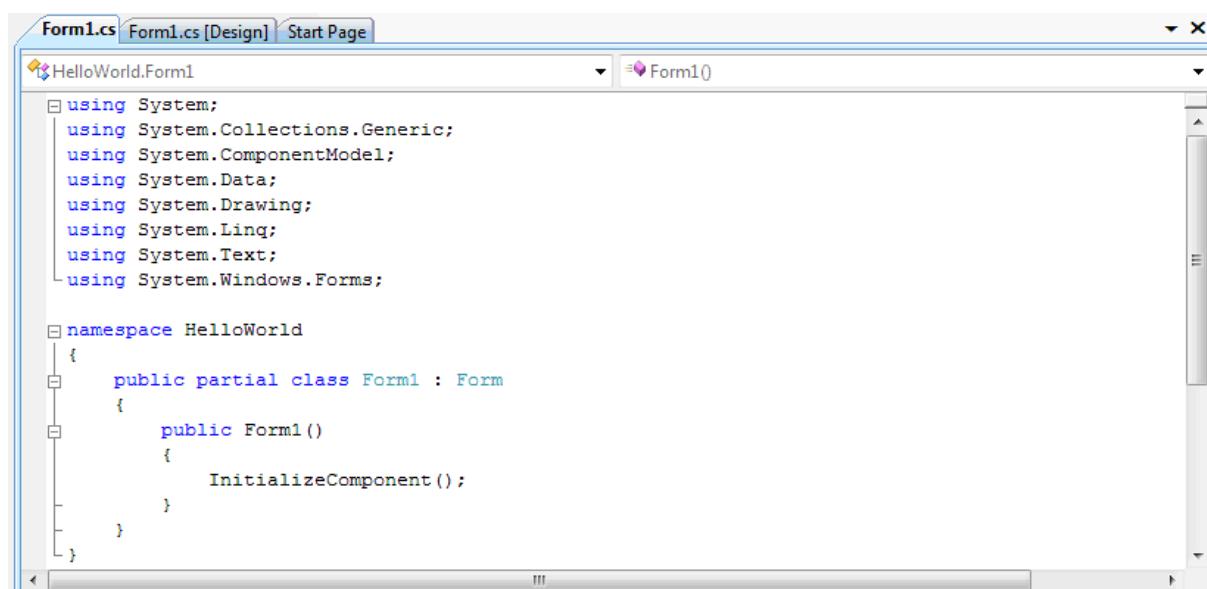
**d. Đóng Solution:** Vào menu File | Close Solution

## 2. Màn hình giao diện của Windows Forms

**a. Cửa sổ thiết kế Form (Designer):**



**b. Cửa sổ thiết kế viết code:**



**\* Các thao tác với Form**

**a. Thêm một Form mới vào Project:**

b1. C1. Vào menu Project | Add New Item ...

C2. Chọn công cụ **Add New Item** trên thanh Standart

b2. Khai báo

+ Categories: chọn Windows Forms

+ Templates: chọn Windows Form

+ Name: đặt tên Form

b3. Nhấp Add

#### **b. Thêm một Form có sẵn vào Project:**

b1. Vào menu Project | Add Existing Item ...

b2. Chọn Form

b3. Nhấp Add

#### **c. Xóa bỏ một Form đang có trong Project:**

b1. Chọn Form cần gỡ bỏ (ở cửa sổ Solution Explorer)

b2. Vào menu Edit | Delete

#### **d. Lưu Form**

- Vào menu File | Save Form.cs

- Ctrl + S

#### **\* Ghi chú**

- Ta đem “bỏ vào” form các đối tượng như: Label, TextBox, Button, ...

+ Label, TextBox, Button, ... được gọi là control hay còn gọi là component.

+ Form được gọi là control “chúa”.

- Khi thay đổi nội dung của Label, TextBox, Button, ... ta thay đổi vào Text. Text được gọi là Property của control.

### **3. Control là gì?**

- Control là lớp (class) các thành phần được thêm vào Windows Forms để tương tác giữa người sử dụng với Windows.

- Có rất nhiều loại control trong Windows Forms như: Label, TextBox, ListBox, ComboBox, Button, ...

- Các control sử dụng trên Windows Forms dùng namespace **System.Windows.Forms**.

### **4. Properties (thuộc tính) của control**

- Properties là những thông tin mà ta có thể thay đổi nội dung, cách trình bày ... của người thiết kế để ứng dụng vào control.

- Mỗi lớp (class) có nhiều property khác nhau. Tuy nhiên, vẫn có một số property giống nhau được xây dựng từ lớp ban đầu.

#### **\* Bảng trình bày các thuộc tính (Properties) giống nhau**

Thuộc tính	Mô tả
Anchor	Có 4 hướng được định nghĩa là: top, bottom, left, right để cố định (neo). Khi control chứa nó thay đổi kích thước thì nó sẽ bị thay đổi kích thước nếu như các hướng left / right / top / bottom bị cố định (neo).
BackColor	Màu nền của control.
Bottom	Là khoảng cách theo chiều dọc từ cạnh đáy của control đến cạnh trên của control chứa nó.
Dock	Giống như Anchor nhưng việc cố định (neo) này theo một cạnh nào đó của control (hoặc cả 4 cạnh) với control chứa nó.
Enabled	Control được phép tương tác (True) hay không được phép tương tác (False)) với người dùng.

ForeColor	Màu chữ của control.
Height	Là chiều cao của control tính từ cạnh trên của control đến cạnh dưới của control.
Left	Là khoảng cách theo chiều ngang từ cạnh trái của control đến cạnh trái của control chứa nó.
Name	Tên của control.
Parent	Chỉ đến control chứa control hiện hành.
Right	Là khoảng cách theo chiều ngang từ cạnh phải của control đến cạnh trái của control chứa nó.
TabIndex	Thứ tự focus khi nhấn phím Tab (trên bàn phím) của control so với các control khác cùng nằm trong control chứa nó.
TabStop	Chỉ định control có được phép “bắt” (True) / không được phép “bắt” (False) phím Tab. Nếu không được phép thì TabIndex cũng không dùng được.
Tag	Là nhãn phân biệt giữa các control giống nhau trong cùng form.
Text	Nội dung hiện trong control.
Top	Là khoảng cách theo chiều dọc từ cạnh trên của control đến cạnh trên của control chứa nó.
Visible	Cho phép control hiện (True) / không hiện (False) khi chạy ứng dụng.
Width	Là chiều rộng của control tính từ cạnh trái của control đến cạnh phải của control.

### \* Bảng trình bày các phương thức (Method) xử lý trên chuỗi

Phương thức	Mô tả
Clear()	Xóa nội dung
ResetText()	Xóa nội dung Text
Trim()	Cắt bỏ khoảng trắng thừa hai bên chuỗi

## II. Các control cơ bản

### 1. Label ( )

#### a. Công dụng:

- Hiển thị chuỗi ký tự không thay đổi trên form (nhãn).

#### b. Tạo Label:

- Chọn công cụ

- Rê chuột và vẽ Label trên form.

#### c. Thuộc tính:

Thuộc tính	Mô tả
AutoSize	Điều chỉnh kích thước đối tượng cho vừa với chiều dài chuỗi ký tự
Font	
Name	Quy định font chữ cho văn bản
Bold	True: đậm / False: bỏ đậm
Italic	True: nghiêng / False: bỏ nghiêng
Size	Quy định cỡ chữ cho văn bản
Underline	True: gạch dưới / False: bỏ gạch dưới
TextAlign	Canh lè (Left / Center / Right)

### 2. TextBox ( )

### a. Công dụng:

- Dùng trình bày văn bản và cho phép người dùng được thay đổi nội dung văn bản.
- Công dụng chính là cho người dùng nhập văn bản.

### b. Tạo TextBox:

- Chọn công cụ **TextBox**
- Rê chuột và vẽ TextBox trên form.

### c. Thuộc tính:

Thuộc tính	Mô tả
PasswordChar	Quy định ký tự hiển thị cho ô mật khẩu.
Multiline	True: hiện thanh cuộn / False: không hiện thanh cuộn
ScrollBars	Thanh cuộn (None / Horizontal / Vertical / Both)

### 3. Button ( **Button**)

#### a. Công dụng:

- Dùng để thực thi lệnh.
- Khi nhấp chuột lên button, chương trình nhận được tín hiệu Click và lệnh được thi hành.

#### b. Tạo Button:

- Chọn công cụ **Button**
- Rê chuột và vẽ Button trên form.

#### c. Thuộc tính:

Thuộc tính	Mô tả
Text	Nhập nội dung vào Button

### 4. CheckBox ( **CheckBox**)

#### a. Công dụng:

- Cho phép người dùng chọn hoặc không chọn.

#### b. Tạo CheckBox:

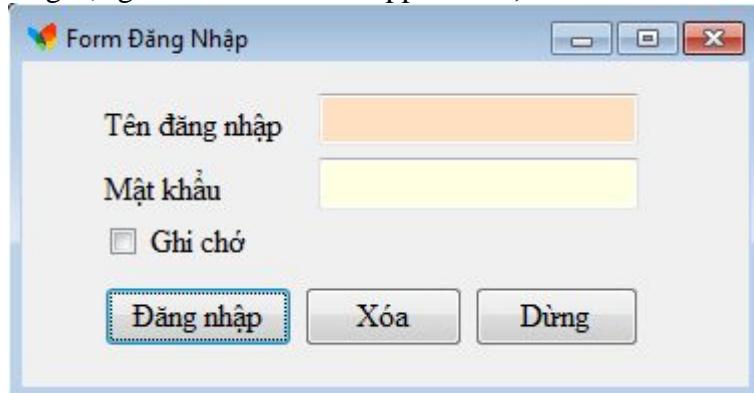
- Chọn công cụ **CheckBox**
- Rê chuột và vẽ CheckBox trên form.

#### c. Thuộc tính:

Thuộc tính	Mô tả
Checked	Không có dấu check (False) / Có dấu check (True)

### Ví dụ 3.1:

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 3.1 như sau:



\* Yêu cầu:

- Nhập:
  - + Username vào TextBox **Tên đăng nhập** (Name: *txtUser*)
  - + Password vào TextBox **Mật khẩu** (Name: *txtPass*)

- Chọn hoặc không chọn ô CheckBox **Ghi nhớ** (Name: *chkNho*)
- Nhấp button **Đăng nhập** thì hiện ra hộp thông báo chứa **Tên đăng nhập, Mật khẩu**; Và **“Bạn có ghi nhớ”** (nếu *chkNho* có đánh dấu chọn).
  - Nhấp button **Xóa** thì xóa trống TextBox **Tên đăng nhập** và TextBox **Mật khẩu**, đồng thời di chuyển con trỏ vào *txtUser*.
  - Nhấp button **Dừng** thì dừng chương trình.
- \* Hướng dẫn:
  - Thiết kế Form như yêu cầu, trong đó form có các thuộc tính sau:
    - + AutoSize: True
    - + Font: Times New Roman
    - + Size: 12
    - + Text: Form Đăng Nhập
    - + Icon: logo.ico
  - Nhấp đúp vào button **Đăng nhập**, thêm đoạn code sau:

```
string thongbao;
thongbao = "Tên đăng nhập là: " ;
thongbao += this.txtUser.Text ;
thongbao += "\n\rMật khẩu là: " ;
thongbao += this.txtPass.Text;
if (this.chkNho.Checked==true) {
    thongbao += "\n\rBạn có ghi nhớ.";
```
  - Nhấp đúp vào button **Xóa**, thêm đoạn code sau:

```
this.txtUser.Clear();
this.txtPass.Clear();
this.txtUser.Focus();
```
  - Nhấp đúp vào button **Dừng**, thêm đoạn code sau:

```
Application.Exit();
```
  - + Có thể thay button **Xóa** bằng button **Reset** với đoạn code như sau:

```
this.txtUser.ResetText();
this.txtPass.ResetText();
this.txtUser.Focus();
```

#### \* Phản Bổ sung:

##### 1. **MessageBox.Show:** hiện hộp thông báo

```
MessageBox.Show("Thông báo" , "Tiêu đề");
```

```
MessageBox.Show("Thông báo" , "Tiêu đề" , Buttons , Icon);
```

+ Buttons (nút lệnh):

```
MessageBoxButtons.OK
```

```
MessageBoxButtons.OKCancel
```

```
MessageBoxButtons.YesNo
```

```
MessageBoxButtons.YesNoCancel
```

+ Icon (biểu tượng):

```
MessageBoxIcon.Information
```

```
MessageBoxIcon.Question
```

```
MessageBoxIcon.Warning
```

+ DialogResult (kết quả trả về):

```
DialogResult.Ok
```

`DialogResult.Yes`

## 2. Dừng chương trình:

```
this.Close();
Application.Exit();
```

## 3. Event (sự kiện) là gì?

- Sự kiện là những phản ứng của đối tượng. Nói cách khác, sự kiện là những tín hiệu phát ra khi người dùng thao tác trên đối tượng.

- Nhờ có event, người lập trình sẽ nhận được những tín hiệu và xử lý những tín hiệu đó để phản hồi lại cho người dùng, tạo nên sự nhịp nhàng cho chương trình.

## 4. Bảng trình bày các sự kiện (Events) của control

Sự kiện	Mô tả
Click	Gọi đến khi control bị Click. Trong một vài control, event này cũng xảy ra khi người dùng nhấn phím Enter.
DoubleClick	Gọi đến khi control bị Double-Click. Trong một vài control, event này không bao giờ được gọi. Ví dụ: control Button.
DragDrop	Gọi đến khi việc “Drag and Drop” được hoàn tất.
DragEnter	Gọi đến khi đối tượng vừa được “Drag” đến biên của control.
DragLeave	Gọi đến khi đối tượng vừa được “Drag” ra ngoài biên của control.
DragOver	Gọi đến khi đối tượng được “Drag” bên trong control.
KeyDown	Gọi đến khi vừa bấm một phím bất kỳ từ 1 control đang focus. Sự kiện này luôn được gọi trước sự kiện KeyUp.
KeyPress	Gọi đến khi vừa bấm một phím bất kỳ từ 1 control được focus. Sự kiện này được gọi sau sự kiện KeyUp.
KeyUp	Gọi đến khi vừa bấm một phím bất kỳ rồi thả ra từ 1 control đang focus. Sự kiện này luôn được gọi sau sự kiện KeyDown.
GotFocus	Gọi đến khi control được focus.
LostFocus	Gọi đến khi control bị mất focus.
MouseDown	Gọi đến khi con trỏ chuột nằm trên 1 control và nút chuột được nhấp nhưng chưa thả ra.
MouseMove	Gọi đến khi con trỏ chuột đi qua 1 control.
MouseUp	Gọi đến khi con trỏ chuột nằm trên 1 control và nút chuột vừa được thả.
Paint	Gọi đến khi control được vẽ.
Validated	Gọi đến khi control focus, property CaucesValidation được đặt là true và sau khi gọi việc kiểm tra bằng Validating.
Validating	Gọi đến khi control mất focus, property CaucesValidation được đặt là true.

### Ví dụ 3.2:

\* Cài tiến Vi Du 3.1 cho button **Dừng** như sau:

- Khi nhấp vào button **Dừng** thì xuất hiện hộp thoại hỏi đáp có 2 button Ok, Cancel.

- Chương trình chỉ dừng khi nhấp tiếp vào nút Ok.

\* Hướng dẫn: Sửa lại button **Dừng** như sau

```
DialogResult traloi;
traloi = MessageBox.Show("Chắc không?", "Trả lời",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
if (traloi == DialogResult.OK) Application.Exit();
```

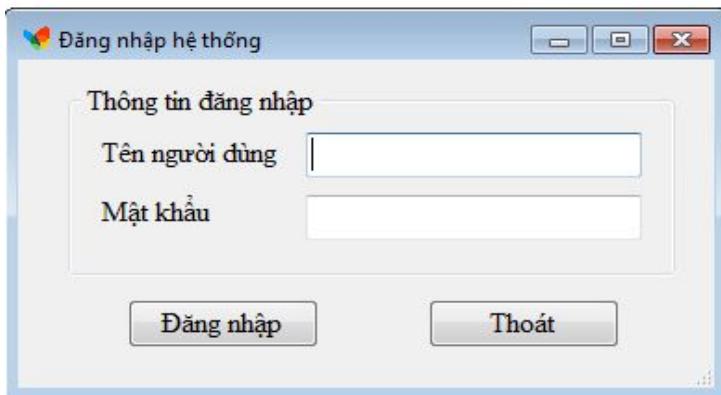
### Ví dụ 3.3:

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 3.3 gồm:

- Form1: được thiết kế như sau



- Form2: được thiết kế như sau



(txtUser, txtPass, btnDangnhap, btnThoat)

\* Yêu cầu:

- Khi chạy chương trình thì Form2 (Đăng nhập hệ thống) được mở lên trước.

- Nhập:

- + Tên người dùng vào TextBox **txtUser**
- + Mật khẩu vào TextBox **txtPass**

- Nhấp button **Đăng nhập** thì thực hiện kiểm tra

- + Nếu **txtUser** = "teonv" và **txtPass** = "123" thì chuyển sang Form1.
- + Ngược lại thông báo "**Không đúng tên người dùng / mật khẩu !!!**"

- Nhấp button **Thoát** thì hiển thị thông báo "Chắc không? "

- + Nếu chọn Yes thì kết thúc chương trình.
- + Ngược lại trở lại màn hình Đăng nhập hệ thống.

\* Hướng dẫn:

- **Form1**

- . Thiết kế Form như yêu cầu, trong đó form có các thuộc tính sau:

- + AutoSize: True
- + Font: Times New Roman
- + Size: 12
- + Text: Màn hình chính
- + Icon: star.ico

. Form load:

```
Form frm = new Form2();
frm.ShowDialog();
```

- **Form2**

- . Thiết kế Form như yêu cầu, trong đó form có các thuộc tính sau:

- + AutoSize: True
- + Font: Times New Roman

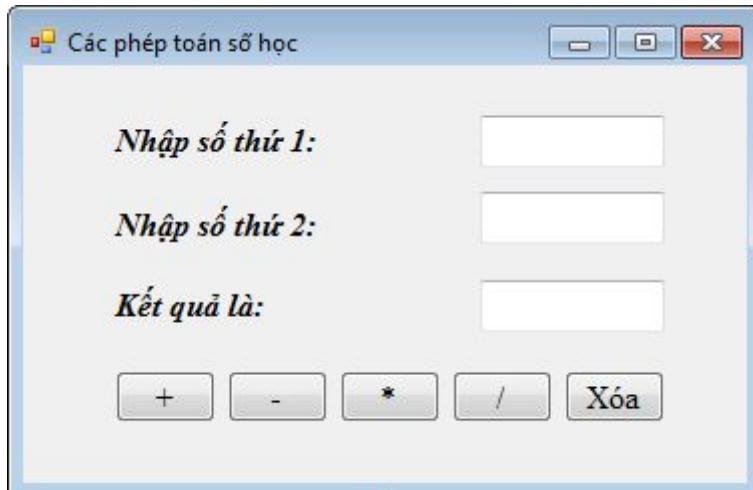
- + Size: 12
- + Text: Đăng nhập hệ thống
- + Icon: logo.ico
- . **Đăng nhập:** Nhấp đúp vào button btnDangnhap, gõ vào đoạn code sau
 

```
if ((this.txtUser.Text=="teonv")&&(this.txtPass.Text=="123"))
    this.Close();
else
{
    MessageBox.Show("Không đúng tên người dùng / mật
    khẩu !!!", "Thông báo");
    this.txtUser.Focus();
}
```
- . **Thoát:** Nhấp đúp vào button btnThoat, gõ vào đoạn code sau
 

```
 DialogResult traloi;
traloi = MessageBox.Show("Chắc không?", "Trả lời",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
if (traloi == DialogResult.OK)
    Application.Exit();
```

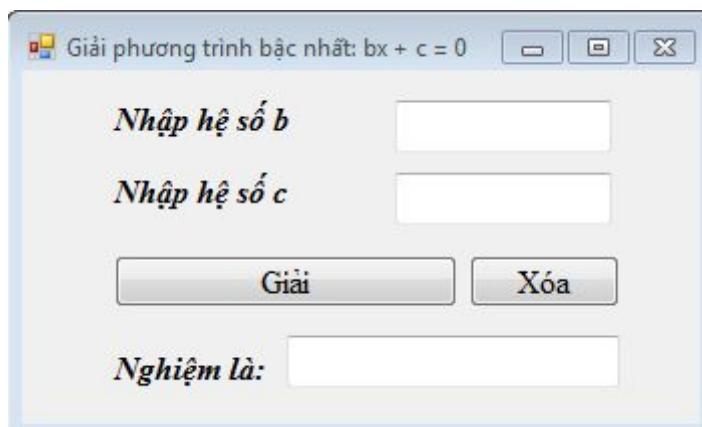
## Bài tập

1. - Thiết kế form như mẫu.



(txtSo1, txtSo2, txtKQ, btnCong, btnTru, btnNhan, btnChia, btnXoa)

- Viết chương trình làm các phép toán: cộng, trừ, nhân, chia cho các button btnCong, btnTru, btnNhan, btnChia.
  - **Xóa:** Xóa trắng các TextBox.
  - Thêm vào button **Dừng** (btnDung), khi nhấp vào btnDung thì dừng chương trình.
  - **Cải tiến:**  
Khi nhấp button btnChia, nếu txtSo2 là 0 thì xuất hiện hộp thông báo lỗi rồi xóa trống txtSo2 và di chuyển con trỏ đến TextBox này.
2. Viết chương trình giải phương trình bậc nhất:  $bx + c = 0$



(txtB, txtC, btnGiai, txtKQ)

- Thêm vào button **Dừng**, khi nhấp vào button này thì dừng chương trình.
- 3. Viết chương trình giải phương trình bậc hai:  $ax^2 + bx + c = 0$
- 4. Viết chương trình nhập vào: họ tên (txtHoTen), nữ (chkNu), điểm văn (txtVan), điểm toán (txtToan), điểm ngoại ngữ (txtNN).
  - Nhấp vào nút **Tính** (btnTinh) thì in ra điểm thấp nhất (txtDTN), điểm kết quả (txtDKQ), xếp loại (txtXL). Biết rằng:
    - + Điểm thấp nhất: txtDTN là điểm thấp nhất trong 3 điểm: văn, toán, ngoại ngữ.
    - + Điểm thêm: DThem = 0.5 nếu là nữ; DThem = 0 nếu là nam.
    - + Điểm kết quả: txtKQ = txtVan \* 2 + txtToan \* 2 + txtNN + DThem
    - + Xếp loại theo tiêu chuẩn:
      - . Giỏi: nếu txtKQ  $\geq 40$  và txtDTN  $\geq 7$
      - . Khá: nếu txtKQ  $\geq 35$  và txtDTN  $\geq 6$
      - . Trung bình: nếu txtKQ  $\geq 25$  và txtDTN  $\geq 5$
      - . Yếu: các trường hợp còn lại
  - Nhấp vào nút **Xóa** (btnXoa) thì xóa hết các nội dung trong các TextBox.
  - Thêm vào button **Dừng**, khi nhấp vào button này thì dừng chương trình.
- 5. Viết chương trình tạo một ứng dụng gồm:
  - Form1: Màn hình chính có 5 button: **Bài tập 1** (btnBT1), **Bài tập 2** (btnBT2), **Bài tập 3** (btnBT3), **Bài tập 4** (btnBT4), **Thoát** (btnThoat).
    - + Khi nhấp vào Button **Bài Tập 1**: mở Form2
    - + Khi nhấp vào Button **Bài Tập 2**: mở Form3
    - + Khi nhấp vào Button **Bài Tập 3**: mở Form4
    - + Khi nhấp vào Button **Bài Tập 4**: mở Form5
    - Form2: thực hiện bài tập 1. Bỏ sung Button **Trở về** để đóng Form2.
    - Form3: thực hiện bài tập 2. Bỏ sung Button **Trở về** để đóng Form3.
    - Form4: thực hiện bài tập 3. Bỏ sung Button **Trở về** để đóng Form4.
    - Form5: thực hiện bài tập 4. Bỏ sung Button **Trở về** để đóng Form5.

--- oOo ---

## Bài 4: (tiếp theo) XÂY DỰNG WINDOWS FORMS APPLICATION

### 5. **ListBox** ( **ListBox** )

#### a. Công dụng:

- Dùng để hiển thị một danh sách các lựa chọn.

#### b. Tạo ListBox:

- Chọn công cụ  **ListBox**

- Rê chuột và vẽ **ListBox** trên form.

#### c. Thuộc tính:

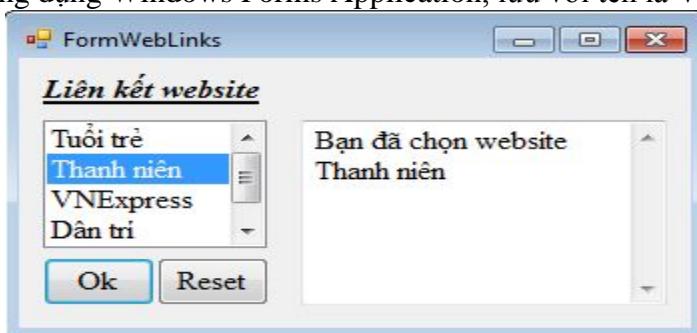
Thuộc tính	Mô tả
Items	Các mục giá trị trong <b>ListBox</b>
SelectedItem	Item được chọn
Phương thức	Mô tả
Add("chuỗi")	Thêm một mục giá trị là "chuỗi"
ToString()	Trả về chuỗi ký tự được chọn

\* Nhập giá trị vào **ListBox**: <Ten\_ListBox>.Items.Add ("Chuỗi") ;

\* Lấy giá trị trong **ListBox**: <Ten\_ListBox>.SelectedItem.ToString() ;

#### Ví dụ 4.1:

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 4.1 như sau:



\* Yêu cầu:

- **ListBox lstWeb** (Liên kết website) chứa các giá trị:

- + Tuổi trẻ
- + Thanh niên
- + VNEexpress
- + Dân trí
- + Công an

- **TextBox txtKQ** (chứa kết quả) để trống.

- Nhấp button **btnOk** (Ok) sẽ hiện trong **txtKQ** tên website được chọn ở **lstWeb**.

- Nhấp button **btnReset** (Reset) sẽ xóa trống **txtKQ**.

\* Hướng dẫn:

- Thiết kế Form như yêu cầu, trong đó form có các thuộc tính sau:

- + AutoSize: True
- + Font: Times New Roman
- + Size: 12
- + Text: FormWebLinks

- Nhấp đúp vào button **Ok** rồi thêm đoạn code sau:

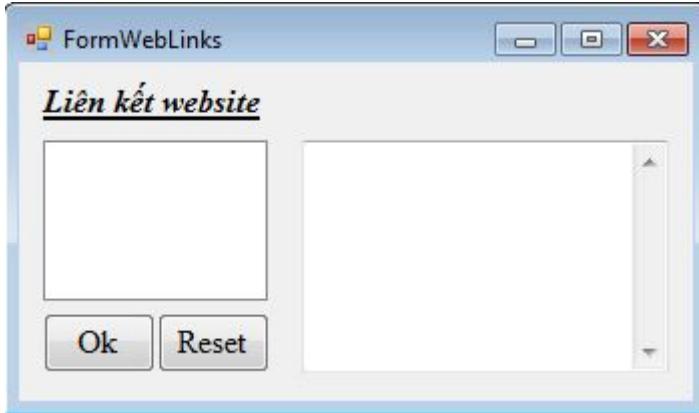
```
this.txtKQ.Text = "Bạn đã chọn website ";
this.txtKQ.Text += this.lstWeb.SelectedItem.ToString();
```

- Nhấp đúp vào button **Reset** rồi thêm đoạn code sau:

```
this.txtKQ.ResetText();
```

- \* Cài tiến: Ta có thể đưa các giá trị của lstWeb trong **Form1\_Load**.

- + Thiết kế lại form như sau



- + Nhấp đúp chuột vào nền form, rồi gõ đoạn code

```
this.lstWeb.Items.Add("Tuổi trẻ");
this.lstWeb.Items.Add("Thanh niên");
this.lstWeb.Items.Add("VNE");
this.lstWeb.Items.Add("Dân trí");
this.lstWeb.Items.Add("Công an");
this.lstWeb.SelectedItem = "Tuổi trẻ";
```

## 6. ComboBox ()

### a. Công dụng:

- Dùng để hiển thị một danh sách các lựa chọn / hoặc nhập vào một giá trị.

### b. Tạo ComboBox:

- Chọn công cụ

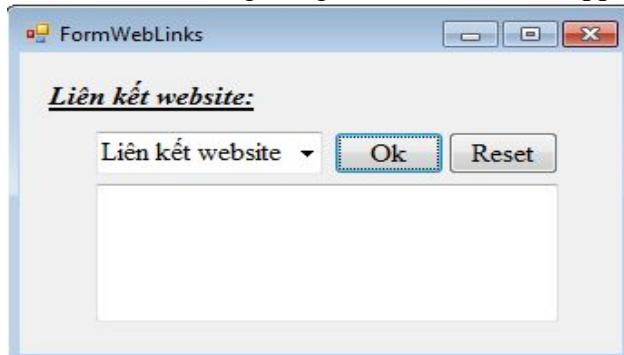
- Rê chuột và vẽ ComboBox trên form.

### c. Thuộc tính:

Thuộc tính	Mô tả
DisplayMember	Gán nội dung thể hiện trên ComboBox
Items	Liệt kê các mục giá trị trong ComboBox
SelectedItem	Lấy Item được chọn
SelectedText	Lấy nội dung thể hiện trên ComboBox từ DisplayMember
SelectedValue	Lấy giá trị từ ValueMember
ValueMember	Gán giá trị cho ComboBox

### Ví dụ 4.2:

- \* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 4.2 như sau:



\* Yêu cầu:

Thực hiện giống như Ví dụ 4.1, nhưng thay ListBox bằng ComboBox cbWeb.

## 7. RadioButton ( RadioButton )

### a. Công dụng:

- Dùng để chọn một trong các lựa chọn trong danh sách.

### b. Tạo RadioButton:

- Chọn công cụ  RadioButton

- R rê chuột và vẽ RadioButton trên form.

### c. Thuộc tính:

Thuộc tính	Mô tả
Checked	Không có dấu chọn (False) / Có dấu chọn (True)

## 8. GroupBox ( GroupBox )

### a. Công dụng:

- Tạo ra một nhóm.

### b. GroupBox:

- Chọn công cụ  GroupBox

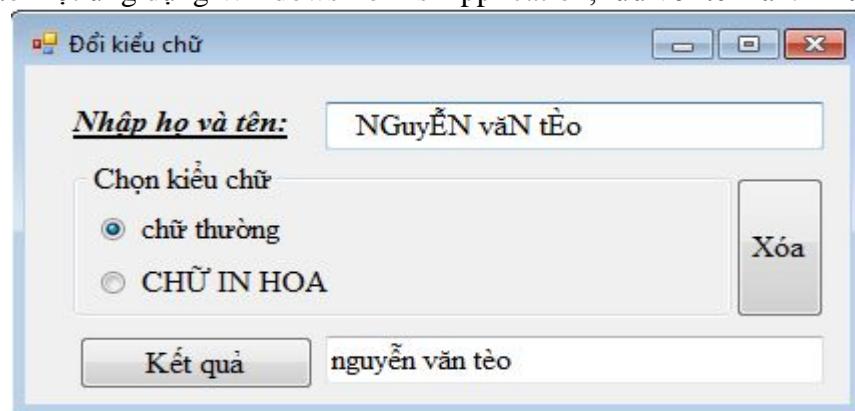
- R rê chuột và vẽ GroupBox trên form.

### c. Thuộc tính:

Thuộc tính	Mô tả
BackgroundImage	Hình nền
BackgroundImageLayout	None / Tile / Center / Stretch / Zoom

### Ví dụ 4.3:

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 4.3 như sau:



#### \* Yêu cầu:

- Nhập họ và tên vào TextBox txtHoTen

#### - Chọn kiểu chữ

- + Radio Button (rad1): chữ thường

- + Radio Button (rad2): chữ HOA

- Nhấp vào Button Kết quả (btnKQ) sẽ in họ và tên bằng kiểu chữ được chọn trong TextBox txtKQ.

- Nhấp vào Button Xóa (btnXoa) sẽ xóa trống txtHoTen, txtKQ, rad1 được chọn và đưa con trỏ vào ô TextBox txtHoTen.

#### \* Hướng dẫn:

- Thiết kế Form như yêu cầu, trong đó form có các thuộc tính sau:

- + AutoSize: True

- + Font: Times New Roman

- + Size: 12
- + Text: Đổi kiểu chữ
- Nhấp đúp vào nút **Kết quả** rồi thêm đoạn code sau:
 

```
string hoten=this.txtHoTen.Text.Trim();
if (this.rad1.Checked == true)
    txtKQ.Text = hoten.ToLower();
if (this.rad2.Checked == true)
    txtKQ.Text = hoten.ToUpper();
```
- Nhấp đúp vào nút **Xóa** rồi thêm đoạn code sau:
 

```
this.txtHoTen.Clear();
this.txtKQ.Clear();
this.rad1.Checked = true;
this.txtHoTen.Focus();
```
- \* Bổ sung:
  - Nhấp vào Button Dừng (btnDung) sẽ dừng chương trình: thiết kế và viết code sau

```
Application.Exit();
```

## 9. Timer ( )

### a. Công dụng:

- Quy định khoảng thời gian định kỳ để thực hiện một công việc.

### b. Tạo Timer:

- Chọn công cụ Timer
- Rê chuột và vẽ Timer → là control dạng invisible (ẩn).

### c. Thuộc tính:

Thuộc tính	Mô tả
Enabled	Bật / tắt chế độ hẹn thời gian
Interval	Khoảng thời gian định kỳ

### Ví dụ 4.4:

- \* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 4.4 như sau:



### \* Yêu cầu:

- Tạo Timer1 có Enabled = false; Interval = 1000
- Khởi tạo biến đếm i = 20
- Button Bắt đầu (btnBatDau): dùng để bật chế độ hẹn thời gian.
- Button Dừng (btnDung): dùng để dừng chương trình.
- Timer1: Đếm ngược từ 20 đến 1 và xuất ra dòng chữ “Hết giờ”.
  - + Xuất giá trị biến đếm i ra Label lblDongHo.
  - + Giảm biến i xuống 1 đơn vị.
  - + Khi biến đếm i < 0 thì tắt chế độ hẹn giờ và xuất “Hết giờ!” ra Label lblDongHo.

### \* Hướng dẫn:

- Thiết kế Form như yêu cầu, trong đó form có các thuộc tính sau:

- + AutoSize: True
- + Font: Times New Roman
- + Size: 12
- + Text: Đồng hồ đếm ngược
- Khai báo biến đếm i: qua code, thêm đoạn code để được kết quả như sau:

```
public Form1()
{
    InitializeComponent();
}
int i = 10; //=20
```

- Nhấp đúp vào nút **Bắt đầu** rồi thêm đoạn code sau:

```
this.timer1.Enabled = true;
```

- Nhấp đúp vào nút **Dừng** rồi thêm đoạn code sau:

```
Application.Exit();
```

- Nhấp đúp vào nút **Timer1** (Timer1\_Tick) rồi thêm đoạn code sau:

```
this.lblDongHo.Text = i.ToString();
i--;
if (i < 0)
{
    this.timer1.Enabled = false;
    this.lblDongHo.Text = "Hết giờ!" }
```

## 10. RichTextBox ( RichTextBox)

### a. Công dụng:

- Dùng để nhập văn bản với định dạng văn bản đa dạng.

### b. Tạo RichTextBox:

- Chọn công cụ  RichTextBox

- Rê chuột và vẽ RichTextBox trên form.

### c. Thuộc tính:

Thuộc tính	Mô tả
ReadOnly	Không cho soạn thảo, chỉ đọc.

## 11. Panel ( Panel)

### a. Công dụng:

- Bảng chứa các control hay một nhóm các control.

### b. Tạo Panel:

- Chọn công cụ  Panel

- Rê chuột và vẽ Panel.

### c. Thuộc tính:

Thuộc tính	Mô tả
AutoScroll	Tự động cuộn nếu số control nằm ngoài vùng
BorderStyle	None / FixedSingle / Fixed3D

## 12. PictureBox ( PictureBox)

### a. Công dụng:

- Khung chứa hình ảnh.

### b. Tạo PictureBox:

- Chọn công cụ  PictureBox

- Rê chuột và vẽ PictureBox.

### c. Thuộc tính:

Thuộc tính	Mô tả
------------	-------

Image	Hình chứa trong PictureBox
-------	----------------------------

### 13. ErrorProvider ( ErrorProvider )

#### a. Công dụng:

- Hỗ trợ thông báo lỗi cho các control khác.
- Thường được dùng với control input (ví dụ: TextBox) ràng buộc với 1 điều kiện nhập nào đó.

#### b. Tạo ErrorProvider:

- Chọn công cụ ErrorProvider
- Rê chuột và vẽ ErrorProvider → là control dạng invisible (ẩn).

## III. Menu và ToolBar

### 1. ToolStrip ( ToolStrip )

#### a. Công dụng:

- Tạo menu.

#### b. Tạo ToolStrip:

- Nhấp đúp vào control ToolStrip .
- Nhập menu.

### 2. ToolStrip ( ToolStrip )

#### a. Công dụng:

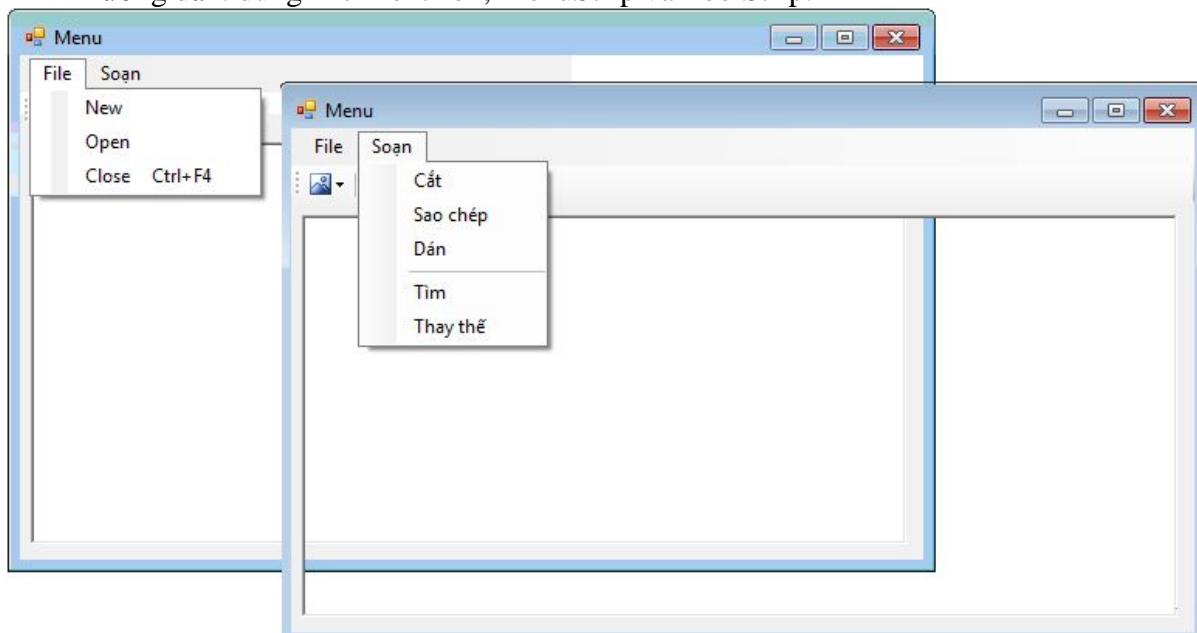
- Tạo ra toolbar.

#### b. Tạo ToolStrip:

- Chọn công cụ ToolStrip
- Kéo control ToolStrip vào trong form.
- Bấm vào ToolStrip vừa tạo, ta có thể tạo ra các tool (công cụ) như: Label, TextBox, Button, ComboBox, DropDownButton....

#### Ví dụ 4.5:

- \* Tạo ứng dụng Windows Forms Application như hình.
- \* Hướng dẫn: dùng RichTextBox, MenuStrip và ToolStrip.



## IV. Common Dialog (Hộp thoại dùng chung)

### 1. **FontDialog** ( **FontDialog** )

#### a. **Công dụng:**

- Tạo ra hộp thoại Font.

#### b. **Tạo FontDialog:**

- Nhấp đúp vào control  **FontDialog**

### 2. **OpenFileDialog** ( **OpenFileDialog** )

#### a. **Công dụng:**

- Tạo ra hộp thoại Open File.

#### b. **Tạo OpenFileDialog:**

- Nhấp đúp vào control  **OpenFileDialog**

### 3. **SaveFileDialog** ( **SaveFileDialog** )

#### a. **Công dụng:**

- Tạo ra hộp thoại Save File.

#### b. **Tạo SaveFileDialog:**

- Nhấp đúp vào control  **SaveFileDialog**

### 4. **PrintPreviewDialog** ( **PrintPreviewDialog** )

#### a. **Công dụng:**

- Xem trước khi in.

#### b. **Tạo SaveFileDialog:**

- Nhấp đúp vào control  **PrintPreviewDialog**

### 5. **PrintDialog** ( **PrintDialog** )

#### a. **Công dụng:**

- Tạo ra hộp thoại Print File.

#### b. **Tạo PrintDialog:**

- Nhấp đúp vào control  **PrintDialog**

### 6. **FolderBrowserDialog** ( **FolderBrowserDialog** )

#### a. **Công dụng:**

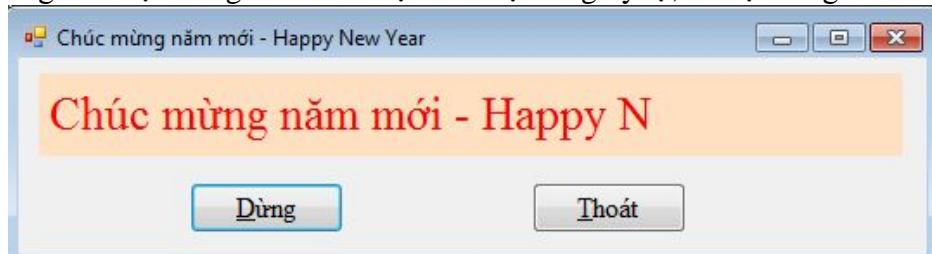
- Tạo ra hộp thoại Browser.

#### b. **Tạo FolderBrowserDialog:**

- Nhấp đúp vào control  **FolderBrowserDialog**

## Bài tập

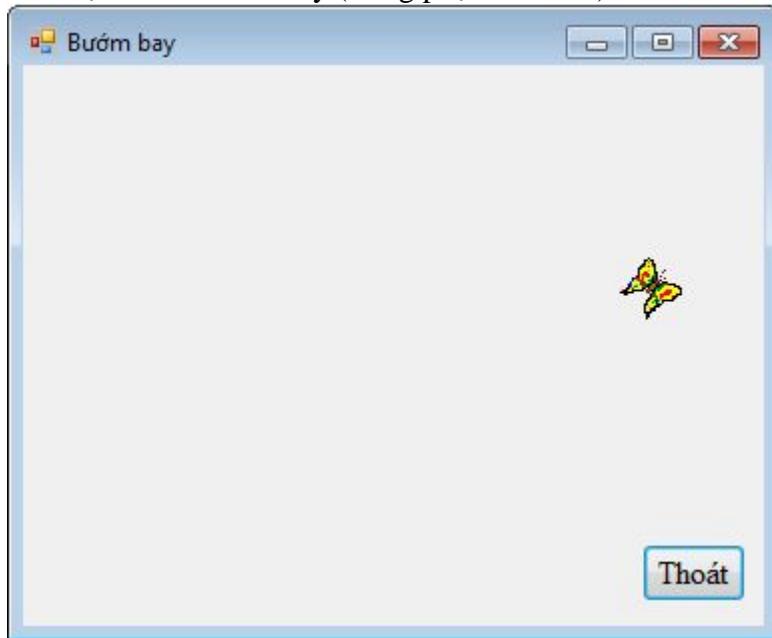
1. Viết chương trình tạo dòng chữ xuất hiện lần lượt từng ký tự, có nội dung như sau:



( lblChucMung, Timer1, btnDungChay, btnThoat)

- Button **Dừng/Chạy** có nhiệm vụ không cho / cho Timer hoạt động và đổi nội dung (Text) của button btnDungChay từ Dừng sang Chạy hoặc ngược lại.
- Button **Thoát** dùng để dừng chương trình.

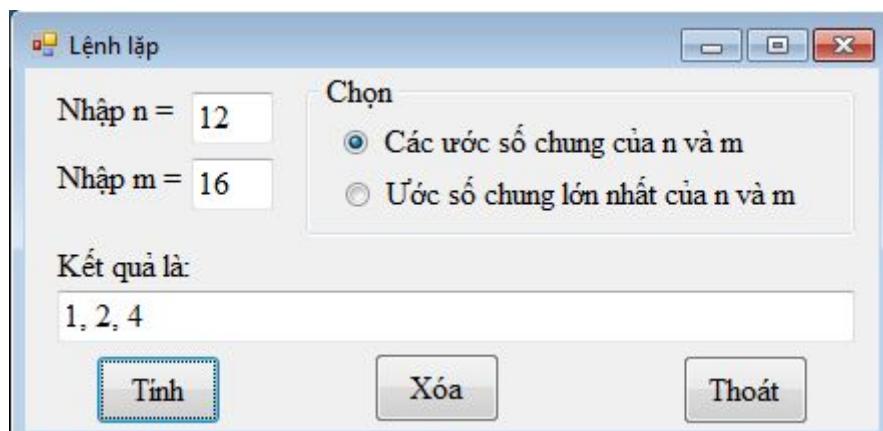
2. Viết chương trình tạo form Bướm bay (trong phạm vi form) như sau:



\* Hướng dẫn:

Sử dụng: timer, PictureBox, Button và 2 file hình BFClose.ico, BFOpen.ico

3. Viết chương trình nhập hai số nguyên dương n và m. Tính theo yêu cầu đã chọn như sau:



(txtN, txtM, rad1, rad2, txtKQ, btnTinh, btnXoa, btnThoat)

- Button **Tính** (btnTinh): tính toán và xuất kết quả ra TextBox txtKQ.
- Button **Xóa**: xóa trống tất cả các TextBox, đưa con trỏ vào ô txtN.
- Button **Thoát**: dừng chương trình.

4. Từ Ví Dụ 4.5, tạo và bổ sung thêm các công cụ CommonDialog để được ứng dụng dạng giao diện đơn văn bản (Single-Document Interface SDI).
5. Phát triển bài tập 4, để có thể mở cùng lúc nhiều cửa sổ - dạng giao diện đa văn bản (Multi-Document Interface MDI)

-- oOo --

## Chương 3: ARRAY - STRING - EXCEPTION

### Bài 5: ARRAY (MẢNG)

#### I. Mảng 1 chiều

##### 1. Định nghĩa

- Mảng là một tập hợp có thứ tự của *những đối tượng (objects)*, tất cả các đối tượng này có cùng một kiểu dữ liệu.
- Mảng trong ngôn ngữ C# sử dụng những phương thức và các thuộc tính. Thể hiện của mảng trong C# có thể truy cập những phương thức và các thuộc tính của **System.Array**.
- Một số các thuộc tính và phương thức của lớp **System.Array**:

Thành viên	Mô tả
Sort()	Phương thức sắp xếp giá trị tăng dần trong mảng một chiều
Reverse()	Phương thức sắp xếp giá trị giảm dần trong mảng một chiều
Length	Thuộc tính chiều dài của mảng
SetValue()	Phương thức thiết lập giá trị cho một thành phần xác định trong mảng

##### 2. Khai báo mảng:

<kiểu dữ liệu>[] <tên mảng>;

Ví dụ:

int[] myIntArray ;

Tạo thể hiện của mảng: sử dụng từ khóa new

Ví dụ:

myIntArray = new int[5] ;

##### 3. Khởi tạo thành phần của mảng

- Tạo thể hiện của mảng đồng thời với khởi tạo các giá trị:

+ Cách 1:

int[] myIntArray = new int[5] {2, 4, 6, 8, 10};

+ Cách 2:

int[] myIntArray = {2, 4, 6, 8, 10};

- Các khai báo trên sẽ thiết lập bên trong bộ nhớ một mảng chứa 5 số nguyên.

Chú ý: Không thể thiết lập lại kích thước cho mảng.

##### 4. Giá trị mặc định:

- Khi chúng ta tạo một mảng có kiểu dữ liệu giá trị, mỗi thành phần sẽ chứa giá trị mặc định của kiểu dữ liệu.

Ví dụ:

Với khai báo **int myIntArray = new int[5]** ; thì:

- Mỗi thành phần của mảng được thiết lập giá trị là 0 (giá trị mặc định của số nguyên).

- Những kiểu tham chiếu trong một mảng không được khởi tạo giá trị mặc định, chúng được khởi tạo giá trị null.

##### 5. Truy cập các thành phần trong mảng:

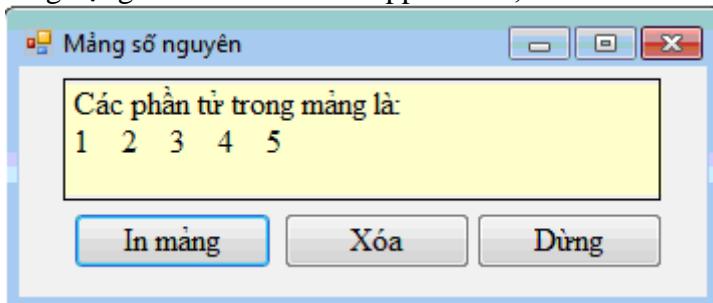
- Để truy cập vào thành phần trong mảng ta sử dụng toán tử chỉ mục ([]).
- Chỉ mục của thành phần đầu tiên trong mảng luôn luôn là 0.
- Thuộc tính Length của mảng cho biết số đối tượng trong một mảng.

## 6. Duyệt mảng 1 chiều:

```
for (int i=0; i < myIntArray.Length; i++) {
    xử lý myIntArray[i];
}
Có thể thay for bằng foreach như sau
foreach (int phantu in a){
    xử lý myIntArray[i];
}
```

### Ví dụ 5.1: (Mảng 5 số nguyên từ 1 đến 5)

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 5.1 như sau:



\* Yêu cầu

- Thiết kế form như mẫu (lblKQ, btnIn, btnXoa, btnDung).
- Khai báo mảng 1 chiều (a) gồm 5 số nguyên từ 1 đến 5.
- Nhấp vào button **In mảng** (btnIn) sẽ in ra label (lblKQ) các giá trị trong mảng.
- Nhấp vào button **Xóa** (btnXoa) sẽ xóa trống nội dung của label (lblKQ).
- Nhấp vào button **Dừng** sẽ dừng chương trình.

\* Hướng dẫn

- Thiết kế form như yêu cầu.
- Khai báo mảng: qua code, thêm đoạn code để được kết quả như sau:

```
public partial class Form1 : Form
{
    // Khai bao mang 1 chieu gom 5 so nguyen tu 1 den 5
    int[] a = { 1, 2, 3, 4, 5 };
}
```

- Nhấp đúp vào button **In mảng**, thêm đoạn code sau:

```
// Xuat cac phan tu trong mang ra man hinh
this.lblKQ.Text="Các phần tử trong mảng là:\n\r";
for (int i=0; i < a.Length; i++) {
    this.lblKQ.Text += a[i]+ "      ";
}
```

Có thể thay for bằng foreach như sau

```
foreach (int phantu in a){
    this.lblKQ.Text += a[i]+ "      ";
}
```

- Nhấp đúp vào button **Xóa**, thêm đoạn code sau:

```
this.lblKQ.Text = " ";
```

- Nhấp đúp vào button **Dừng**, thêm đoạn code sau:

```
Application.Exit();
```

### Ví dụ 5.2: (Mảng 5 số nguyên)

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 5.2 như sau:



\* Yêu cầu

- Thiết kế form: btnNhap, txtNhap, btnTang, btnGiam, lblKQ, btnIn, btnXoa, btnDung.
- Khai báo mảng 1 chiều (a) chứa 5 số nguyên, số phần tử hiện có (sopt) là 0.
- Nhập số vào TextBox txtNhap, nhấp vào button **Nhập 1 phần tử mảng** (btnNhap) cho phép đưa giá trị trong TextBox txtNhap vào mảng. Khi mảng đã đủ 5 số nguyên thì phải thông báo “Mảng đã đầy” và không cho nhập nữa.
  - Nhấp vào button **Sắp tăng** (btnTang) sẽ sắp xếp mảng theo thứ tự tăng dần.
  - Nhấp vào button **Sắp giảm** (btnGiam) sẽ sắp xếp mảng theo thứ tự giảm dần.
  - Nhấp vào button **In mảng** (btnIn) sẽ in ra label (lblKQ) các giá trị trong mảng.
  - Nhấp vào button **Xóa** (btnXoa) sẽ xóa trống nội dung của label (lblKQ) đồng thời khai báo lại số phần tử hiện có (sopt) của mảng là 0.
  - Nhấp vào button **Dừng** sẽ dừng chương trình.

\* Hướng dẫn

- Thiết kế form như yêu cầu.
- Khai báo mảng như sau:

```
public partial class Form1 : Form
{
    // Khai bao mang 1 chieu gom 5 so nguyen
    int[] a = new int[5];
    // Khai bao so phan tu hien co cua mang
    int sopt = 0;
}
```

- Nhấp đúp vào button **Nhập 1 phần tử mảng**, thêm đoạn code sau:

```
// Nhập mot phan tu cho mang
if (sopt == 5)
    MessageBox.Show("Mảng đã đầy!");
else
{
    a[sopt] = Convert.ToInt32(this.txtNhap.Text);
    sopt++;
    this.txtNhap.ResetText();
    this.txtNhap.Focus();
}
```

- Nhấp đúp vào button **Sắp tăng**, thêm đoạn code sau:

```
// Sap xep mang giam
if (sopt == 0)
    this.lblKQ.Text = "Mảng rỗng!";
```

```
        else
            Array.Sort(a, 0, sopt);
        this.lblKQ.Text = "Đã sắp xếp mảng tăng dần!";
    - Nhấp đúp vào button Sắp giảm, thêm đoạn code sau:
        // Sap xep mang giam
        if (sopt == 0)
            this.lblKQ.Text = "Mảng rỗng!";
        else
            Array.Reverse(a, 0, sopt);
        this.lblKQ.Text = "Đã sắp xếp mảng giảm dần!";
    - Nhấp đúp vào button Xóa, thêm đoạn code sau:
        this.lblKQ.Text = " ";
        sopt = 0;
        this.txtNhap.Focus();
    - Nhấp đúp vào button In mảng, thêm đoạn code sau:
        // Xuat cac phan tu trong mang ra man hinh
        if (sopt==0)
            this.lblKQ.Text = "Mảng rỗng!";
        else
        {
            this.lblKQ.Text="Các phần tử trong mảng là:\n\r";
            for (int i = 0; i < sopt; i++)
                this.lblKQ.Text += a[i] + "      ";
        }
    - Nhấp đúp vào button Dừng, thêm đoạn code sau:
        Application.Exit();
```

## **II. Mảng nhiều chiều**

### **1. Định nghĩa**

- Mảng đa chiều là mảng mà mỗi thành phần là một mảng khác.
- Ngôn ngữ C# hỗ trợ hai kiểu mảng đa chiều là:
  - + Mảng đa chiều cùng kích thước.
  - + Mảng đa chiều khác kích thước.
- Trong phạm vi bài học này, ta chỉ khảo sát mảng 2 chiều mà thôi.

### **2. Khai báo mảng 2 chiều**

<kiểu dữ liệu>[ , ] <tên mảng>

Ví dụ:

```
int[ , ] myRectangularArray ;
```

### **3. Khởi tạo thành phần của mảng**

```
int[] myRectangularArray = new int[sodong , socot] ;
```

### **4. Duyệt mảng 2 chiều**

```
for (int i = 0; i < sodong; i++)
{
    for (int j = 0; j < socot; j++)
    {
```

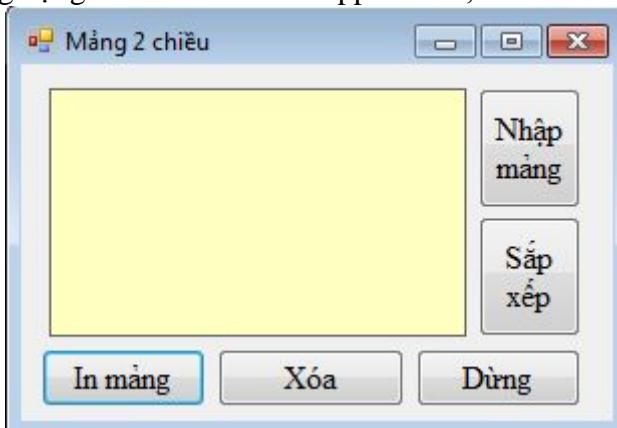
```

        Xử lý myRectangularArray[i,j];
    }
}

```

### Ví dụ 5.3:

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 5.3 như sau:



\* Yêu cầu

- Thiết kế form như mẫu (lblKQ, btnIn, btnXoa, btnDung, btnNhap, btnSapXep).
- Khai báo mảng 2 chiều gồm 4 dòng, 3 cột chứa các số nguyên.
- Nhấp vào button **Nhập mảng** để nhập các phần tử cho mảng (có giá trị = dòng - cột).
- Nhấp vào button **Sắp Xếp** sẽ sắp xếp mảng tăng dần theo từng hàng.
- Nhấp vào button **In mảng** (btnIn) sẽ in ra label (lblKQ) các giá trị trong mảng.
- Nhấp vào button **Xóa** (btnXoa) sẽ xóa trống nội dung của label (lblKQ).
- Nhấp vào button **Dừng** sẽ dừng chương trình.

\* Hướng dẫn

- Thiết kế form như yêu cầu.
- Khai báo mảng: qua code, thêm đoạn code để được kết quả như sau:

```

public partial class Form1 : Form
{
    // Khai bao 4 dong 3 cot
    const int sodong = 4;
    const int socot = 3;
    // Khai bao mang 2 chieu gom 4 dong, 3 cot chua 12 so
    // nguyen
    int[,] Array2 = new int[sodong,socot];
}

```

- Nhấp đúp vào button **Nhập mảng**, thêm đoạn code sau:

```

// Nhap cac phan tu cho mang
for (int i=0; i < sodong; i++)
    for (int j=0; j< socot; j++)
        Array2[i,j] = i-j;

```

- Nhấp đúp vào button **Sắp xếp**, thêm đoạn code sau:

```

// Sap xep mang
int[] t = new int[sodong * socot];
for (int i = 0; i < sodong; i++)
{
    for (int j = 0; j < socot; j++)
        t[j] = Array2[i, j];
    Array.Sort(t, 0, socot);
}

```

```
        for (int j = 0; j < socot; j++)
            Array2[i, j]=t[j];
    }
```

- Nhấp đúp vào button **In mảng**, thêm đoạn code sau:

```
// Xuat cac phan tu trong mang ra man hinh
this.lblKQ.Text = "Các phần tử trong mảng là:\n\r";
for (int i = 0; i < sodong; i++)
{
    for (int j = 0; j < socot; j++)
    {
        this.lblKQ.Text += Array2[i, j] + "      ";
    }
    this.lblKQ.Text += "\n\r";
}
```

- Nhấp đúp vào button **Xóa**, thêm đoạn code sau:

```
this.lblKQ.Text = " ";
```

- Nhấp đúp vào button **Dừng**, thêm đoạn code sau:

```
Application.Exit();
```

## Bài tập

Khởi tạo một ứng dụng Windows Forms Application:

1. Khai báo 1 mảng nguyên 1 chiều tối đa 10 phần tử. Viết chương trình:
  - Nhập vào giá trị cho 1 phần tử trong mảng.
  - In giá trị của các phần tử trong mảng.
  - In giá trị lớn nhất, giá trị nhỏ nhất của các phần tử trong mảng.
  - In tổng số các giá trị, trung bình cộng các giá trị của các phần tử trong mảng.
2. Khai báo 1 mảng nguyên 2 chiều 4 dòng, 5 cột. Viết chương trình:
  - Nhập giá trị cho các phần tử trong mảng (giá trị = số thứ tự dòng + số thứ tự cột).
  - In giá trị các phần tử trong mảng.
  - In giá trị lớn nhất, giá trị nhỏ nhất của các phần tử trong mảng.
  - In tổng số các giá trị, trung bình cộng các giá trị của các phần tử trong mảng.

-- oOo --

## Bài 6: (tiếp theo) STRING (CHUỖI) – EXCEPTION (NGOẠI LỆ)

### III. String (Chuỗi)

#### 1. Tạo một chuỗi

##### a. Chuỗi hằng

```
string TenChuoi = "Chuỗi" ;
```

Ví dụ:

```
string thongbao = "Đây là một câu thông báo." ;
```

Chú ý: Ta có 2 khai báo chuỗi sau là như nhau

```
string chuoit = "Dong mot \n Dong hai";
```

```
string chuoit = @"Dong mot
```

```
Dong hai";
```

##### b. Chuỗi dùng phương thức ToString

Ví dụ:

```
int myInt = 9 ;
```

```
string intString = myInt.ToString();
```

#### 2. Thao tác trên chuỗi

Lớp string cung cấp rất nhiều các phương thức để so sánh, tìm kiếm, thay thế ...; các phương thức này được trình bày trong bảng sau:

Phương thức	Ý nghĩa
Compare()	So sánh hai chuỗi (Chuỗi 1 ? Chuỗi 2) = (-1 ; 0 ; 1) tương ứng (<, =, >)
Concat()	Nối chuỗi
EndsWith()	Xem chuỗi có kết thúc bằng một nhóm ký tự xác định hay không.
IndexOf()	Chỉ ra vị trí xuất hiện đầu tiên của một chuỗi con trong chuỗi lớn.
Insert()	Trả về một chuỗi mới đã được chèn thêm.
LastIndexOf()	Chỉ ra vị trí xuất hiện cuối cùng của một chuỗi con trong chuỗi lớn.
Length	Chiều dài của chuỗi.
Remove()	Xoá đi một chuỗi con.
Replace()	Thay thế chuỗi cũ bằng chuỗi mới.
Split()	Trả về chuỗi con được phân định bởi ký tự xác định.
StartsWith()	Xem chuỗi có bắt đầu bằng một nhóm ký tự xác định hay không.
Substring()	Lấy chuỗi con.
ToLower()	Trả về bản sao của chuỗi ở kiểu chữ thường.
ToUpper()	Trả về bản sao của chuỗi ở kiểu chữ IN HOA.

#### 3. Ví dụ 6.1

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 6.1 như sau:

\* Yêu cầu

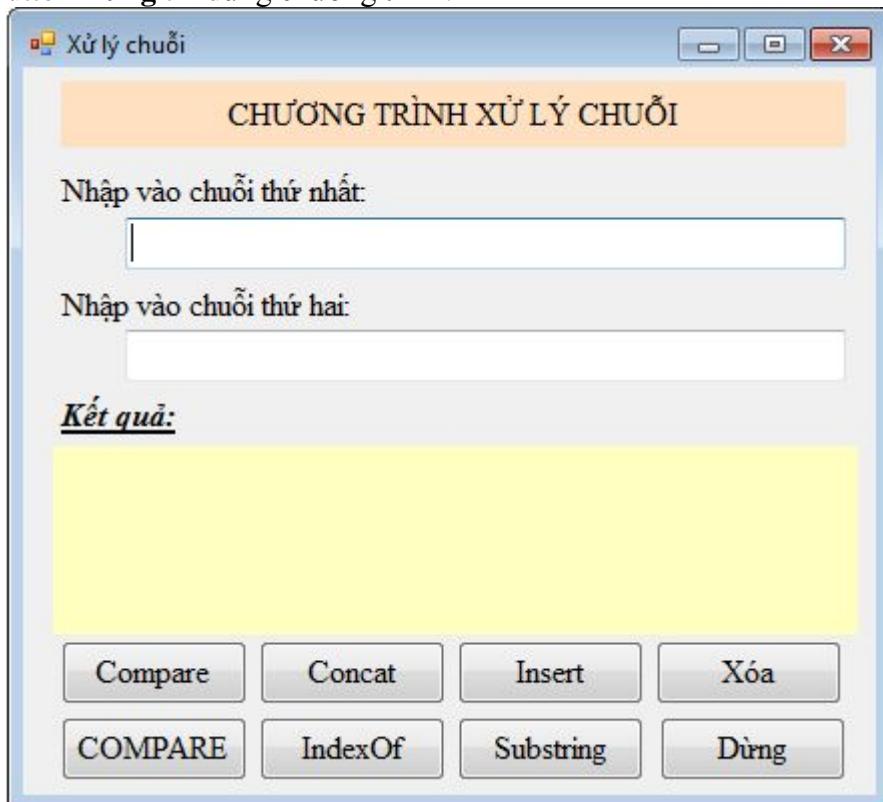
- Thiết kế form gồm: lblTieuDe, lbl1, txtS1, lbl2, txtS2, lbl3, lblKQ, và các button (xem hình).

- Nhấp vào button Compare: so sánh 2 chuỗi txtS1 và txtS2 (có phân biệt chữ HOA và chữ thường), kết quả xuất trong lblKQ.

- Nhấp vào button COMPARE: so sánh 2 chuỗi txtS1 và txtS2 (không phân biệt chữ HOA và chữ thường), kết quả xuất trong lblKQ.

- Nhấp vào button Concat, nối 2 chuỗi txtS1 và txtS2, kết quả xuất trong lblKQ.

- Nhập button **IndexOf**, cho biết vị trí xuất hiện của chuỗi txtS2 trong chuỗi txtS1. Nếu có txtS2 trong txtS1 thì thay thế txtS2 (trong txtS1) bằng chuỗi "CHỖ NÀY". Xuất kết quả trong lblKQ.
- Nhập vào button **Insert**, chèn chuỗi txtS2 vào sau từ đầu tiên của chuỗi txtS1; và chèn chuỗi txtS2 vào trước từ sau cùng của chuỗi txtS1. Xuất kết quả trong lblKQ.
- Nhập vào button **Substring**, cho biết vị trí xuất hiện của chuỗi "TRÌNH XỬ LÝ" trong lblTieuDe. Nếu có thì xóa chuỗi ra khỏi lblTieuDe. Xuất kết quả trong lblKQ.
- Nhập button **Xóa** thì xóa trống: TextBox txtS1, TextBox txtS2, Label lblKQ đồng thời đưa con trỏ vào TextBox txtS1.
- Nhập button **Dừng** thì dừng chương trình.



\* Hướng dẫn

- Thiết kế form như yêu cầu.
- Nhấp đúp vào button **Compare**, thêm vào đoạn code:

```
string s1=this.txtS1.Text;
string s2=this.txtS2.Text;
// So sánh hai chuỗi với nhau có phân biệt chữ thường và
// chữ hoa
int kq = string.Compare(s1,s2);
this.lblKQ.Text = "txtS1 ";
if (kq == -1)
    this.lblKQ.Text += "<";
else if (kq == 0)
    this.lblKQ.Text += "=";
else
    this.lblKQ.Text += ">";
this.lblKQ.Text += " txtS2";
```

- Nhấp đúp vào button **COMPARE**, thêm vào đoạn code:

```
string s1=this.txtS1.Text;
string s2=this.txtS2.Text;
// So sánh hai chuỗi với nhau không phân biệt chữ thường
và chữ hoa
int kq = string.Compare(s1,s2, true);
this.lblKQ.Text = "txtS1 ";
if (kq == -1)
    this.lblKQ.Text += "<";
else if (kq == 0)
    this.lblKQ.Text += "=";
else
    this.lblKQ.Text += ">";
this.lblKQ.Text += " txtS2";
```

- Nhấp đúp vào button **Concat**, thêm vào đoạn code:

```
string s1 = this.txtS1.Text;
string s2 = this.txtS2.Text;
// Nối chuỗi
this.lblKQ.Text = string.Concat(s1,s2);
```

- Nhấp đúp vào button **IndexOf**, thêm vào đoạn code:

```
string s1 = this.txtS1.Text;
string s2 = this.txtS2.Text;
// Chỉ ra vị trí xuất hiện của chuỗi 2 trong chuỗi 1
if (s1.IndexOf(s2) >= 0)
{
    this.lblKQ.Text = "txtS2 xuất hiện trong txtS1 tại
vị trí ";
    this.lblKQ.Text += s1.IndexOf(s2);
    this.lblKQ.Text += ".!";
    // Thay thế chuỗi s="CHỖ NÀY" vào vị trí chuỗi 2
    trong chuỗi 1
    string s = "CHỖ NÀY";
    this.lblKQ.Text += "\n\rThay thế txtS2 trong txtS1
    bằng chuỗi CHỖ NÀY,";
    this.lblKQ.Text+= "\n\rKết quả:" +s1.Replace(s2, s);
}
else this.lblKQ.Text = "txtS2 không xuất hiện trong
txtS1!";
```

- Nhấp đúp vào button **Insert**, thêm vào đoạn code:

```
string s1 = this.txtS1.Text;
string s2 = this.txtS2.Text;
// Chèn chuỗi 2 vào sau từ đầu tiên của chuỗi 1
this.lblKQ.Text = "Chèn txtS2 vào sau từ đầu tiên của
txtS1:\n\r";
this.lblKQ.Text += s1.Insert(s1.IndexOf(" "), s2);
// Chèn chuỗi 2 vào trước từ cuối cùng của chuỗi 1
this.lblKQ.Text += "\n\rChèn txtS2 vào trước từ cuối cùng
của txtS1:\n\r ";
this.lblKQ.Text += s1.Insert(s1.LastIndexOf(" "), s2);
```

- Nhấp đúp vào button **Substring**, thêm vào đoạn code:

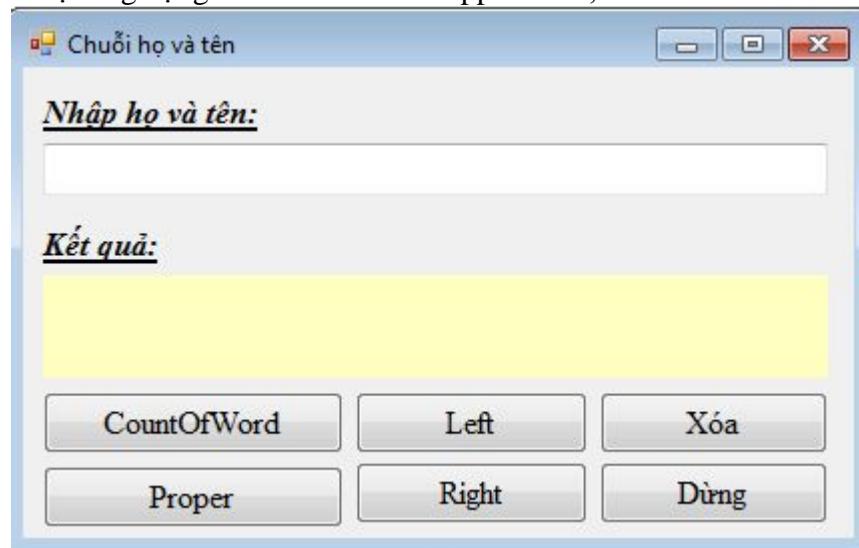
```

// Chỉ ra vị trí xuất hiện của s="TRÌNH XỬ LÝ" trong
lblTieuDe
string s="TRÌNH XỬ LÝ";
int ix;
ix = this.lblTieuDe.Text.IndexOf(s);
// Trích s từ lblTieuDe và in vào lblKQ
if (ix >= 0)
{
    this.lblKQ.Text = this.lblTieuDe.Text.Substring(ix,
s.Length);
    this.lblKQ.Text += " xuất hiện trong tiêu đề tại vị
trí ";
    this.lblKQ.Text += ix;
    // Xóa s ra khỏi lblTieuDe
    this.lblKQ.Text += "\n\rTiêu đề sau khi xóa " + s;
    this.lblKQ.Text += "\n\rKết quả là: ";
    this.lblKQ.Text += this.lblTieuDe.Text.Remove(ix,
s.Length);
}
else this.lblKQ.Text = "Không xuất hiện trong chuỗi 1!";
- Nhấp đúp vào button Xóa, thêm vào đoạn code:
this.txtS1.ResetText();
this.txtS2.ResetText();
this.lblKQ.Text = "";
this.txtS1.Focus();
- Nhấp đúp vào button Dừng, thêm vào đoạn code:
Application.Exit();

```

#### 4. Ví dụ 6.2

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 6.2 như sau:



\* Yêu cầu

- Thiết kế form gồm: lbl1, txtHoTen, lbl2, lblKQ, và các button (xem hình).

- Nhấp vào button **CountOfWord**: đếm số từ trong chuỗi txtHoTen, kết quả xuất trong lblKQ.

- Nhấp vào button **Proper**: đổi thành chữ Hoa Đầu Từ cho chuỗi txtHoTen, kết quả xuất trong lblKQ.

- Nhấp vào button **Left**, lấy ra từ bên trái của chuỗi txtHoTen, kết quả xuất trong lblKQ.

- Nhấp vào button **Right**, lấy ra từ bên phải của chuỗi txtHoTen, kết quả xuất trong lblKQ.

- Nhấp button **Xóa** thì xóa trống: TextBox txtS1, TextBox txtS2, Label lblKQ đồng thời đưa con trỏ vào TextBox txtS1.

- Nhấp button **Dừng** thì dừng chương trình.

\* Hướng dẫn

- Thiết kế form như yêu cầu.

- Nhấp đúp vào button **CountOfWord**, thêm vào đoạn code:

```
// Đếm số từ trong chuỗi
// Tạo ra hàng ký tự khoảng trắng
const char Space = ' ';
// Gán giá trị cho chuỗi
string s = hoten.Trim();
// Thực hiện việc chia chuỗi vào mảng --> Đếm từ = độ dài
mảng
int cWord = s.Split(Space).Length;
// Gởi kết quả trả về
return cWord;
```

\*\* **Tham khảo thêm đoạn code sau:**

```
// Đếm số từ trong chuỗi txtHoTen
int count=0;
string s=this.txtHoTen.Text.Trim();
for (int i = 0; i < s.Length-1; i++)
    if ((s.Substring(i, 1)==" ") && (s.Substring(i + 1,
1)!= " "))
        count++;
if (s.Length > 0) count++;
this.lblKQ.Text = "Số từ là: " + count;
```

- Nhấp đúp vào button **Proper**, thêm vào đoạn code:

```
// Đổi thành chữ Hoa Đầu Từ trong chuỗi txtHoTen
string s = this.txtHoTen.Text.Trim();
if (s.Length == 0)
    this.lblKQ.Text = "Chuỗi rỗng!";
else
{
    this.lblKQ.Text = "Chuỗi kết quả là: ";
    this.lblKQ.Text += s.Substring(0,1).ToUpper();
    for (int i = 1; i < s.Length; i++)
    {
        if ((s[i-1].ToString() == " "
)&&(s[i].ToString() != " "))
        {
            string ss = s[i].ToString();
            this.lblKQ.Text += ss.ToUpper();
```

```

        }
        else this.lblKQ.Text += s[i].ToString();
    }
}

```

- Nhấp đúp vào button **Left**, thêm vào đoạn code:

```

// Từ đầu tiên của chuỗi txtHoTen
string s = this.txtHoTen.Text.Trim();
if (s.Length == 0)
    this.lblKQ.Text = "Chuỗi rỗng!";
else
{
    this.lblKQ.Text = "Từ đầu tiên của chuỗi là: ";
    this.lblKQ.Text += s.Substring(0, s.IndexOf(" "));
}

```

- Nhấp đúp vào button **Right**, thêm vào đoạn code:

```

// Từ cuối cùng của chuỗi txtHoTen
string s = this.txtHoTen.Text.Trim();
if (s.Length == 0)
    this.lblKQ.Text = "Chuỗi rỗng!";
else
{
    this.lblKQ.Text = "Từ đầu tiên của chuỗi là: ";
    this.lblKQ.Text += s.Substring(s.LastIndexOf(" ") + 1,
        s.Length - s.LastIndexOf(" ") - 1);
}

```

- Nhấp đúp vào button **Xóa**, thêm vào đoạn code:

```

this.txtHoTen.ResetText();
this.lblKQ.Text = "";
this.txtHoTen.Focus();

```

- Nhấp đúp vào button **Dừng**, thêm vào đoạn code:

```
Application.Exit();
```

## IV. Exception (Ngoại lệ)

### 1. Khái niệm

- Exception có thể được hiểu là bắt giữ lỗi với những đoạn mã hợp lệ để không tổn hại đến chương trình.

- Lỗi có thể do nguyên nhân từ chính người sử dụng; hoặc có thể do những vấn đề không mong đợi khác như: thiếu bộ nhớ, thiếu tài nguyên hệ thống ....

- Một trình xử lý ngoại lệ là một khôi lệnh chương trình được thiết kế xử lý các ngoại lệ mà chương trình phát sinh.

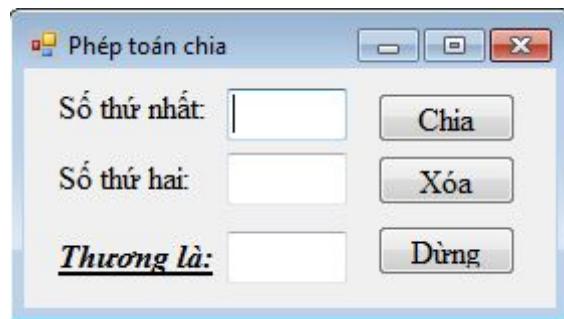
- Xử lý ngoại lệ được thực thi trong câu lệnh **catch**.

- Các câu lệnh có khả năng xảy ra ngoại lệ thực thi trong câu lệnh **try**.

\* Một cách lý tưởng, nếu một ngoại lệ được bắt và được xử lý thì chương trình có thể sửa chữa được vấn đề bị lỗi và tiếp tục thực hiện hoạt động. Thậm chí nếu chương trình không tiếp tục, bằng việc bắt giữ ngoại lệ chúng ta cũng có cơ hội để in ra những thông điệp có ý nghĩa và kết thúc chương trình một cách rõ ràng.

### 2. Ví dụ 6.3

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 6.3 như sau:



\* Yêu cầu

- Thiết kế form gồm: lbl1, txtSo1, lbl2, txtSo2, lbl3, txtKQ và các button (xem hình).

- Nhập số vào 2 TextBox txtSo1 và txtSo2.

- Nhấp vào button **Chia**:

+ Nếu txtSo1, txtSo2 không phải là số; hoặc nhập vào txtSo2 là 0 thì báo lỗi: "Lỗi rồi!".

+ Nếu txtSo1, txtSo2 là số thì xuất kết quả là txtSo1 / txtSo2 vào TextBox txtKQ.

- Nhấp button **Xóa** thì xóa trắng: TextBox txtS1, TextBox txtS2, Label lblKQ đồng thời đưa con trỏ vào TextBox txtS1.

- Nhấp buuton **Dừng** thì dừng chương trình.

\* Hướng dẫn

- Thiết kế form như yêu cầu.

- Nhấp đúp vào button **Chia**, thêm vào đoạn code:

```
// Xóa trắng TextBox txtKQ
txtKQ.ResetText();
// Đoạn code có xảy ra ngoại lệ khi thực hiện
try
{
    int so1 = int.Parse(this.txtSo1.Text);
    int so2 = int.Parse(this.txtSo2.Text);
    this.txtKQ.Text += (float)so1 / so2;
}
// Xử lý ngoại lệ
catch (Exception ex)
{
    this.txtKQ.Text = "Lỗi rồi!";
}
```

## Bài tập

1. Viết chương trình nhập vào một chuỗi họ và tên. In ra:

- Độ dài và số từ của chuỗi họ tên.

- Chuỗi họ và tên dưới dạng chữ thường.

- Chuỗi họ và tên dưới dạng chữ IN HOA.

- Chuỗi họ và tên dưới dạng Hoa Đầu Tù.

- Chuỗi họ và tên đã được loại bỏ các khoảng trắng thừa (đầu chuỗi, cuối chuỗi, **bên trong chuỗi**).

2. Viết chương trình giải phương trình bậc 1:  $bx + c = 0$

Lưu ý: có xử lý trường hợp nhập vào b, c không phải là số.

-- oOo --

## Chương 4: CLASS – OBJECT – METHOD

### Bài 7: CLASS (LỚP) – OBJECT (ĐỐI TƯỢNG) – METHOD (PHƯƠNG THỨC)

#### I. Khái niệm

- Kiểu dữ liệu trong C# được định nghĩa là một lớp (class).
- Thể hiện riêng của từng lớp được gọi là đối tượng (object).
- Hai thành phần chính cấu thành một lớp (class) là thuộc tính / tính chất và phương thức (method) / hành động ứng xử của đối tượng.

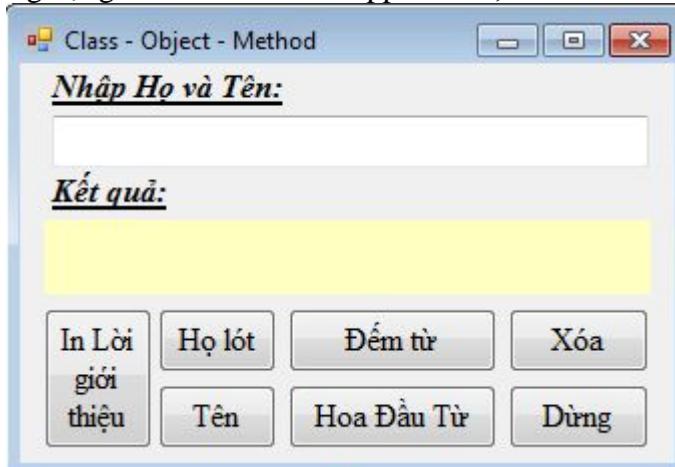
#### II. Định nghĩa lớp (class)

##### 1. Cú pháp

[Thuộc tính] [Bổ sung truy cập] class <Tên lớp> [: Lớp cơ sở]  
 {  
     // Các thuộc tính  
     <Thuộc tính>  
     // Các phương thức  
     <Phương thức>  
 }

##### 2. Ví dụ 7.1

\* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 7.1 như sau:



\* Yêu cầu

- Thiết kế form gồm: lbl1, txtHoTen, lbl2, lblKQ, và các button (xem hình).
- Tạo class Chuoi như sau:

```
public class Chuoi
{
    // Thuộc tính
    ...
    // Phương thức
    ...
}
```

+ Trong phần thuộc tính, khai báo:

string tenchuongtrinh = "Chương trình xử lý họ và tên!" ;  
+ Trong phần phương thức, khai báo các phương thức sau:

```
// Phương thức
public string In()
{
    // In tên chương trình
    return tenchuongtrinh;
}

public string Ten(string hoten)
{
    // Lấy tên
    int lio=hoten.LastIndexOf(" ");
    return hoten.Substring(lio+1, hoten.Length-lio-1);
}

public string HoLot(string hoten)
{
    // Lấy họ và lót
    int lio = hoten.LastIndexOf(" ");
    return hoten.Substring(0,lio);
}

public int CountOfWord(string hoten)
{
    // Đếm số từ trong chuỗi
    // Tạo ra hằng ký tự khoảng trắng
    const char Space = ' ';
    // Gán giá trị cho chuỗi
    string s = hoten.Trim();
    // Thực hiện việc chia chuỗi thành mảng --> Đếm từ
    cWord = s.Split(Space)).Length ;
    // Gởi kết quả trả về
    return cWord;
}
```

\*\* Tham khảo thêm đoạn code sau:

```
// Đếm số từ trong chuỗi HoTen
int count = 0;
string s = hoten.Trim();
for (int i = 0; i < s.Length - 1; i++)
    if ((s.Substring(i,1)== " ")&&(s.Substring(i+1,1) != " "))
        count++;
if (s.Length > 0)
    count++;
return count;

public string Proper(string hoten)
{
```

```

// Đổi thành chữ Hoa Đầu Từ trong chuỗi txtHoTen
// Tạo ra hằng ký tự khoảng trắng
const char Space = ' ';
// Gán giá trị cho chuỗi
string s = hoten.Trim();
// Đổi chuỗi
string kq = "Chuỗi rỗng!";
if (s.Length == 0)
    return kq;
else
{
    kq = " ";
    string [] s1 = s.Split(Space);
    foreach (string tu in s1) {
        string ss = tu[0].ToString();
        kq += ss.ToUpper();
        kq += tu.Substring(1,tu.Length-1);
        kq += " ";
    }
    return kq.Trim();
}

```

**\*\* Tham khảo thêm đoạn code sau:**

```

// Đổi thành chữ Hoa Đầu Từ trong chuỗi txtHoTen
string s = hoten.Trim();
string kq = "Chuỗi rỗng!";
if (s.Length > 0)
{
    kq = s.Substring(0, 1).ToUpper();
    for (int i = 1; i < s.Length; i++)
    {
        if ((s[i - 1].ToString() == " ") &&
            (s[i].ToString() != " "))
        {
            string ss = s[i].ToString();
            kq += ss.ToUpper();
        }
        else kq += s[i].ToString();
    }
}
return kq;

```

- Sử dụng các phương thức trên để thực hiện các việc sau đây (xuất kết quả trong lblKQ):
  - + Nhập vào button **In Lời giới thiệu**, sẽ in lời giới thiệu.
  - + Nhập button **Họ Lót**, tách lấy họ lót của chuỗi trong txtHoTen.
  - + Nhập button **Tên**, tách lấy tên của chuỗi trong txtHoTen.
  - + Nhập button **Đếm từ**, đếm số từ của chuỗi trong txtHoTen.
  - + Nhập button **Hoa Đầu Từ**, đổi thành chuỗi Hoa Đầu Từ của chuỗi trong txtHoTen.

- Nhấp button **Xóa** thì xóa trống: TextBox txtHoTen, Label lblKQ đồng thời đưa con trỏ vào TextBox txtHoTen.

- Nhấp button **Dừng** thì dừng chương trình.

\* Hướng dẫn

- Thiết kế form như yêu cầu.

- Khai báo class: qua code, thêm đoạn code để được kết quả như sau

```
public Form1()
{
    InitializeComponent();
}
public class Chuoi
{
    // Thuộc tính
    ... (1)
    // Phương thức
    ... (2)
}
```

+ Trong phần (1), khai báo thuộc tính (xem đề bài)

+ Trong phần (2), khai báo phương thức (xem đề bài)

- Nhấp đúp vào button **In Lời giới thiệu**, thêm vào đoạn code:

```
Chuoi s = new Chuoi();
this.lblKQ.Text = s.In();
```

- Nhấp đúp vào button **Hẹ lót**, thêm vào đoạn code:

```
Chuoi s = new Chuoi();
this.lblKQ.Text = "Hẹ lót: " + s.HoLot(this.txtHoTen.Text);
```

- Nhấp đúp vào button **Tên**, thêm vào đoạn code:

```
Chuoi s = new Chuoi();
this.lblKQ.Text = "Tên là: " + s.Ten(this.txtHoTen.Text);
```

- Nhấp đúp vào button **Đếm từ**, thêm vào đoạn code:

```
Chuoi s = new Chuoi();
this.lblKQ.Text = "Tổng số từ là: " ;
this.lblKQ.Text += s.CountOfWord(this.txtHoTen.Text);
```

- Nhấp đúp vào button **Hoa Đầu Từ**, thêm vào đoạn code:

```
Chuoi s = new Chuoi();
this.lblKQ.Text = "Kết quả là: " ;
this.lblKQ.Text += s.Proper(this.txtHoTen.Text);
```

- Nhấp đúp vào button **Xóa**, thêm vào đoạn code:

```
this.lblKQ.Text = " ";
this.txtHoTen.ResetText();
this.txtHoTen.Focus();
```

- Nhấp đúp vào button **Dừng**, thêm vào đoạn code:

```
Application.Exit();
```

\* **Bổ sung**

- Nút button **In Hoa**: đổi thành chuỗi IN HOA của chuỗi trong txtHoTen .

- Nút button **In Thường**: đổi thành chuỗi in thường của chuỗi trong txtHoTen.

### **III. Properties - Method**

#### **1. Thuộc tính (Properties):**

Thuộc tính là những thông tin có thể thay đổi được.

## 2. Thuộc tính truy cập

Thuộc tính	Giới hạn truy cập
public	Không hạn chế. Những thành viên được đánh dấu public có thể được dùng bất kỳ các phương thức của lớp, bao gồm cả những lớp khác.
private	Thành viên trong lớp được đánh dấu private chỉ được dùng các phương thức của lớp này mà thôi.
Protected	Thành viên trong lớp được đánh dấu protected chỉ được dùng các phương thức của lớp này; và các phương thức của lớp dẫn xuất từ lớp này.
Internal	Thành viên trong lớp được đánh dấu là internal được dùng các phương thức của bất kỳ lớp nào cùng khôi hợp ngữ với lớp này.
protected internal	Thành viên trong lớp được đánh dấu là protected internal được dùng các phương thức của lớp này; các phương thức của lớp dẫn xuất từ lớp này; và các phương thức của bất kỳ lớp nào trong cùng khôi hợp ngữ với lớp này.

## 3. Phương thức (Method)

- Phương thức (method) chính là các hàm (function) được tạo trong lớp (class).
- Tên của phương thức thường được đặt theo tên của hành động.

## 4. Tham số của phương thức

### a) Khái niệm:

- Các tham số sau tên phương thức và được bọc bên trong dấu ngoặc tròn () .
- Mỗi tham số phải khai báo kèm theo kiểu dữ liệu.
- Trong C# có 2 dạng truyền tham số:
  - + Truyền tham chiếu: dùng thêm từ khóa **ref**.
  - + Truyền tham trị

### b) Ví dụ:

- \* Truyền tham số cho phương thức theo kiểu tham chiếu

```
public class Hoandoi
{
    public void HoanVi(ref int a, ref int b)
    {
        int c = a ;
        a = b ;
        b = c ;
    }
}
```

Khi đó: khi gọi hàm HoanVi ta phải truyền tham số dưới dạng tham chiếu như sau:

```
HoanDoi s = new HoanDoi();
s.HoanVi(ref a, ref b);
```

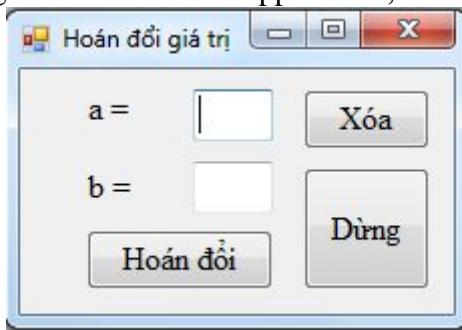
- \* Truyền tham số cho phương thức theo kiểu tham trị

```
public class HoanDoi
{
    public void HoanVi(int a, int b)
```

```
{  
    int c = a ;  
    a = b ;  
    b = c ;  
}
```

### 5. Ví dụ 7.2

- \* Khởi tạo một ứng dụng Windows Forms Application, lưu với tên là Vi Du 7.2 như sau:



#### \* Yêu cầu

- Thiết kế form gồm: lbl1, txta, lbl2, txtb, và các button (xem hình).
- Tạo các class HoanDoi, trong class có hàm HoanVi cho phép hoán vị 2 giá trị số nguyên.

- Nhấp button **Hoán Đổi** sẽ hoán đổi 2 giá trị trong txta và txtb.

- Nhấp button **Xóa** sẽ xóa trống 2 TextBox và đưa con trỏ vào ô txta.

- Nhấp button **Dùng** sẽ dừng chương trình.

#### \* Hướng dẫn

- Thiết kế form như yêu cầu.

- Khai báo class: qua code, thêm đoạn code để được kết quả như sau

```
public Form1()  
{  
    InitializeComponent();  
}  
  
public class HoanDoi  
{  
    public void HoanVi(ref int a, ref int b)  
    {  
        int c = a ;  
        a = b ;  
        b = c ;  
    }  
}
```

- Nhấp đúp vào button **Hoán đổi**, thêm vào đoạn code:

```
int a = int.Parse(this.txta.Text);  
int b = int.Parse(this.txtb.Text);  
HoanDoi s = new HoanDoi();  
s.HoanVi(ref a, ref b);  
this.txta.Text = a.ToString();  
this.txtb.Text = b.ToString();
```

- Nhấp đúp vào button **Xóa**, thêm vào đoạn code:

```
this.txta.ResetText();  
this.txtb.ResetText();  
this.txta.Focus();
```

- Nhấp đúp vào button **Dừng**, thêm vào đoạn code:

```
Application.Exit();
```

## Bài tập

Khởi tạo một ứng dụng Windows Forms Application:

1. Tạo một class có tên là BAI\_TAP\_CLASS, có các hàm:

- Hàm **TEN** nhận vào một chuỗi chỉ họ và tên, giá trị trả lại của hàm là chuỗi chỉ tên.
- Hàm **NGTO** nhận vào một số nguyên n, giá trị trả lại của hàm là true nếu n là số nguyên tố; là false nếu n không là số nguyên tố.

2. Thiết kế form có:

- TextBox txtHoTen: nhập họ và tên.
- TextBox txtN: nhập số nguyên n.
- TextBox txtKQ: xuất kết quả.
- Button btnTEN: gọi hàm TEN xử lý chuỗi họ và tên, xuất kết quả vào txtKQ.
- Button btnNGTO: gọi hàm NGTO xử lý số nguyên n, xuất kết quả vào txtKQ.
- Button btnXoa: xóa trống tất cả các TextBox và đưa con trỏ vào ô txtHoTen
- Button btnDung: dừng chương trình.

-- oOo --

## Chương 5: SQL SERVER 2008

### Bài 8: SQL SERVER 2008

#### I. Tổng quan về SQL

##### **1. Khái niệm SQL**

- SQL (Structured Query Language – ngôn ngữ hỏi có cấu trúc) là công cụ sử dụng để tổ chức, quản lý và truy xuất dữ liệu được lưu trữ trong các cơ sở dữ liệu.
- SQL là một hệ thống ngôn ngữ bao gồm tập các câu lệnh sử dụng để tương tác với cơ sở dữ liệu quan hệ.
- SQL được sử dụng để điều khiển tất cả các chức năng mà một hệ quản trị cơ sở dữ liệu cung cấp cho người dùng bao gồm:
  - **Định nghĩa dữ liệu:** SQL cung cấp khả năng định nghĩa các cơ sở dữ liệu, các cấu trúc lưu trữ và tổ chức dữ liệu cũng như mối quan hệ giữa các thành phần dữ liệu.
  - **Truy xuất và thao tác dữ liệu:** Với SQL, người dùng có thể dễ dàng thực hiện các thao tác truy xuất, bổ sung, cập nhật và loại bỏ dữ liệu trong các cơ sở dữ liệu.
  - **Điều khiển truy cập:** SQL có thể được sử dụng để cấp phát và kiểm soát các thao tác của người sử dụng trên dữ liệu, đảm bảo sự an toàn cho cơ sở dữ liệu.
  - **Đảm bảo toàn vẹn dữ liệu:** SQL định nghĩa các ràng buộc toàn vẹn trong cơ sở dữ liệu nhằm đảm bảo tính hợp lệ và chính xác của dữ liệu trước các thao tác cập nhật cũng như các lỗi của hệ thống.

##### **2. Vai trò của SQL**

- SQL không phải là một hệ quản trị cơ sở dữ liệu, do nó không thể tồn tại độc lập.
- SQL là một phần của hệ quản trị cơ sở dữ liệu, nó xuất hiện trong các hệ quản trị cơ sở dữ liệu với vai trò ngôn ngữ và là công cụ giao tiếp giữa người sử dụng và hệ quản trị cơ sở dữ liệu.
  - SQL có những vai trò như sau:
    - **SQL là ngôn ngữ có tính tương tác:** Người sử dụng có thể dễ dàng thông qua các trình tiện ích để gửi các yêu cầu dưới dạng các câu lệnh SQL đến cơ sở dữ liệu và nhận kết quả trả về từ cơ sở dữ liệu.
    - **SQL là ngôn ngữ lập trình cơ sở dữ liệu:** Các lập trình viên có thể nhúng các câu lệnh SQL vào trong các ngôn ngữ lập trình để xây dựng nên các chương trình ứng dụng giao tiếp với cơ sở dữ liệu.
    - **SQL là ngôn ngữ quản trị cơ sở dữ liệu:** Thông qua SQL, người quản trị cơ sở dữ liệu có thể quản lý được cơ sở dữ liệu, định nghĩa các cấu trúc lưu trữ dữ liệu, điều khiển truy cập cơ sở dữ liệu, ...
    - **SQL là ngôn ngữ cho các hệ thống khách/chủ (client/server):** Trong các hệ thống cơ sở dữ liệu khách/chủ, SQL được sử dụng như là công cụ để giao tiếp giữa các trình ứng dụng phía máy khách với máy chủ cơ sở dữ liệu.
    - **SQL là ngôn ngữ truy cập dữ liệu trên Internet:** Cho đến nay, hầu hết các máy chủ Web cũng như các máy chủ trên Internet sử dụng SQL với vai trò là ngôn ngữ để tương tác với dữ liệu trong các cơ sở dữ liệu.
    - **SQL là ngôn ngữ cơ sở dữ liệu phân tán:** Đối với các hệ quản trị cơ sở dữ liệu phân tán, mỗi một hệ thống sử dụng SQL để giao tiếp với các hệ thống khác trên mạng, gửi và nhận các yêu cầu truy xuất dữ liệu với nhau.

- **SQL là ngôn ngữ sử dụng cho các cổng giao tiếp cơ sở dữ liệu:** Trong một hệ thống mạng máy tính với nhiều hệ quản trị cơ sở dữ liệu khác nhau, SQL thường được sử dụng như là một chuẩn ngôn ngữ để giao tiếp giữa các hệ quản trị cơ sở dữ liệu.

## **II. Tổng quan về cơ sở dữ liệu (CSDL) quan hệ**

### **1. Mô hình dữ liệu quan hệ**

- CSDL quan hệ là một CSDL trong đó tất cả dữ liệu được tổ chức trong các bảng (table) có mối quan hệ với nhau. Mỗi bảng (table) bao gồm các dòng (record/bản ghi/bộ) và các cột (field/trường/thuộc tính).

- Tóm lại, một CSDL bao gồm nhiều bảng (table) có mối quan hệ với nhau (relationship). Ví dụ:

MADONVI	TENDONVI	DIENTHOAI	MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
01	Phòng Kế toán	821451	NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
02	Phòng Tổ chức	831414	NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
03	Phòng điều hành	823351	NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
04	Phòng đối ngoại	841457	NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
05	Phòng Tài vụ	821451	NV03002	Nguyễn Hữu Tình	18/08/1976	7 Hà Nội	849290	1.92	03
			NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

### **2. Bảng (Table)**

Bảng (table) bao gồm các yếu tố sau:

- Tên của bảng: được xác định duy nhất.
- Cấu trúc của bảng: tập hợp các cột (field/trường/thuộc tính).
- Dữ liệu của bảng: tập hợp các dòng (record/bản ghi/bộ) hiện có trong bảng.

Ví dụ: Table **DONVI**

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

### **3. Khóa chính của bảng (Primary Key)**

- Mỗi bảng phải có một cột (hoặc một tập các cột) mà giá trị dữ liệu của nó xác định duy nhất một dòng trong tập hợp các dòng trong bảng.
- Một cột (hoặc một tập các cột) có tính chất này gọi là khóa chính của bảng (Primary Key).

Ví dụ: Table **DONVI**

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

(có khóa chính là **MADONVI**)

#### 4. Mối quan hệ (Relationship) và khóa ngoại (Foreign Key)

- Mối quan hệ (Relationship) được thể hiện thông qua ràng buộc *giá trị dữ liệu xuất hiện ở bảng này phải có xuất hiện trước ở một bảng khác*.

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tình	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

- Một cột (hoặc tập hợp các cột) (field/trường/thuộc tính) trong một bảng mà giá trị của nó được xác định từ khóa chính (Primary Key) của một bảng khác được gọi là khóa ngoại (Foreign Key).

#### 5. Sơ lược về câu lệnh SQL

Câu lệnh	Chức năng
<i>Thao tác dữ liệu</i>	
SELECT	Truy xuất dữ liệu
INSERT	Bổ sung dữ liệu
UPDATE	Cập nhật dữ liệu
DELETE	Xóa dữ liệu
TRUNCATE	Xóa toàn bộ dữ liệu trong bảng
<i>Định nghĩa dữ liệu</i>	
CREATE TABLE	Tạo bảng
DROP TABLE	Xóa bảng
ALTER TABLE	Sửa cấu trúc bảng
CREATE FUNCTION	Tạo hàm (do người sử dụng định nghĩa)
ALTER FUNCTION	Sửa đổi hàm
DROP FUNCTION	Xóa hàm

CREATE TRIGGER	Tạo trigger
ALTER TRIGGER	Sửa trigger
DROP TRIGGER	Xóa trigger

## 6. Quy tắc sử dụng tên trong SQL

- Trong câu lệnh SQL, nếu ta cần chỉ đến một bảng *do một người dùng khác sở hữu* (hiển nhiên là phải được phép) thì *tên của bảng* phải được *viết sau tên của người sở hữu* và *phân cách* với tên người sở hữu *bởi dấu chấm* theo công thức: *tên\_người\_sở\_hữu.tên\_bảng*

- Trong câu lệnh SQL, nếu có sử dụng từ *hai cột* trở lên có *cùng tên* trong các bảng khác nhau thì bắt buộc phải chỉ định thêm *tên bảng trước tên cột*; *tên bảng* và *tên cột* được *phân cách* nhau *bởi dấu chấm* theo công thức: *tên\_bảng.tên\_cột*

## 7. Kiểu dữ liệu

Tên kiểu	Mô tả
CHAR (n)	Kiểu chuỗi với độ dài cố định
NCHAR (n)	Kiểu chuỗi với độ dài cố định hỗ trợ UNICODE
VARCHAR (n)	Kiểu chuỗi với độ dài chính xác
NVARCHAR (n)	Kiểu chuỗi với độ dài chính xác hỗ trợ UNICODE
INTEGER	Số nguyên có giá trị từ $-2^{31}$ đến $2^{31} - 1$
INT	Như kiểu Integer
TINYINT	Số nguyên có giá trị từ 0 đến 255
SMALLINT	Số nguyên có giá trị từ $-2^{15}$ đến $2^{15} - 1$
BIGINT	Số nguyên có giá trị từ $-2^{63}$ đến $2^{63} - 1$
NUMERIC (p,s)	Kiểu số với độ chính xác cố định
DECIMAL (p,s)	Tương tự kiểu Numeric
FLOAT	Số thực có giá trị từ $-1.79E+308$ đến $1.79E+308$
REAL	Số thực có giá trị từ $-3.40E + 38$ đến $3.40E + 38$
MONEY	Kiểu tiền tệ
BIT	Kiểu bit (có giá trị 0 hoặc 1)
DATETIME	Kiểu ngày giờ (chính xác đến phần trăm của giây)
SMALLDATETIME	Kiểu ngày giờ (chính xác đến phút)
BINARY	Dữ liệu nhị phân với độ dài cố định (tối đa 8000 bytes)
VARBINARY	Dữ liệu nhị phân với độ dài chính xác (tối đa 8000 bytes)
IMAGE	Dữ liệu nhị phân với độ dài chính xác ( $\leq 2,147,483,647$ bytes)
TEXT	Dữ liệu kiểu chuỗi với độ dài lớn (tối đa 2,147,483,647 ký tự)
NTEXT	Dữ liệu kiểu chuỗi với độ dài lớn và hỗ trợ UNICODE (tối đa 1,073,741,823 ký tự)

## 8. Toán tử

Toán tử	Ý nghĩa
a) Logic	
AND / OR	Và / Hoặc
b) So sánh	
=	Bằng
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng

<=	Nhỏ hơn hoặc bằng	
<>	Khác	
!>	Không lớn hơn	
!<	Không nhỏ hơn	
c) Danh sách		
IN	Nằm trong danh sách	
NOT IN	Không nằm trong danh sách	
d) Giới hạn dữ liệu		
BETWEEN	Giá trị BETWEEN a AND b nghĩa là ( $a \leq \text{Giá trị} \leq b$ )	
NOT BETWEEN	Giá trị NOT BETWEEN a AND b nghĩa là ( $\text{Giá trị} < a$ ) AND ( $\text{Giá trị} > b$ )	
LIKE	Mô tả khuôn dạng dữ liệu cần tìm kiếm có sử dụng ký tự đại diện:	
	%	Chuỗi ký tự bất kỳ
	_	Một ký tự bất kỳ
	[]	Ký tự bất kỳ trong giới hạn được chỉ định
	[^]	Ký tự bất kỳ không nằm trong giới hạn được chỉ định

### III. Table (Bảng)

#### \* Quy tắc khi viết câu lệnh SQL:

- Các câu lệnh SQL không phân biệt chữ hoa và chữ thường. Tuy nhiên, để dễ đọc nên viết hoa từ khóa trong mệnh đề.
- Câu lệnh SQL có thể viết trên một dòng hay nhiều dòng. Nhưng nên viết mỗi dòng một mệnh đề.
- Không tách từ khóa trên nhiều dòng, không viết tắt từ khóa.
- Câu lệnh SQL kết thúc bằng dấu chấm phẩy (;

#### 1. Tạo bảng

Cú pháp:

```
CREATE TABLE tên_bảng
(
    tên_cột      thuộc_tính_cột      các_ràng_buộc
    [ , ...
     , tên_cột_n  thuộc_tính_cột_n  các_ràng_buộc_cột_n ]
    [ , các_ràng_buộc_trên_bảng ]
)
```

Trong đó:

thuộc\_tính\_cột Tên của bảng cần tạo (<=128 ký tự)

thuộc\_cột Tên của cột (field / trường) cần định nghĩa

thuộc\_tính\_cột Gồm:

- Kiểu dữ liệu của cột (field).
- Giá trị mặc định của cột (filed).
- IDENTITY - giá trị tự động tăng, dùng với field kiểu số.
- NULL / NOT NULL

các\_ràng\_buộc Các ràng buộc được sử dụng trên mỗi cột (field) hoặc trên bảng.

#### Ví dụ 8.1:

Tạo bảng NHANVIEN gồm các field MANV (mã nhân viên), HOTEN (họ và tên), NGAYSINH (ngày sinh của nhân viên), DIACHI (địa chỉ của nhân viên), HSLUONG (hệ số lương), MADONVI (mã đơn vị).

\* Hướng dẫn:

```
CREATE TABLE nhanvien
```

```
(
```

manv	NVARCHAR(10)	NOT NULL,
hoten	NVARCHAR(50)	NOT NULL,
ngaysinh	DATE	NULL,
diachi	NVARCHAR(100)	NULL,
dienthoai	NVARCHAR(10)	NULL,
hsluong	DECIMAL(3,2)	DEFAULT (1.92)
madonvi	NVARCHAR(10)	NOT NULL

```
)
```

## 2. Tạo ràng buộc

### a. Ràng buộc CHECK:

- Chỉ định điều kiện hợp lệ đối với dữ liệu khi có sự thay đổi dữ liệu trên bảng.
- Dùng với các lệnh INSERT, UPDATE.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]
    CHECK (điều_kiện)
```

### Ví dụ 8.2:

Tạo bảng NHANVIEN như ví dụ 8.1, trong đó:

- NGAYSINH < '1/1/1990'
- DIENTHOAI của nhân viên là một chuỗi 6 chữ số.

\* Hướng dẫn:

```
CREATE TABLE nhanvien
```

```
(
```

manv	NVARCHAR(10)	NOT NULL,
hoten	NVARCHAR(50)	NOT NULL,
ngaysinh	DATE	NULL
	CONSTRAINT CK_nhanvien_ngaysinh	
	CHECK (ngaysinh < '1/1/1990'),	
diachi	NVARCHAR(100)	NULL,
dienthoai	NVARCHAR(10)	NULL,
	CONSTRAINT CK_nhanvien_dienthoai	
	CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9]____')	
hsluong	DECIMAL(3,2)	DEFAULT (1.92)
madonvi	NVARCHAR(10)	NOT NULL

```
)
```

### b. Ràng buộc PRIMARY KEY:

- Chỉ định khoá chính của bảng.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]
    PRIMARY KEY [(danh_sách_cột)]
```

Lưu ý:

- Mỗi bảng có nhiều nhất một ràng buộc PRIMARY KEY.
- Một khoá chính có thể bao gồm nhiều cột nhưng không vượt quá 16 cột.

### Ví dụ 8.3:

Tạo bảng NHANVIEN như ví dụ 8.2, với khoá chính là MANV

\* Hướng dẫn:

```
CREATE TABLE nhanvien
(
    manv      NVARCHAR(10)      NOT NULL,
    hoten     NVARCHAR(50)       NOT NULL,
    ngaysinh   DATE             NULL
        CONSTRAINT CK_nhanvien_ngaysinh
        CHECK (ngaysinh < '1/1/1990'),
    diachi    NVARCHAR(100)      NULL,
    dienthoai NVARCHAR(10)       NULL,
    CONSTRAINT CK_nhanvien_dienthoai
    CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9]_____')
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)
    madonvi   NVARCHAR(10)      NOT NULL,
    CONSTRAINT PK_nhanvien_manv PRIMARY KEY
)
```

### c. Ràng buộc UNIQUE:

- Chỉ định khoá phụ cho bảng.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]
UNIQUE [(danh_sách_cột)]
```

### Ví dụ 8.4:

Tạo bảng NHANVIEN như ví dụ 8.3, trong đó không cho phép các nhân viên khác nhau được trùng điện thoại với nhau.

\* Hướng dẫn:

```
CREATE TABLE nhanvien
(
    manv      NVARCHAR(10)      NOT NULL,
    hoten     NVARCHAR(50)       NOT NULL,
    ngaysinh   DATE             NULL
        CONSTRAINT CK_nhanvien_ngaysinh
        CHECK (ngaysinh < '1/1/1990'),
    diachi    NVARCHAR(100)      NULL,
    dienthoai NVARCHAR(10)       NULL,
    CONSTRAINT CK_nhanvien_dienthoai
    CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9]_____')
    hsluong   DECIMAL(3,2)      DEFAULT (1.92)
    madonvi   NVARCHAR(10)      NOT NULL,
    CONSTRAINT PK_nhanvien_manv PRIMARY KEY,
    CONSTRAINT UNIQUE_nhanvien_dienthoai UNIQUE(dienthoai)
)
```

### d. Ràng buộc FOREIGN KEY (khóa ngoại)

- Một cột (hay một tập các cột) trong một bảng được gọi là khoá ngoại (ràng buộc FOREIGN KEY) nếu giá trị của nó được xác định từ khoá chính (PRIMARY KEY) hoặc khoá phụ (UNIQUE) của một bảng dữ liệu khác.

Cú pháp:

```
[CONSTRAINT tên_ràng_buộc]
```

FOREIGN KEY [(danh\_sách\_cột)]

REFERENCES tên\_bảng\_tham\_chiếu(danh\_sách\_cột\_tham\_chiếu)

[ON DELETE CASCADE | NO ACTION | SET NULL | SET DEFAULT]

[ON UPDATE CASCADE | NO ACTION | SET NULL | SET DEFAULT]

Trong đó:

- CASCADE: Tự động xoá (cập nhật) nếu bản ghi được tham chiếu bị xoá (cập nhật).
- NO ACTION (Mặc định): Nếu bản ghi trong bảng tham chiếu đang được tham chiếu bởi một bản ghi bất kỳ trong bảng được định nghĩa thì bản ghi đó không được phép xoá hoặc cập nhật (đối với cột được tham chiếu).
- SET NULL: Cập nhật lại khoá ngoài của bản ghi thành giá trị NULL (nếu cột cho phép nhận giá trị NULL).
- SET DEFAULT: Cập nhật lại khoá ngoài của bản ghi nhận giá trị mặc định (nếu cột có qui định giá trị mặc định).

### Ví dụ 8.5:

Tạo bảng NHANVIEN như ví dụ 8.4, trong đó khoá ngoài trên cột MADONVI (bảng DONVI). Giả sử rằng bảng DONVI đã được định nghĩa.

\* Hướng dẫn:

CREATE TABLE nhanvien

(

```

manv      NVARCHAR(10)    NOT NULL,
hoten     NVARCHAR(50)    NOT NULL,
ngaysinh  DATE           NULL
                      CONSTRAINT CK_nhanvien_ngaysinh
                      CHECK (ngaysinh < '1/1/1990'),
diachi    NVARCHAR(100)   NULL,
dienthoai NVARCHAR(10)   NULL,
                      CONSTRAINT CK_nhanvien_dienthoai
                      CHECK (dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9][0-9]____')
hsluong   DECIMAL(3,2)    DEFAULT (1.92)
madonvi   NVARCHAR(10)    NOT NULL,
                      CONSTRAINT PK_nhanvien_manv PRIMARY KEY,
                      CONSTRAINT UNIQUE_nhanvien_dienthoai UNIQUE(dienthoai) ,
                      CONSTRAINT FK_nhanvien_madonvi
                        FOREIGN KEY(madonvi)
                        REFERENCES donvi(madonvi)
                        ON DELETE CASCADE
                        ON UPDATE CASCADE
)

```

### 3. Sửa cấu trúc bảng

#### a. Cú pháp

ALTER TABLE tên\_bảng

ADD định\_nghĩa\_cột |

ALTER COLUMN tên\_cột kiêu\_dữ\_liệu [NULL | NOT NULL] |

DROP COLUMN tên\_cột |

ADD CONSTRAINT tên\_ràng\_buộc định\_nghĩa\_ràng\_buộc |

DROP CONSTRAINT tên\_ràng\_buộc

#### b. Ví dụ 8.6

Bổ sung vào bảng DONVI ở ví dụ 8.5, cột GHICHU

\* Hướng dẫn:

```
ALTER TABLE nhanvien
ADD
    ghichu NVARCHAR(50)
```

#### 4. Xóa bảng

##### a. Cú pháp

```
DROP TABLE tên_bảng
```

##### b. Ví dụ 8.7:

Xoá bảng DONVI ra khỏi CSDL.

Lưu ý: Cột MADONVI trong bảng DONVI đang được tham chiếu bởi khoá ngoài FK\_nhanvien\_madonvi trong bảng NHANVIEN.

\* Hướng dẫn:

- Xoá bỏ ràng buộc FK\_nhanvien\_madonvi khỏi bảng NHANVIEN

```
ALTER TABLE nhanvien
```

```
DROP CONSTRAINT FK_nhanvien_madonvi
```

- Xoá bảng DONVI:

```
DROP TABLE donvi
```

## Thực hành

1. Khởi động SQL SERVER 2008:

a. Start → All programs → Microsoft SQL Server 2008 → SQL Server Management Studio

b. **Chứng thực:** chọn 1 trong 2 chế độ

- ❖ Là của Windows
- ❖ Là của SQL Server



c. Chọn **Connect**

2. Tạo database:

Ở cửa sổ Object Explorer, click chuột phải vào **Database** sau đó chọn **New database**

- Database name: **banhang**

- Chọn **Ok**

3. Tạo cấu trúc table:

a. Tạo table **Nhanvien**

\* Ở cửa sổ Object Explorer:

- Nhấp dấu + trước database **banhang** (thành dấu -)
- Click chuột phải vào **Table** sau đó chọn **New table**
- Khai báo cấu trúc: **Ví dụ 8.1**
- Lưu: nhấp 

### b. Tạo table Donvi : tự thực hiện.

4. Tạo ràng buộc cho table

\* Chọn table Nhanvien

#### a. Ràng buộc CHECK

**Ví dụ 8.2**

\* Ở cửa sổ hiển thị cấu trúc của table Nhanvien

- Click chuột phải vào field **ngaysinh**, chọn **CheckConstraints ...**
- Nhấp **Add**
- + (Name): gõ CK\_nhanvien\_ngaysinh  
Expression: gõ **([ngaysinh] < '1/1/1990')**
- Nhấp **Add** (lần 2)
- + (Name): gõ CK\_nhanvien\_dienthoai
- + Expression: gõ **(dienthoai LIKE '[0-9][0-9][0-9][0-9][0-9][0-9]\_\_\_\_\_')**
- Nhấp **Close**

\* Chọn table **Donvi**: tự thực hiện cho field dienthoai.

#### b. Ràng buộc PRIMARY KEY

\* Chọn table Nhanvien

**Ví dụ 8.3**

\* Ở cửa sổ hiển thị cấu trúc của table Nhanvien

- Click phải chuột vào field **manv** và chọn **Set Primary Key**

\* Cho table **Donvi**: tự thực hiện cho field madonvi .

#### c. Ràng buộc UNIQUE

\* Chọn table Nhanvien

**Ví dụ 8.4**

\* Ở cửa sổ hiển thị cấu trúc của table Nhanvien

- Click phải chuột vào field **dienthoai** và chọn **Indexes / Keys...**
- Nhấp **Add**
- + (Name): gõ **UNIQUE\_nhanvien\_dienthoai**
- + Columns: dienthoai (ASC)
- + Type: Unique Key
- Nhấp **Close**

5. Tạo quan hệ (relationship) cho các table: Donvi, Nhanvien

**Ví dụ 8.5**

\* Ở cửa sổ Object Explorer

- Click chuột phải vào **Database Diagrams** sau đó chọn **New database Diagram**
- Chọn các table tham gia tạo quan hệ, nhấp **Close**.
- Rê field madonvi (từ table nhanvien) thả vào field madonvi (của table donvi).
- Khai báo:

Primary key table	Foreign key table
Donvi	Nhanvien
madonvi	madonvi

6. Thêm field GHICHU vào table DONVI

**Ví dụ 8.6**

- \* Ở cửa sổ hiển thị cấu trúc của table Donvi
- Khai báo (ở cuối) thêm field: ghichu, nvarchar(50), NULL

7. Nhập dữ liệu cho các table: Donvi, Nhanvien

- \* Ở cửa sổ Object Explorer:

- Click chuột phải vào **dbo.donvi / dbo.nhanvien** sau đó chọn **Edit top 200 Rows**
- Nhập dữ liệu

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tình	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

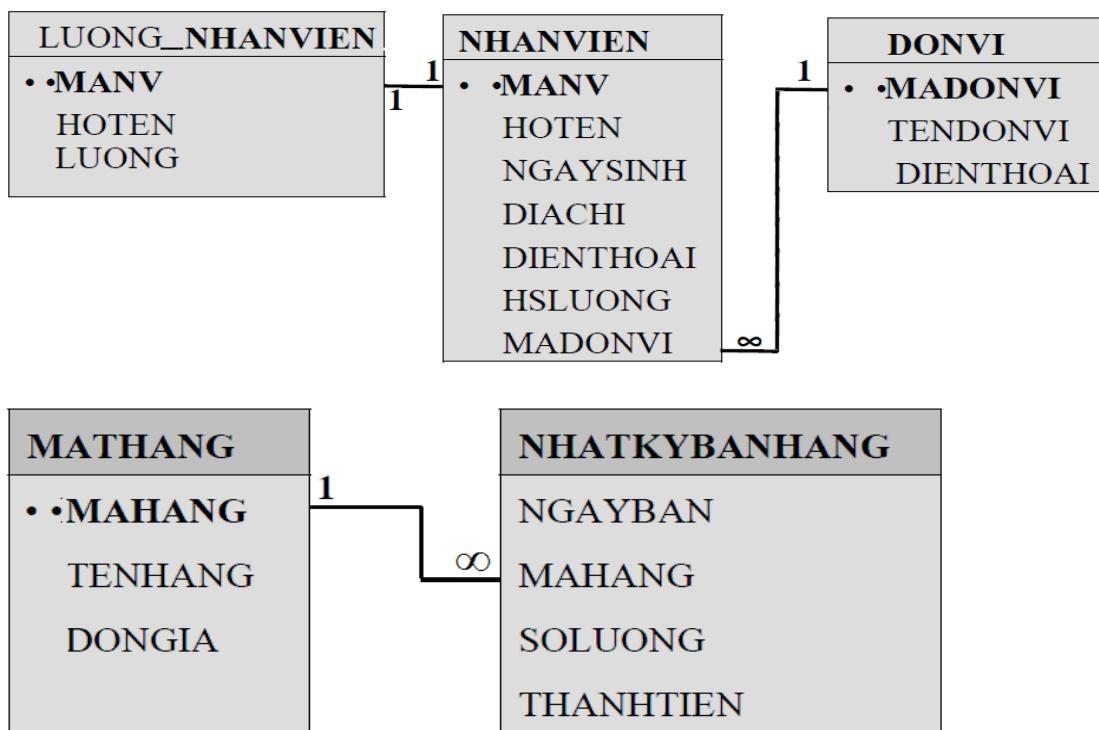
-- oOo --

## Bài 9: CÁC CÂU LỆNH TRUY VẤN

\* Xét CSDL banhang gồm các table:

**NHANVIEN, DONVI, MATHANG, NHATKYBANHANG**

- Câu trúc:



- Dữ liệu:

**Bảng NHANVIEN**

MANV	HOTEN	NGAYSINH	DIACHI	DIENTHOAI	HSLUONG	MADONVI
NV01001	Nguyễn Thị Hoa	05/05/1976	56 Lê Duẩn	521304	2.11	01
NV02001	Lê Hoài Nam	03/05/1976	32 Trần Phú	823145	1.86	02
NV02002	Hoàng Nam Phong	05/08/1971	66 Hoàng Diệu	521247	3.21	02
NV03001	Trần Nguyên Phong	20/12/1976	7 Hà Nội	849290	1.92	03
NV03002	Nguyễn Hữu Tình	18/08/1976	7 Hà Nội	849290	1.92	03
NV05001	Nguyễn Trung Kiên	14/05/1972	77 Nguyễn Huệ	823236	1.86	05

**Bảng DONVI**

MADONVI	TENDONVI	DIENTHOAI
01	Phòng Kế toán	821451
02	Phòng Tổ chức	831414
03	Phòng điều hành	823351
04	Phòng đối ngoại	841457
05	Phòng Tài vụ	821451

#### IV. Các câu lệnh truy vấn

##### 1. Câu lệnh SELECT

Công dụng:

- Chọn các field từ bảng.

Cú pháp:

```
SELECT [ALL | DISTINCT][TOP n] danh_sách_chọn
      [INTO tên_bảng_mới]
      FROM danh_sách_bảng [ | khung_nhìn]
      [WHERE điều_kiện]
      [GROUP BY danh_sách_cột]
      [HAVING điều_kiện]
      [ORDER BY cột_sắp_xếp]
      [COMPUTE danh_sách_hàm_gộp [BY danh_sách_cột]]
```

Chú ý: Các thành phần trong câu lệnh SELECT phải được sử dụng theo đúng thứ tự trên.

##### c. Danh sách chọn trong câu lệnh SELECT

###### - Chọn tất cả các field

###### Ví dụ 9.1a:

Câu lệnh sau đây cho biết thông tin của các nhân viên trong bảng NHANVIEN.

```
SELECT * FROM nhanvien n1
```

###### - Đổi tên các cột trong kết quả

###### Ví dụ 9.1b:

Câu lệnh sau đây cho biết HOTEN (họ tên) được đổi thành **Họ và Tên**, DIACHI (địa chỉ) được đổi thành **Địa Chỉ** của các nhân viên trong bảng NHANVIEN.

```
SELECT 'Họ và Tên' = hoten, 'Địa Chỉ' = diachi
      FROM nhanvien
```

Hoặc:

```
SELECT hoten 'Họ và Tên', diachi 'Địa Chỉ'
      FROM nhanvien
```

###### - Sử dụng cấu trúc CASE để đổi tên các cột trong kết quả

###### Ví dụ 9.1c:

Câu lệnh sau đây cho biết HOTEN (họ tên), HSLUONG (hệ số lương) và (xếp loại lương) của các nhân viên trong bảng NHANVIEN theo HSLUONG (hệ số lương).

```
SELECT 'Họ và Tên' = hoten, 'Hệ số lương' = hsluong, 'Hệ số lương' =
      CASE
```

WHEN hsluong = NULL	THEN 'Không xác định'
WHEN hsluong <= 1.92	THEN 'Lương thấp'
WHEN hsluong <= 3.11	THEN 'Lương trung bình'
WHEN hsluong <= 5.2	THEN 'Lương cao'
ELSE 'Lương rất cao'	

END

FROM nhanvien

###### - Thêm chuỗi ký tự trong kết quả

###### Ví dụ 9.1d:

Câu lệnh sau đây sẽ cho thêm chuỗi 'Hệ số lương là:' ở trước cột HSLUONG (hệ số lương) trong từng dòng kết quả.

```
SELECT 'Họ và Tên' = hoten, 'Hệ số lương là:', 'Hệ số lương' = hsluong
      FROM nhanvien
```

###### - Tính toán các giá trị trong câu lệnh SELECT

### Ví dụ 9.1e:

Câu lệnh sau đây sẽ cho HOTEN (họ tên) và LUONG (lương) của nhân viên theo công thức LUONG = HSLUONG \* 730000.

```
SELECT 'Họ và Tên' = hoten, 'Lương' = hsluong * 730000
FROM nhanvien
```

### - Từ khóa DISTINCT: dùng để loại bỏ những dòng dữ liệu có kết quả giống nhau

#### Ví dụ 9.1f:

Câu lệnh sau sẽ cho các giá trị hsluong khác nhau trong bảng NHANVIEN

```
SELECT hsluong
FROM nhanvien
```

### - Tạo bảng mới bằng câu lệnh SELECT ... INTO

#### Ví dụ 9.1g:

Câu lệnh sau sẽ tạo bảng có tên NHANVIEN\_LUU gồm các field HOTEN (họ tên), DIACHI (địa chỉ) của các nhân viên có HSLUONG > 1.92 từ bảng NHANVIEN.

```
SELECT hoten, diachi
INTO nhanvien_luu
FROM nhanvien
WHERE hsluong > 1.92
```

### - Sắp xếp kết quả (ASC: tăng, DESC: giảm) bằng ORDER BY

#### Ví dụ 9.1h:

Câu lệnh sau đây sẽ sắp xếp các nhân viên theo thứ tự giảm dần của HSLUONG (hệ số lương), nếu HSLUONG bằng nhau thì sắp xếp kết quả theo thứ tự tăng dần của NGAYSINH (ngày sinh)

```
SELECT hoten, ngaysinh, hsluong
FROM nhanvien
ORDER BY hsluong DESC, ngaysinh ASC
```

### d. Xác định bảng bằng mệnh đề FROM

```
FROM danh_sách_bảng [ | khung_nhìn]
```

#### Ví dụ 9.1i:

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) của các nhân viên bằng cách gán bí danh (alias) cho bảng NHANVIEN.

```
SELECT hoten, diachi
FROM nhanvien n1
```

### e. Đặt điều kiện truy xuất dữ liệu bằng mệnh đề WHERE

```
WHERE điều_kiện
```

điều\_kiện: sử dụng các phép toán sau

- + So sánh: =, >, <, >=, <=, <>, !=
- + Giới hạn: BETWEEN ... AND ..., NOT BETWEEN ... AND ...
- + Danh sách: IN , NOT IN
- + Khuôn dạng: LIKE , NOT LIKE
  - Với các ký tự đại diện: % , \_ , [ ] , [^]
- + Các giá trị chưa biết: IS NULL , IS NOT NULL
- + Kết hợp các điều kiện: AND , OR

#### Ví dụ 9.1j:

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) và DIENTHOAI (điện thoại) của các nhân viên có HSLUONG (hệ số lương) lớn hơn 1.92

```
SELECT hoten, diachi
FROM nhanvien
WHERE hsluong > 1.92
```

**Ví dụ 9.1k:**

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) và DIENTHOAI (điện thoại) của các nhân viên có HSLUONG (hệ số lương) trong khoảng 1.92 đến 3.11

```
SELECT hoten, diachi
FROM nhanvien
WHERE hsluong BETWEEN 1.92 AND 3.11
```

**Ví dụ 9.1l:**

Câu lệnh sau đây cho biết HOTEN (họ tên), DIACHI (địa chỉ) và DIENTHOAI (điện thoại) của các nhân viên có HSLUONG (hệ số lương) là 1.86, 1.92, 2.11

```
SELECT hoten, diachi
FROM nhanvien
WHERE hsluong IN (1.86, 1.92, 2.11)
```

\* Hoặc:

```
SELECT hoten, diachi
FROM nhanvien
WHERE hsluong = 1.86 OR hsluong = 1.92 OR hsluong = 2.11
```

**Ví dụ 9.1m:**

Câu lệnh sau đây cho biết thông tin của nhân viên có tên là NAM.

```
SELECT *
FROM nhanvien
WHERE hoten LIKE '%NAM'
```

**Ví dụ 9.1n:**

Câu lệnh sau đây cho biết thông tin của nhân viên không có điện thoại.

```
SELECT *
FROM nhanvien
WHERE dienthoai IS NULL
```

## 2. Thêm dữ liệu INSERT

Công dụng:

- Thêm dòng dữ liệu (mẫu tin/record) vào bảng.

Cú pháp:

```
INSERT INTO tên_bảng
[(danh_sách_cột)] VALUES(danh_sách_trị)
```

**Ví dụ 9.2a:**

Câu lệnh sau đây thêm một dòng dữ liệu vào bảng DONVI

```
INSERT INTO donvi
VALUES('06', 'Phòng CTCT-HSSV', '821460')
```

**Ví dụ 9.2b:**

Câu lệnh sau đây thêm một dòng dữ liệu vào bảng NHANVIEN

```
INSERT INTO nhanvien
VALUES('NV02003', 'Lê Thị Mai', '23/05/1972', NULL, '523312', 1.92, '02')
```

**Ví dụ 9.2c:**

Câu lệnh sau đây thêm một dòng dữ liệu vào bảng NHANVIEN nhưng chỉ điền dữ liệu vào một số cột.

```
INSERT INTO nhanvien(manv, hoten, diachi, madonvi)
VALUES('NV05002', 'Nguyễn Thị Hạnh Dung', '56 Trần Phú', '05')
```

**Ví dụ 9.2d:** Thêm dữ liệu vào bảng với dữ liệu lấy từ bảng khác

Câu lệnh sau đây thêm dữ liệu vào bảng LUONG\_NHANVIEN với dữ liệu lấy từ bảng NHANVIEN.

```
INSERT INTO luong_nhanvien
SELECT manv, hoten, hsluong*730000 FROM nhanvien
```

### 3. Cập nhật dữ liệu UPDATE

Công dụng:

- Cập nhật dữ liệu trong các bảng.

Cú pháp:

```
UPDATE tên_bảng
SET tên_cột = biểu_thúc
[ , ...
, tên_cột_k = biểu_thúc_k]
[FROM danh_sách_bảng]
[WHERE điều_kiện]
```

#### Ví dụ 9.3a:

Câu lệnh sau đây tăng HSLUONG (hệ số lương) thêm 0.2 cho các nhân viên có MADONVI là 04.

```
UPDATE nhanvien
SET hsluong = hsluong + 0.2
WHERE madonvi = '04'
```

#### Ví dụ 9.3b:

Câu lệnh sau đây sẽ cập nhật giá trị cho field THANHTIEN (thành tiền) trong bảng NHATKYBANHANG theo công thức THANHTIEN = SOLUONG \* DONGIA.

```
UPDATE nhatkybanhang
SET thanhtien = soluong * MATHANG.dongia
FROM MATHANG
WHERE nhatkybanhang.mahang = MATHANG.mahang
```

### 4. Xóa dữ liệu DELETE

Công dụng:

- Để xóa dữ liệu trong bảng.

Cú pháp:

```
DELETE FROM tên_bảng
[FROM danh_sách_bảng]
[WHERE điều_kiện]
```

#### Ví dụ 9.4:

Câu lệnh sau đây xoá khỏi bảng NHANVIEN những nhân viên làm tại đơn vị có SODIENTHOAI (số điện thoại) là '848484'

```
DELETE FROM nhanvien
FROM donvi
WHERE nhanvien.madonvi = donvi.madonvi AND donvi.dienthoai = '848484'
```

### 5. Xóa toàn bộ dữ liệu TRUNCATE

Công dụng:

- Để xóa toàn bộ dữ liệu trong bảng.

Cú pháp:

```
TRUNCATE TABLE tên_bảng
```

#### Ví dụ 9.5:

Câu lệnh sau xoá toàn bộ dữ liệu trong bảng LUONG\_NHANVIEN

```
DELETE FROM luong_nhanvien
```

Tương đương câu lệnh  
 TRUNCATE TABLE luong\_nhanvien

## V. Một số hàm thường dùng trong SQL Server

### 1. Hàm ngày – giờ

#### a. Hàm DATEADD

Cú pháp:

DATEADD(datepart, number, date)

Datepart: tham số chỉ định thành phần sẽ được cộng thêm vào ngày *date*.

DatePart	Viết tắt
year	yy, yyyy
quarter	qq, q
month	mm, m
dayofyear	dy, y
day	dd, d
week	wk, ww
hour	hh
minute	mi, n
second	ss, s
milisecond	ms

Công dụng: Hàm trả về một giá trị kiểu DateTime bằng cách cộng thêm một khoảng giá trị là *number* vào ngày *date* được chỉ định.

#### b. Hàm DATEDIFF

Cú pháp:

DATEDIFF(datepart, startdate, enddate)

Công dụng: Hàm trả về khoảng thời gian giữa 2 giá trị kiểu ngày *startdate* và *enddate* tùy thuộc vào *datepart*.

Ví dụ:

DateDiff(year, '3/10/2003', '6/15/2010') → kết quả: 7

#### c. Hàm DATEPART

Cú pháp:

DATEPART(datepart, date)

Công dụng: Hàm trả về một số nguyên được trích ra từ thành phần được chỉ định bởi *datepart* trong giá trị ngày *date*.

Ví dụ:

DatePart(year, '6/15/2010') → kết quả: 2010

#### d. Hàm GETDATE

Cú pháp:

GETDATE()

Công dụng: Hàm trả về giá trị là ngày hiện tại.

#### e. Hàm DAY, MONTH, YEAR

Cú pháp:

DAY(date) / MONTH(date) / YEAR(date)

Công dụng: Hàm trả về giá trị là ngày / tháng / năm của ngày *date*.

Ví dụ:

Day('6/15/2010') → kết quả: 15

Month('6/15/2010') → kết quả: 6

Year('6/15/2010') → kết quả: 2010

## 2. Hàm chuỗi

### a. Hàm LEFT

Cú pháp:

LEFT(string, n)

Công dụng: Hàm trích từ chuỗi *string* n ký tự tính từ bên trái.

### b. Hàm RIGHT

Cú pháp:

RIGHT(string, n)

Công dụng: Hàm trích từ chuỗi *string* n ký tự tính từ bên phải.

### c. Hàm SUBSTRING

Cú pháp:

SUBSTRING(string, m, n)

Công dụng: Hàm trích từ chuỗi *string* n ký tự tính từ ký tự thứ m.

### d. Hàm LTRIM

Cú pháp:

LTRIM(string)

Công dụng: Hàm cắt bỏ khoảng trắng thừa bên trái chuỗi *string*.

### e. Hàm RTRIM

Cú pháp:

RTRIM(string)

Công dụng: Hàm cắt bỏ khoảng trắng thừa bên phải chuỗi *string*.

### f. Hàm LEN

Cú pháp:

LEN(string)

Công dụng: Hàm trả về độ dài của chuỗi *string*.

## Thực hành

1. Khởi động SQL SERVER 2008:

a. **Start** → All programs → Microsoft SQL Server 2008 → SQL Server Management Studio

b. **Chứng thực**

c. Chọn **Connect**

2. Ở cửa sổ Object Explorer, click phải chuột lên **banhang** và chọn lệnh **New Query**

3. Ở cửa sổ query, thực hiện các câu lệnh trong các Ví dụ ở trên.

-- oOO --

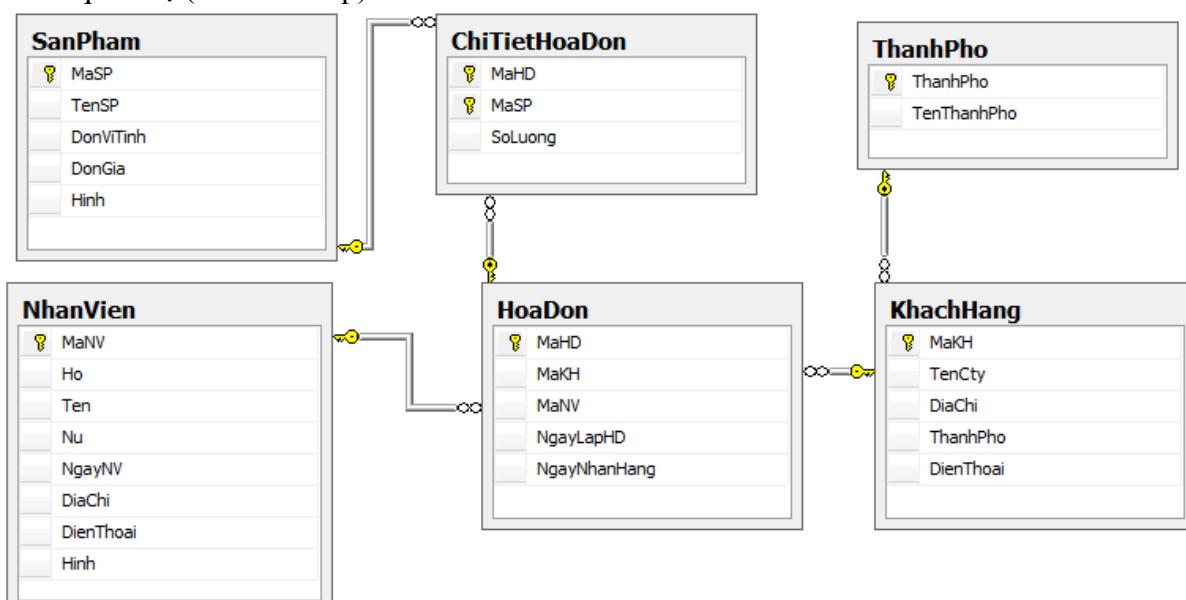
## Chương 6: LẬP TRÌNH KẾT NỐI CSDL SQL SERVER 2008

### Bài 10-11: LẬP TRÌNH KẾT NỐI CƠ SỞ DỮ LIỆU

\* Chuẩn bị:

- Tên máy được sử dụng (**SERVERNAME**) là PC-PC
- Database được sử dụng (**DATABASENAME**) là **QuanLyBanHang**, gồm có các table:
 

+ Nhanvien	+ HoaDon
+ Sanpham	+ ChiTietHoaDon
+ Khachhang	+ ThanhPho
- Với quan hệ (Relationship) như sau:



- Sử dụng Visual Studio 2008 tạo một project mới (Windows Forms Application)

#### I. Tao kết nối - Vân chuyển dữ liệu

##### **1. Khai báo namespace sử dụng**

```
using System.Data.SqlClient;
```

##### **2. Khai báo ở mức class**

```
// Chuỗi kết nối
string strConnectionString = "Data
Source=SERVERNAME;Initial Catalog=DATABASENAME;Integrated
Security=True";
// Đối tượng kết nối
SqlConnection conn = null;
// Đối tượng đưa dữ liệu vào DataTable dtTABLENAME
SqlDataAdapter daTABLENAME = null;
// Đối tượng hiển thị dữ liệu lên Form
DataTable dtTABLENAME = null;
```

##### **3. Khai báo ở Form Load**

```
// Khởi động kết nối
conn = new SqlConnection(strConnectionString);
// Vận chuyển dữ liệu lên DataTable dtTABLENAME
daTABLENAME = new SqlDataAdapter("SELECT * FROM
TABLENAME", conn);
dtTABLENAME = new DataTable();
daTABLENAME.Fill(dtTABLENAME);
```

#### 4. Giải phóng tài nguyên

- . Chuyển Form về chế độ Design View
- . Ở cửa sổ properties của form đang chọn, click Events
- . Nhấp đúp lên sự kiện **FormClosing**
- . Viết code cho sự kiện này như sau:

```
// Giải phóng tài nguyên
dtTABLENAME.Dispose();
dtTABLENAME = null;
// Hủy kết nối
conn = null;
```

## II. Đưa dữ liệu lên các đối tượng ListBox / ComboBox – DataGridView

### 1. Đưa dữ liệu lên ListBox / ComboBox

**Ví dụ 10.1:** Thiết kế form như sau



(lstThanhPho, btnThoat)

\* Yêu cầu:

- . Khi Form load: đưa dữ liệu từ table ThanhPho lên ListBox (lstThanhPho), kèm bẫy lỗi.
- . Nhấp vào Button **Thoát**: dừng chương trình (có hiện hộp thoại hỏi đáp trước khi dừng).

\* Hướng dẫn:

- Thiết kế form như yêu cầu.

- Khai báo namespace sử dụng:

```
using System.Data.SqlClient;
```

- Ở mức class, khai báo:

```
// Chuỗi kết nối
string strConnectionString = "Data Source=PC-PC;Initial
Catalog=QuanLyBanHang;Integrated Security=True";
// Đối tượng kết nối
SqlConnection conn = null;
// Đối tượng đưa dữ liệu vào DataTable dtThanhPho
SqlDataAdapter daThanhPho = null;
// Đối tượng hiển thị dữ liệu lên Form
DataTable dtThanhPho = null;
```

- Form load:

```

try
{
    // Khởi động connection
    conn = new SqlConnection(strConnectionString);
    // Vận chuyển dữ liệu lên DataTable dtThanhPho
    daThanhPho = new SqlDataAdapter("SELECT * FROM
    THANHPHO", conn);
    dtThanhPho = new DataTable();
    dtThanhPho.Clear();
    daThanhPho.Fill(dtThanhPho);
    // Đưa dữ liệu lên ListBox
    this.lstThanhPho.DataSource = dtThanhPho;
    this.lstThanhPho.DisplayMember = "TenThanhPho";
    this.lstThanhPho.ValueMember = "ThanhPho";
}
catch (SqlException)
{
    MessageBox.Show("Không lấy được nội dung trong table
    THANHPHO. Lỗi rồi!!!");
}

```

- FormClosing:

```

// Giải phóng tài nguyên
dtThanhPho.Dispose();
dtThanhPho = null;
// Hủy kết nối
conn = null;

```

- Button Thoát:

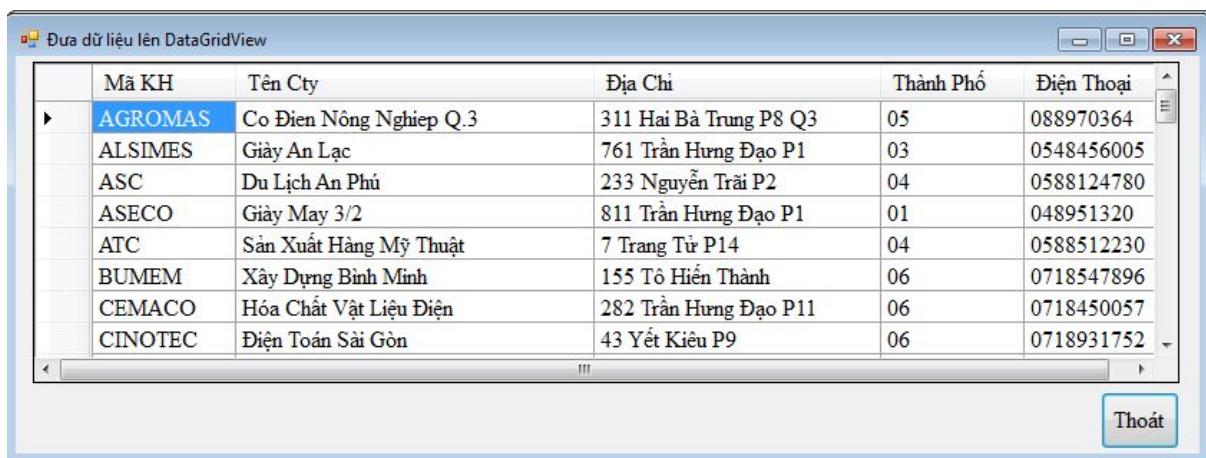
```

// Khai báo biến traloi
DialogResult traloi;
// Hiện hộp thoại hỏi đáp
traloi = MessageBox.Show("Chắc không?", "Trả lời",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
// Kiểm tra có nhấp chọn nút Ok không?
if (traloi == DialogResult.OK) Application.Exit();

```

## 2. Đưa dữ liệu lên DataGridView

**Ví dụ 10.2:** Thiết kế form như sau



(dgvKHACHHANG, btnThoat)

\* Yêu cầu:

- . Khi Form load: đưa dữ liệu từ table KhachHang lên DataGridView (dgvKHACHHANG), có bẫy lỗi.
- . Nhấp vào Button **Thoát**: dừng chương trình.

\* Hướng dẫn:

- Thiết kế form theo yêu cầu.

- DataGridView có thuộc tính:

- + Name: dgvKHACHHANG
- + Nhấp phải chuột lên DataGridView chọn Edit Columns ...
- + Trong hộp thoại **Edit Columns**, nhấp Add để mở hộp thoại **Add column** và các column (cột) theo danh sách sau:

Name	Type	Header text	DataPropertyName	Width	Ghi chú
MaKH	TextBox	Mã KH	MaKH	100	Frozen
TenCty	TextBox	Tên Cty	TenCty	250	
DiaChi	TextBox	Địa chỉ	DiaChi	200	
ThanhPho	TextBox	Thành Phố	ThanhPho	100	
DienThoai	TextBox	Ngày Nhận Hàng	DienThoai	100	

Trong đó:

TextBox: là DataGridViewTextBoxColumn

CheckBox: là DataGridViewCheckBoxColumn

ComboBox: là DataGridViewComboBoxColumn

Thuộc tính **DataPropertyName**, **Width** điều chỉnh trong hộp thoại **Edit Column**.

- Khai báo namespace sử dụng:

```
using System.Data.SqlClient;
```

- Ở mức class, khai báo:

```
// Chuỗi kết nối
string strConnectionString = "Data Source=PC-PC;Initial Catalog=QuanLyBanHang;Integrated Security=True";
// Đối tượng kết nối
SqlConnection conn = null;
// Đối tượng đưa dữ liệu vào DataTable dtKhachHang
SqlDataAdapter daKhachHang = null;
// Đối tượng hiển thị dữ liệu lên Form
```

```

DataTable dtKhachHang = null;
- Form load:
try
{
    // Khởi động connection
    conn = new SqlConnection(strConnectionString);
    // Vận chuyển dữ liệu lên DataTable dtKhachHang
    daKhachHang = new SqlDataAdapter("SELECT * FROM KHACHHANG", conn);
    dtKhachHang = new DataTable();
    dtKhachHang.Clear();
    daKhachHang.Fill(dtKhachHang);
    // Đưa dữ liệu lên DataGridView
    dgvKHACHHANG.DataSource = dtKhachHang;
}
catch (SqlException)
{
    MessageBox.Show("Không lấy được nội dung trong table KHACHHANG. Lỗi rồi!!!");
}
- FormClosing:
// Giải phóng tài nguyên
dtKhachHang.Dispose();
dtKhachHang = null;
// Hủy kết nối
conn = null;
- Button Thoát:
// Khai báo biến traloi
DialogResult traloi;
// Hiện hộp thoại hỏi đáp
traloi = MessageBox.Show("Chắc không?", "Trả lời",
    MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
// Kiểm tra có nhấp chọn nút Ok không?
if (traloi == DialogResult.OK) Application.Exit();

```

### 3. Đưa dữ liệu vào ComboBox trong DataGridView

**Ví dụ 10.3:** Từ Ví Dụ 10.2, bổ sung button

**ReLoad** (btnReLoad): load lại nội dung của table KhachHang vào DataGridView.

SELECT - INSERT - UPDATE - DELETE

	Mã KH	Tên CTy	Địa Chỉ	Thành Phố	Điện Thoại
▶	AGROMAS	Cơ Điện Nông Nghiệp Q.3	311 Hai Bà Trưng P8 Q3	TP HCM	088970364
	ALSIMES	Giày An Lạc	761 Trần Hưng Đạo P1	Huế	0548456005
	ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
	ASECO	Giày May 3/2	811 Trần Hưng Đạo P1	Hà Nội	048951320
	ATC	Sản Xuất Hàng Mỹ Thuật	7 Trang Tử P14	Nha Trang	0588512230
	BUMEM	Xây Dựng Bình Minh	155 Tô Ô Hiển Thành	Cần Thơ	0718547896
	CEMACO	Hóa Chất Vật Liệu Điện	282 Trần Hưng Đạo P11	Cần Thơ	0718450057
	CINOTEC	Điện Toán Sài Gòn	43 Yết Kiêu P9	Cần Thơ	0718931752
	CODACO	Cơ Khi Dân Dụng	534 Lê Văn Sỹ P14	Nha Trang	0588647207
	COEIDEC	Điện Thiết Bị Công Nghiệp	04 Phan Đăng Lưu	Đà Nẵng	010452710

III

ReLoad      Thoát

\* Hướng dẫn:

- Ở mức class, bổ sung đoạn code và khai báo hàm LoadData() như sau:

```
// Đổi tượng đưa dữ liệu vào DataTable dtThanhPho
SqlDataAdapter daThanhPho = null;
// Đổi tượng hiển thị dữ liệu lên Form
DataTable dtThanhPho = null;

void LoadData()
{
    try
    {
        // Khởi động connection
        conn = new SqlConnection(strConnectionString);
        // Vận chuyển dữ liệu vào DataTable dtThanhPho
        daThanhPho = new SqlDataAdapter("SELECT * FROM THANHPHO", conn);
        dtThanhPho = new DataTable();
        dtThanhPho.Clear();
        daThanhPho.Fill(dtThanhPho);
        // Đưa dữ liệu lên ComboBox trong DataGridView
        (dgvKHACHHANG.Columns[ "ThanhPho" ] as
        DataGridViewComboBoxColumn).DataSource = dtThanhPho;
        (dgvKHACHHANG.Columns[ "ThanhPho" ] as
        DataGridViewComboBoxColumn).DisplayMember =
        "TenThanhPho";
        (dgvKHACHHANG.Columns[ "ThanhPho" ] as
        DataGridViewComboBoxColumn).ValueMember =
        "ThanhPho";
        // Vận chuyển dữ liệu vào DataTable dtKhachHang
        daKhachHang = new SqlDataAdapter("SELECT * FROM KHACHHANG", conn);
        dtKhachHang = new DataTable();
        dtKhachHang.Clear();
        daKhachHang.Fill(dtKhachHang);
        // Đưa dữ liệu lên DataGridView
```

```

        dgvKHACHHANG.DataSource = dtKhachHang;
    }
    catch (SqlException)
    {
        MessageBox.Show("Không lấy được nội dung trong table KHACHHANG. Lỗi rồi!!!");
    }
}
- Form Load: bỏ nội dung cũ, thay bằng LoadData();
- Reload: Nhấp đúp vào Button Reload, viết đoạn code như sau LoadData();

```

### III. Các thao tác trên dữ liệu: Thêm – Sửa - Xóa

**1. Ví dụ 10.4:** Từ Ví dụ 10.3, bổ sung button sau Xóa (btnXoa): xóa record hiện hành ra khỏi table KhachHang.

	Mã KH	Tên CTy	Địa Chỉ	Thành Phố	Điện Thoại
▶	AGROMAS	Co Đien Nông Nghiep Q.3	311 Hai Bà Trưng P8 Q3	TP HCM	088970364
	ALSIMES	Giày An Lạc	761 Trần Hưng Đạo P1	Huế	0548456005
	ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
	ASECO	Giày May 3/2	811 Trần Hưng Đạo P1	Hà Nội	048951320
	ATC	Sàn Xuất Hàng Mỹ Thuật	7 Trang Tử P14	Nha Trang	0588512230
	BUMEM	Xây Dựng Bình Minh	155 Tô Hiến Thành	Cần Thơ	0718547896
	CEMACO	Hóa Chất Vật Liệu Điện	282 Trần Hưng Đạo P11	Cần Thơ	0718450057
	CINOTEC	Điện Toán Sài Gòn	43 Yết Kiêu P9	Cần Thơ	0718931752
	CODACO	Cơ Khí Dân Dụng	534 Lê Văn Sỹ P14	Nha Trang	0588647207
	COFIDEC	Phát Triển Kinh Tế DnVN H&E	94 Điện Biên Phủ	Hà Nội	048453710

\* Hướng dẫn:

- Xóa: Nhấp đúp vào Button Xóa, thực hiện như sau

```

// Mở kết nối
conn.Open();
try
{
    // Thực hiện lệnh
    SqlCommand cmd = new SqlCommand();
    cmd.Connection = conn;
    cmd.CommandType = CommandType.Text;
    // Lấy thứ tự record hiện hành
    int r = dgvKHACHHANG.CurrentCell.RowIndex;
    // Lấy MaKH của record hiện hành
    string strMAKH =
        dgvKHACHHANG.Rows[r].Cells[0].Value.ToString();
    // Viết câu lệnh SQL
    cmd.CommandText = System.String.Concat("Delete From
    KhachHang Where MaKH=' " + strMAKH + " '");
    cmd.CommandType = CommandType.Text;
    // Thực hiện câu lệnh SQL

```

```

        cmd.ExecuteNonQuery();
        // Cập nhật lại DataGridView
        LoadData();
        // Thông báo
        MessageBox.Show("Đã xóa xong!");
    }
    catch (SqlException)
    {
        MessageBox.Show("Không xóa được. Lỗi rồi!!!!");
    }
    // Đóng kết nối
    conn.Close();
}

```

## 2. Ví dụ 10.5:

Từ Ví dụ 10.4, bổ sung các đối tượng như sau ([xem hình](#))

- Panel, trong đó có
  - + 4 TextBox: txtMaKH, txtTenCty, txtDiachi, txtDienthoai
  - + 1 ComboBox: cbThanhpho
- 4 Button: btnThem, btnSua, btnLuu, btnHuy
- \* Yêu cầu:
  - Thiết kế không cho người dùng thao tác (Enabled = false) khi load form:
    - + Panel
    - + Các Button **Lưu, Hủy**
  - Điều chỉnh lại thuộc tính (properties) của DataGridView
    - + AllowUserToAddRows = False
    - +EditMode = EditProgrammatically
  - Khi Form load: đưa dữ liệu từ table KhachHang lên DataGridView (dgvKHACHHANG), có bẫy lỗi.
  - Nhấp vào Button **Reload**: load lại nội dung table KhachHang lên dgvKHACHHANG
  - Nhấp vào Button **Thêm**:
    - + Xóa trống các đối tượng trong Panel.
    - + Cho phép nhập thông tin khách hàng vào các đối tượng trên Panel
    - + Không cho phép thao tác trên các Button: **Thêm, Sửa, Xóa, Thoát**.
    - + Cho phép thao tác trên các Button: **Lưu, Hủy**.
  - Nhấp vào Button **Sửa**:
    - + Đưa thông tin của khách hàng đang được chọn trong DataGridView lên Panel.
    - + Cho phép nhập / sửa thông tin khách hàng vào / trong các đối tượng trên Panel
    - + Không cho phép thao tác trên các Button: **Thêm, Sửa, Xóa, Thoát**.
    - + Cho phép thao tác trên các Button: **Lưu, Hủy**.
  - Nhấp vào Button **Lưu**.
    - + Insert / Update thông tin khách hàng từ Panel vào table KhachHang.
    - + ReLoad lại DataGridView
  - Nhấp vào Button **Hủy**:
    - + Xóa trống các đối tượng trong Panel.
    - + Không cho phép nhập thông tin khách hàng vào các đối tượng trên Panel

SELECT - INSERT - UPDATE - DELETE

Mã KH:	<input type="text"/>	Thành Phố:	<input type="text"/>
Tên CTy:	<input type="text"/>		
Địa Chỉ:	<input type="text"/>		

	Mã KH	Tên CTy	Địa Chỉ	Thành Phố	Điện Thoại
▶	AGROMAS	Cơ Điện Nông Nghiệp Q.3	311 Hai Bà Trưng P8 Q3	TP HCM	088970364
	ALSIMES	Giày An Lạc	761 Trần Hưng Đạo P1	Huế	0548456005
	ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
	ASECO	Giày May 3/2	811 Trần Hưng Đạo P1	Hà Nội	048951320
	ATC	Sản Xuất Hàng Mỹ Thuật	7 Tràng Tử P14	Nha Trang	0588512230
	BUMEM	Xây Dựng Bình Minh	155 Tô Hiến Thành	Cần Thơ	0718547896
	CEMACO	Hóa Chất Vật Liệu Điện	282 Trần Hưng Đạo P11	Cần Thơ	0718450057
	CINOTEC	Điện Toán Sài Gòn	43 Yết Kiêu P9	Cần Thơ	0718931752
	CODACO	Cơ Khí Dân Dụng	534 Lê Văn Sỹ P14	Nha Trang	0588647207

\* Hướng dẫn:

- Thiết kế bổ sung Panel với 4 TextBox và 1 ComboBox, Panel có Enabled = false
- Ở mức class, bổ sung khai báo biến như sau:

```
// Khai báo biến kiểm tra việc Thêm hay Sửa dữ liệu
bool Them;
```

- Hàm LoadData(): bổ sung dưới hàng

```
// Đưa dữ liệu lên DataGridView
dgvKHACHHANG.DataSource = dtKhachHang;
```

đoạn code sau

```
// Xóa trống các đối tượng trong Panel
this.txtMaKH.ResetText();
this.txtTenCty.ResetText();
this.txtDiaChi.ResetText();
this.txtDienThoai.ResetText();
// Không cho thao tác trên các nút Lưu / Hủy
this.btnLuu.Enabled = false;
this.btnHuy.Enabled = false;
this.panel.Enabled = false;
// Cho thao tác trên các nút Thêm / Sửa / Xóa / Thoát
this.btnThem.Enabled = true;
this.btnSua.Enabled = true;
this.btnXoa.Enabled = true;
this.btnThoat.Enabled = true;
```

- **Thêm:** nhấp đúp vào button Thêm, bổ sung đoạn code

```
// Kích hoạt biến Them
Them = true;
// Xóa trống các đối tượng trong Panel
this.txtMaKH.ResetText();
this.txtTenCty.ResetText();
this.txtDiaChi.ResetText();
```

```

this.txtDienThoai.ResetText();
// Cho thao tác trên các nút Lưu / Hủy / Panel
this.btnLuu.Enabled = true;
this.btnHuy.Enabled = true;
this.panel.Enabled = true;
// Không cho thao tác trên các nút Thêm / Xóa / Thoát
this.btnThem.Enabled = false;
this.btnSua.Enabled = false;
this.btnXoa.Enabled = false;
this.btnExit.Enabled = false;
// Đưa dữ liệu lên ComboBox
this.cbThanhPho.DataSource = dtThanhPho;
this.cbThanhPho.DisplayMember = "TenThanhPho";
this.cbThanhPho.ValueMember = "ThanhPho";
// Đưa con trỏ đến TextField txtMaKH
this.txtMaKH.Focus();

```

	Mã KH	Tên CTy	Địa Chỉ	Thành Phố	Điện Thoại
▶	AGROMAS	Cô Điển Nông Nghiệp Q.3	311 Hai Bà Trưng P8 Q3	TP HCM	088970364
	ALSIMES	Giày An Lạc	761 Trần Hưng Đạo P1	Huế	0548456005
	ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
	ASECO	Giày May 3/2	811 Trần Hưng Đạo P1	Hà Nội	048951320
	ATC	Sàn Xuất Hàng Mỹ Thuật	7 Trang Tử P14	Nha Trang	0588512230
	BUMEM	Xây Dựng Bình Minh	155 Tô Hiến Thành	Cần Thơ	0718547896
	CEMACO	Hóa Chất Vật Liệu Điện	282 Trần Hưng Đạo P11	Cần Thơ	0718450057
	CINOTEC	Điện Toán Sài Gòn	43 Yết Kiêu P9	Cần Thơ	0718931752
	CODACO	Cơ Khí Dân Dụng	534 Lê Văn Sỹ P14	Nha Trang	0588647207

- **Sửa:** Nhấp đúp vào button Sửa, bổ sung đoạn code

```

// Kích hoạt biến Sửa
Them = false;
// Đưa dữ liệu lên ComboBox
this.cbThanhPho.DataSource = dtThanhPho;
this.cbThanhPho.DisplayMember = "TenThanhPho";
this.cbThanhPho.ValueMember = "ThanhPho";
// Cho phép thao tác trên Panel
this.panel.Enabled = true;
// Thứ tự dòng hiện hành
int r = dgvKHACHHANG.CurrentCell.RowIndex;
// Chuyển thông tin lên panel
this.txtMaKH.Text =
    dgvKHACHHANG.Rows[r].Cells[0].Value.ToString();

```

```

this.txtTenCty.Text =
    dgvKHACHHANG.Rows[r].Cells[1].Value.ToString();
this.txtDiaChi.Text =
    dgvKHACHHANG.Rows[r].Cells[2].Value.ToString();
this.cbThanhPho.SelectedValue =
    dgvKHACHHANG.Rows[r].Cells[3].Value.ToString();
this.txtDienThoai.Text =
    dgvKHACHHANG.Rows[r].Cells[4].Value.ToString();
// Cho thao tác trên các nút Lưu / Hủy / Panel
this.btnLuu.Enabled = true;
this.btnHuy.Enabled = true;
this.panel.Enabled = true;
// Không cho thao tác trên các nútThêm / Xóa / Thoát
this.btnThem.Enabled = false;
this.btnSua.Enabled = false;
this.btnXoa.Enabled = false;
this.btnThoat.Enabled = false;
// Đưa con trỏ đến TextField txtMaKH
this.txtMaKH.Focus();

```

SELECT - INSERT - UPDATE - DELETE

Mã KH:	AGROMAS	Thành Phố:	TP HCM
Tên CTy:	Co Dien Nong Nghiep Q.3	Điện Thoại:	088970364
Địa Chỉ:	311 Hai Bà Trưng P8 Q3		

Mã KH	Tên CTy	Địa Chi	Thành Phố	Điện Thoại
AGROMAS	Co Dien Nong Nghiep Q.3	311 Hai Bà Trưng P8 Q3	TP HCM	088970364
ALSIMES	Giày An Lạc	761 Trần Hưng Đạo P1	Huế	0548456005
ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
ASECO	Giày May 3/2	811 Trần Hưng Đạo P1	Hà Nội	048951320
ATC	Sản Xuất Hàng Mỹ Thuật	7 Trang Tử P14	Nha Trang	0588512230
BUMEM	Xây Dựng Bình Minh	155 Tô Hiến Thành	Cần Thơ	0718547896
CEMACO	Hóa Chất Vật Liệu Điện	282 Trần Hưng Đạo P11	Cần Thơ	0718450057
CINOTEC	Điện Toán Sài Gòn	43 Yết Kiêu P9	Cần Thơ	0718931752
CODACO	Cơ Khí Dân Dụng	534 Lê Văn Sỹ P14	Nha Trang	0588647207

- **Lưu:** Nhấp đúp vào button Lưu, bổ sung đoạn code

```

// Mở kết nối
conn.Open();
// Thêm dữ liệu
if (Them)
{
    try
    {
        // Thực hiện lệnh
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = conn;

```

```

        cmd.CommandType = CommandType.Text;
        // Lệnh Insert InTo
        cmd.CommandText = System.String.Concat("Insert
        Into KhachHang Values(" + "'' +
        this.txtMaKH.Text.ToString() + "','" +
        this.txtTenCty.Text.ToString() + "','" +
        this.txtDiaChi.Text.ToString() + "','" +
        this.cbThanhPho.SelectedValue.ToString() +
        "','" + this.txtDienThoai.Text.ToString() +
        "')");
        cmd.CommandType = CommandType.Text;
        cmd.ExecuteNonQuery();
        // Load lại dữ liệu trên DataGridView
        LoadData();
        // Thông báo
        MessageBox.Show("Đã thêm xong!");
    }
    catch (SqlException)
    {
        MessageBox.Show("Không thêm được. Lỗi rồi!");
    }
}
if (!Them)
{
    try
    {
        // Thực hiện lệnh
        SqlCommand cmd = new SqlCommand();
        cmd.Connection = conn;
        cmd.CommandType = CommandType.Text;
        // Thứ tự dòng hiện hành
        int r = dgvKHACHHANG.CurrentCell.RowIndex;
        // MaKH hiện hành
        string strMAKH =
        dgvKHACHHANG.Rows[r].Cells[0].Value.ToString();
        // Câu lệnh SQL
        cmd.CommandText = System.String.Concat("Update
        KhachHang Set TenCty=' " +
        this.txtTenCty.Text.ToString() + "', DiaChi=' "
        + this.txtDiaChi.Text.ToString() + ',
        ThanhPho=' +
        this.cbThanhPho.SelectedValue.ToString() + ',
        DienThoai=' +
        this.txtDienThoai.Text.ToString() + ' Where
        MaKH=' + strMAKH + "''");
        // Cập nhật
        cmd.CommandType = CommandType.Text;
        cmd.ExecuteNonQuery();
        // Load lại dữ liệu trên DataGridView
    }
}

```

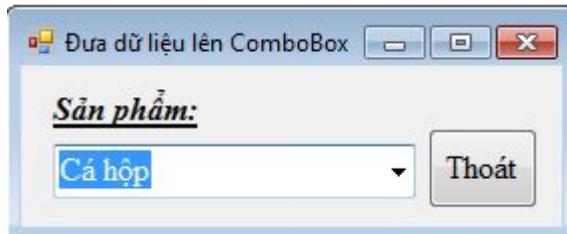
```

        LoadData();
        // Thông báo
        MessageBox.Show("Đã sửa xong!");
    catch (SqlException)
    {
        MessageBox.Show("Không sửa được. Lỗi rồi!");
    }
}
// Đóng kết nối
conn.Close();
- Hủy: Nhấp đúp vào button Hủy, bổ sung đoạn code
// Xóa trống các đối tượng trong Panel
this.txtMaKH.ResetText();
this.txtTenCity.ResetText();
this.txtDiaChi.ResetText();
this.txtDienThoai.ResetText();
// Cho thao tác trên các nútThêm / Sửa / Xóa / Thoát
this.btnThem.Enabled = true;
this.btnSua.Enabled = true;
this.btnXoa.Enabled = true;
this.btnThoat.Enabled = true;
// Không cho thao tác trên các nút Lưu / Hủy / Panel
this.btnLuu.Enabled = false;
this.btnHuy.Enabled = false;
this.panel.Enabled = false;

```

## Bài Tập

- Thiết kế form như sau:



(cbSanpham, btnThoat)

- \* Yêu cầu:
  - . Form load: đưa dữ liệu từ table SanPham lên ComboBox (cbSanPham), kèm bẫy lỗi.
  - . Nhấp vào Button **Thoát**: dừng chương trình (có hiện hộp thoại hỏi / đáp).
- 2. Đưa dữ liệu lên DataGridView (như II. 2) cho các table: NhanVien, SanPham.
- 3. Thiết kế form (như III) thực hiện các thao tác: Thêm, Sửa, Xóa cho các table: HoaDon, ChiTietHoaDon. Lưu ý: Các field MaKH, MaNV, MaSP thể hiện dưới dạng ComboBox.

-- oOo --

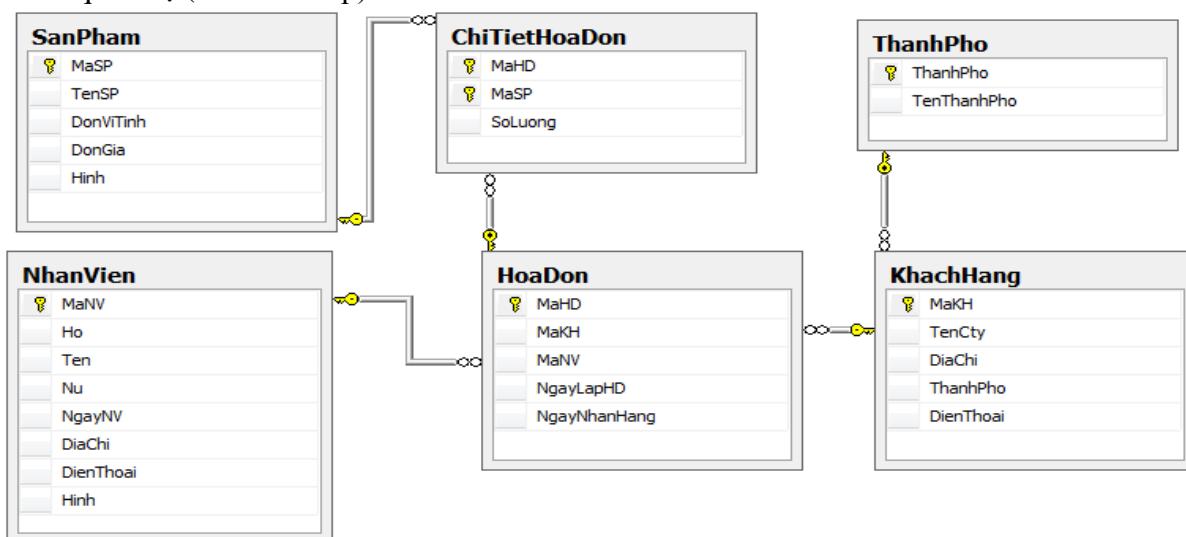
## Chương 7: XÂY DỰNG ỨNG DỤNG

### Bài 12-13-14-15: ỨNG DỤNG QUẢN LÝ BÁN HÀNG

#### I. Chuẩn bị:

- Tên máy được sử dụng (**SERVERNAME**) là PC-PC (thay đổi cho đúng máy đang dùng!)
- Database được sử dụng (**DATABASENAME**) là **QuanLyBanHang**, gồm có các table sau:
 

+ ThanhPho	+ Sanpham
+ Khachhang	+ HoaDon
+ Nhanvien	+ ChitietHoaDon
- Với quan hệ (Relationship) như sau:



- Khởi động SQL Server 2008, tạo một DataBase mới có tên **QuanLyBanHang**, import dữ liệu từ file QuanLyBanHang.mdb
- Khởi động Visual Studio 2008, tạo một project mới (Windows Forms Application), lưu với tên **QUANLYBANHANG**

#### II. Sử dụng control:

**Câu 01: (Form1 – Màn hình chính)** Thiết kế như sau



\* Yêu cầu:

Khi Form1 được thực hiện sẽ thể hiện màn hình “Đăng nhập hệ thống” là Form2.

\* Hướng dẫn:

- Ở mức class, khai báo hàm frmLogin() như sau:

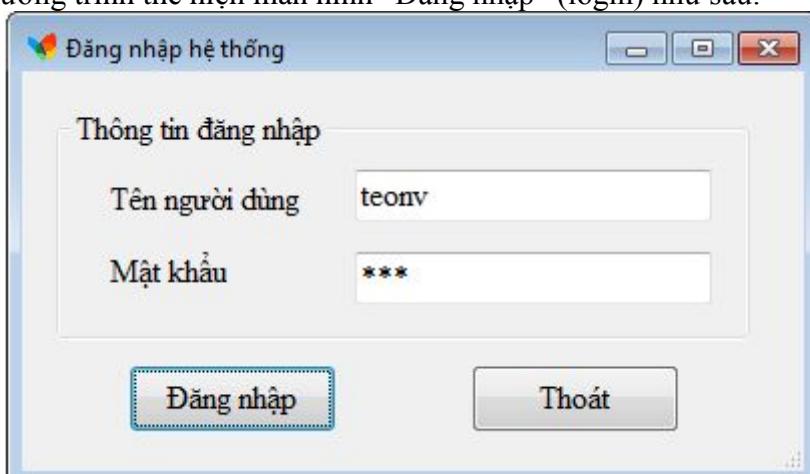
```
Form frm = new Form2();
frm.ShowDialog();
```

- Form Load:

```
frmlogin();
```

### Câu 02: (Form2 – Màn hình Đăng nhập - login)

Viết chương trình thể hiện màn hình “Đăng nhập” (login) như sau:



(txtUser, txtPass, btnDangNhap, btnThoat)

\* Yêu cầu:

- Khi nhấp vào nút **Đăng nhập** (btnDangNhap) sẽ thực hiện kiểm tra:
  - + Nếu txtUser = "teonv" và txtPass = "123" thì chuyển sang Form1.
  - + Ngược lại thông báo "**Không đúng tên người dùng / mật khẩu !!!**"
- Nhấp button **Thoát** thì hiển thị thông báo "Chắc không?"
- + Nếu chọn Yes thì kết thúc chương trình.
- + Ngược lại thì trở lại màn hình Đăng nhập hệ thống.

\* Hướng dẫn:

- **Đăng nhập:** Nhấp đúp vào button btnDangnhap, gõ vào đoạn code sau
 

```
if ((this.txtUser.Text == "teonv") && (this.txtPass.Text == "123"))
    this.Close();
else
{
    MessageBox.Show("Không đúng tên người dùng / mật
    khẩu !!!", "Thông báo");
    this.txtUser.Focus();
}
```
- **Thoát:** Nhấp đúp vào button btnThoat, gõ vào đoạn code sau
 

```
DialogResult traloi;
traloi = MessageBox.Show("Chắc không?", "Trả lời",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
if (traloi == DialogResult.OK)
    Application.Exit();
```

### Câu 03: (Form1 – Hệ thống Menu)

Thêm vào Form1, hệ thống menu như sau



\* Hướng dẫn: Sử dụng ToolStrip

#### **Câu 04: (Form1 – Menu Hệ thống)**

\* Yêu cầu: Trên Form1, viết lệnh cho menu hệ thống như sau

- Khi chọn menu “**Hệ thống \ Đăng nhập**” sẽ thực hiện đăng nhập lại hệ thống (Form2).
- Khi chọn menu “**Hệ thống \ Thoát**” sẽ dừng chương trình.

\* Hướng dẫn:

- Menu “**Hệ thống \ Đăng nhập**”: Nhấp đúp vào menu “Hệ thống \ Đăng nhập”, gõ vào đoạn code sau:

```
frmlogin();
```

- Menu “**Hệ thống \ Thoát**”: Nhấp đúp vào menu “Hệ thống \ Đăng nhập”, gõ vào đoạn code sau:

```
DialogResult traloi;
traloi = MessageBox.Show("Chắc không?", "Trả lời",
MessageBoxButtons.OKCancel, MessageBoxIcon.Question);
if (traloi == DialogResult.OK)
    Application.Exit();
```

### **III. Sử dụng DataBase:**

#### **Câu 05:**

##### **a) (Form1 – Menu Danh mục)**

\* Yêu cầu: Trên Form1, viết lệnh cho menu danh mục như sau

- Viết hàm XemDanhMuc(int intDanhMuc) thực hiện mở Form3 và gán Form3.Text = intDanhMuc.

- Khi chọn menu “**Xem Danh mục \ <int>**” sẽ thực hiện gọi hàm XemDanhMuc(<int>), với <int> là giá trị tương ứng với danh mục được chọn.

\* Hướng dẫn:

- Ở mức class, khai báo hàm XemDanhMuc(int intDanhMuc) như sau:

```
// Hàm xem danh mục
void XemDanhMuc(int intDanhMuc)
{
    Form frm = new Form3();
    frm.Text = intDanhMuc.ToString();
    frm.ShowDialog();
}
```

- Menu “**Xem Danh mục \ Danh mục Thành Phố**”: Nhấp đúp vào menu “Xem Danh mục \ Danh mục Thành Phố”, gõ vào đoạn code sau:

```
XemDanhMuc(1);
```

- Menu “**Xem Danh mục \ Danh mục Khách Hàng**”: Nhấp đúp vào menu “Xem Danh mục \ Danh mục Khách Hàng”, gõ vào đoạn code sau:

```
XemDanhMuc(2);
```

- Menu “**Xem Danh mục \ Danh mục Nhân Viên**”: Nhấp đúp vào menu “Xem Danh mục \ Danh mục Nhân Viên”, gõ vào đoạn code sau:

```
XemDanhMuc(3);
```

- Menu “**Xem Danh mục \ Danh mục Sản Phẩm**”: Nhấp đúp vào menu “Xem Danh mục \ Danh mục Sản Phẩm”, gõ vào đoạn code sau:

```
XemDanhMuc(4);
```

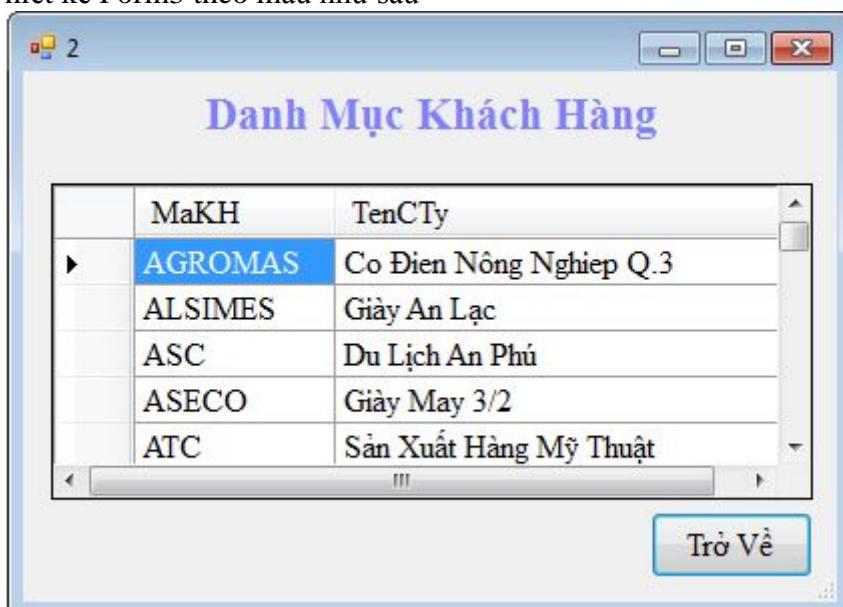
- Menu “**Xem Danh mục \ Danh mục Hóa Đơn**”: Nhấp đúp vào menu “Xem Danh mục \ Danh mục Hóa Đơn”, gõ vào đoạn code sau:

```
XemDanhMuc( 5 );
```

- Menu “**Xem Danh mục \ Danh mục Chi Tiết Hóa Đơn**”: Nhấp đúp vào menu “Xem Danh mục \ Danh mục Chi Tiết Hóa Đơn”, gõ vào đoạn code sau:

```
XemDanhMuc( 6 );
```

b) **(Form3)** Thiết kế Form3 theo mẫu như sau



(lblDanhMuc, dgvDANHMUC, btnTroVe)

\* Yêu cầu: Trên Form3

- Khi Form được load sẽ hiển thị tên table (được chọn) lên Label lblDanhMuc và nội dung của table này lên DataGridView dgvDANHMUC.

- Khi nhấp vào button Trở Về sẽ đóng Form3.

\* Hướng dẫn:

- Khai báo namespace sử dụng:

```
using System.Data.SqlClient;
```

- Ở mức class, khai báo như sau:

```
// Chuỗi kết nối
string strConnectionString = "Data Source=PC-PC;Initial Catalog=QuanLyBanHang;Integrated Security=True";
// Đối tượng kết nối
SqlConnection conn = null;
// Đối tượng đưa dữ liệu vào DataTable dtTable
SqlDataAdapter dataTable = null;
// Đối tượng hiển thị dữ liệu lên Form
DataTable dtTable = null;
```

- Form Load

```
try
{
    // Khởi động connection
    conn = new SqlConnection(strConnectionString);
    // Xử lý danh mục
    int intDM = Convert.ToInt32(this.Text);
    switch (intDM)
```

```

    {
        case 1:
            lblDM.Text = "Danh Mục Thành Phố";
            dataTable = new SqlDataAdapter("SELECT
                ThanhPho, TenThanhPho FROM THANHPHO",
                conn);
            break;
        case 2:
            lblDM.Text = "Danh Mục Khách Hàng";
            dataTable = new SqlDataAdapter("SELECT MaKH,
                TenCTy FROM KHACHHANG", conn);
            break;
        case 3:
            lblDM.Text = "Danh Mục Nhân Viên";
            dataTable = new SqlDataAdapter("SELECT MaNV,
                Ho, Ten FROM NHANVIEN", conn);
            break;
        case 4:
            lblDM.Text = "Danh Mục Sản Phẩm";
            dataTable = new SqlDataAdapter("SELECT MaSP,
                TenSP, DonViTinh, DonGia FROM SANPHAM",
                conn);
            break;
        case 5:
            lblDM.Text = "Danh Mục Hóa Đơn";
            dataTable = new SqlDataAdapter("SELECT MaHD,
                MaKH, MaNV FROM HOADON", conn);
            break;
        case 6:
            lblDM.Text = "Danh Mục Chi Tiết Hóa Đơn";
            dataTable = new SqlDataAdapter("SELECT *
                FROM CHITIETHOADON", conn);
            break;
        default:
            break;
    }
    // Vận chuyển dữ liệu lên DataTable dtTable
    dtTable = new DataTable();
    dtTable.Clear();
    dataTable.Fill(dtTable);
    // Đưa dữ liệu lên DataGridView
    dgvDANHMUC.DataSource = dtTable;
    // Thay đổi độ rộng cột
    dgvDANHMUC.AutoResizeColumns();
}
catch (SqlException)
{
    MessageBox.Show("Không lấy được nội dung trong
        table. Lỗi rồi!!!!");
}

```

```
}
```

- Trở Về: Nhấp đúp vào button Trở Về, thêm vào đoạn code sau

```
this.Close();
```

### Câu 06: (Form4 – Menu Quản lý danh mục đơn \ Danh mục Thành Phố )

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục đơn \ Danh mục Thành Phố” sẽ mở Form4.
- Trên Form4: Thiết kế như sau:



(panel, txtThanhPho, txtTenThanhPho, dgvTHANHPHO, btnReLoad, btnThem, btnSua, btnXoa, btnLuu, btnHuyBo, btnTroVe)

\* Hướng dẫn:

#### a) (Form1 – Menu Quản lý danh mục đơn)

- Menu “Quản lý danh mục đơn \ Danh mục Thành Phố”: Nhấp đúp vào menu “Quản lý danh mục đơn \ Danh mục Thành Phố”, gõ vào đoạn code sau:

```
Form frm = new Form4();
frm.Text = "Quản lý Danh mục Thành Phố";
frm.ShowDialog();
```

#### b) (Form4) Thiết kế Form4 như mẫu

- Khai báo namespace sử dụng:

```
using System.Data.SqlClient;
```

- Ở mức class, khai báo như sau:

```
// Chuỗi kết nối
string strConnectionString = "Data Source=PC-PC;Initial
Catalog=QuanLyBanHang;Integrated Security=True";
// Đối tượng kết nối
SqlConnection conn = null;
```

```

// Đôi tượng đưa dữ liệu vào DataTable dtThanhPho
SqlDataAdapter daThanhPho = null;
// Đôi tượng hiển thị dữ liệu lên Form
DataTable dtThanhPho = null;
// Khai báo biến kiểm tra việc Thêm hay Sửa dữ liệu
bool Them;

void LoadData()
{
    try
    {
        // Khởi động connection
        conn = new SqlConnection(strConnectionString);
        // Vận chuyển dữ liệu lên DataTable dtThanhPho
        daThanhPho = new SqlDataAdapter("SELECT * FROM THANHPHO", conn);
        dtThanhPho = new DataTable();
        dtThanhPho.Clear();
        daThanhPho.Fill(dtThanhPho);
        // Đưa dữ liệu lên DataGridView
        dgvTHANHPHO.DataSource = dtThanhPho;
        // Thay đổi độ rộng cột
        dgvTHANHPHO.AutoResizeColumns();
        // Xóa trống các đối tượng trong Panel
        this.txtThanhPho.ResetText();
        this.txtTenThanhPho.ResetText();
        // Không cho thao tác trên các nút Lưu / Hủy
        this.btnLuu.Enabled = false;
        this.btnHuyBo.Enabled = false;
        this.panel.Enabled = false;
        // Cho thao tác trên các nút Thêm / Sửa / Xóa /
        // Thoát
        this.btnThem.Enabled = true;
        this.btnSua.Enabled = true;
        this.btnXoa.Enabled = true;
        this.btnTroVe.Enabled = true;
    }
    catch (SqlException)
    {
        MessageBox.Show("Không lấy được nội dung trong
table THANHPHO. Lỗi rồi!!!");
    }
}
- Form Load:
LoadData();
- FormClosing:
// Giải phóng tài nguyên
dtThanhPho.Dispose();
dtThanhPho = null;

```

```
// Hủy kết nối
conn = null;
- ReLoad: Nhấp đúp vào button ReLoad, thêm vào đoạn code sau
    LoadData();
- Trở Về: Nhấp đúp vào button Trở Về, thêm vào đoạn code sau
    this.Close();
- Thêm: Nhấp đúp vào button Thêm, thêm vào đoạn code sau
    // Kích hoạt biến Them
    Them = true;
    // Xóa trống các đối tượng trong Panel
    this.txtThanhPho.ResetText();
    this.txtTenThanhPho.ResetText();
    // Cho thao tác trên các nút Lưu / Hủy / Panel
    this.btnLuu.Enabled = true;
    this.btnHuyBo.Enabled = true;
    this.panel.Enabled = true;
    // Không cho thao tác trên các nút Thêm / Xóa / Thoát
    this.btnThem.Enabled = false;
    this.btnSua.Enabled = false;
    this.btnXoa.Enabled = false;
    this.btnTroVe.Enabled = false;
    // Đưa con trỏ đến TextField txtThanhPho
    this.txtThanhPho.Focus();
- Sửa: Nhấp đúp vào button Sửa, thêm vào đoạn code sau
    // Kích hoạt biến Sửa
    Them = false;
    // Cho phép thao tác trên Panel
    this.panel.Enabled = true;
    // Thú tự dòng hiện hành
    int r = dgvTHANHPHO.CurrentCell.RowIndex;
    // Chuyển thông tin lên panel
    this.txtThanhPho.Text =
        dgvTHANHPHO.Rows[r].Cells[0].Value.ToString();
    this.txtTenThanhPho.Text =
        dgvTHANHPHO.Rows[r].Cells[1].Value.ToString();
    // Cho thao tác trên các nút Lưu / Hủy / Panel
    this.btnLuu.Enabled = true;
    this.btnHuyBo.Enabled = true;
    this.panel.Enabled = true;
    // Không cho thao tác trên các nút Thêm / Xóa / Thoát
    this.btnThem.Enabled = false;
    this.btnSua.Enabled = false;
    this.btnXoa.Enabled = false;
    this.btnTroVe.Enabled = false;
    // Đưa con trỏ đến TextField txtMaKH
    this.txtThanhPho.Focus();
- Xóa: Nhấp đúp vào button Xóa, thêm vào đoạn code sau
    // Mở kết nối
    conn.Open();
```

```

try
{
    // Thực hiện lệnh
    SqlCommand cmd = new SqlCommand();
    cmd.Connection = conn;
    cmd.CommandType = CommandType.Text;
    // Lấy thứ tự record hiện hành
    int r = dgvTHANHPHO.CurrentCell.RowIndex;
    // Lấy MaKH của record hiện hành
    string strTHANHPHO =
        dgvTHANHPHO.Rows[r].Cells[0].Value.ToString();
    // Viết câu lệnh SQL
    cmd.CommandText = System.String.Concat("Delete From
    ThanhPho Where ThanhPho= '" + strTHANHPHO + "' ");
    cmd.CommandType = CommandType.Text;
    // Thực hiện câu lệnh SQL
    cmd.ExecuteNonQuery();
    // Cập nhật lại DataGridView
    LoadData();
    // Thông báo
    MessageBox.Show("Đã xóa xong!");
}
catch (SqlException)
{
    MessageBox.Show("Không xóa được. Lỗi rồi!");
}
// Đóng kết nối
conn.Close();
- Hủy Bỏ: Nhấp đúp vào button Hủy Bỏ, thêm vào đoạn code sau
// Xóa trống các đối tượng trong Panel
this.txtThanhPho.ResetText();
this.txtTenThanhPho.ResetText();
// Cho thao tác trên các nút Thêm / Sửa / Xóa / Thoát
this.btnThem.Enabled = true;
this.btnSua.Enabled = true;
this.btnXoa.Enabled = true;
this.btnTroVe.Enabled = true;
// Không cho thao tác trên các nút Lưu / Hủy / Panel
this.btnLuu.Enabled = false;
this.btnHuyBo.Enabled = false;
this.panel.Enabled = false;
- Lưu: Nhấp đúp vào button Lưu, thêm vào đoạn code sau
// Mở kết nối
conn.Open();
// Thêm dữ liệu
if (Them)
{
    try
    {

```

```
// Thực hiện lệnh
SqlCommand cmd = new SqlCommand();
cmd.Connection = conn;
cmd.CommandType = CommandType.Text;
// Lệnh Insert Into
cmd.CommandText = System.String.Concat("Insert
Into ThanhPho Values(" + "'' +
this.txtThanhPho.Text.ToString() + "','" +
this.txtTenThanhPho.Text.ToString() + "')");
cmd.CommandType = CommandType.Text;
cmd.ExecuteNonQuery();
// Load lại dữ liệu trên DataGridView
LoadData();
// Thông báo
MessageBox.Show("Đã thêm xong!");
}
catch (SqlException)
{
    MessageBox.Show("Không thêm được. Lỗi rồi!");
}
}
if (!Them)
{
    // Thực hiện lệnh
    SqlCommand cmd = new SqlCommand();
    cmd.Connection = conn;
    cmd.CommandType = CommandType.Text;
    // Thứ tự dòng hiện hành
    int r = dgvTHANHPHO.CurrentCell.RowIndex;
    // MaKH hiện hành
    string strTHANHPHO =
dgvTHANHPHO.Rows[r].Cells[0].Value.ToString();
    // Câu lệnh SQL
    cmd.CommandText = System.String.Concat("Update
ThanhPho Set TenThanhPho=''" +
this.txtTenThanhPho.Text.ToString() + "' Where
ThanhPho=''" + strTHANHPHO + "'");
    // Cập nhật
    cmd.CommandType = CommandType.Text;
    cmd.ExecuteNonQuery();
    // Load lại dữ liệu trên DataGridView
    LoadData();
    // Thông báo
    MessageBox.Show("Đã sửa xong!");
}
// Đóng kết nối
conn.Close();
```

**Câu 07: (Form5 – Menu Quản lý danh mục đơn \ Danh mục Khách Hàng)**

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục đơn \ Danh mục Khách Hàng” sẽ mở Form5.
- Trên Form5: thiết kế như sau

Mã KH	Tên Cty	Địa Chỉ	Thành Phố	Điện Thoại
AGROMAS	Cô Dien Nông Nghiep Q.3	311 Hai Bà Trưng P8 Q3	TP HCM	088970364
ALSIMES	Giày An Lạc	761 Trần Hưng Đạo P1	Huế	0548456005
ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
ASECO	Giày May 3/2	811 Trần Hưng Đạo P1	Hà Nội	048951320
ATC	Sàn Xuất Hàng Mỹ Thuật	7 Tràng Tử P14	Nha Trang	0588512230

Buttons at the bottom: Reload,Thêm,Sửa,Lưu,Hủy Bỏ,Xóa,Trở Về.

#### Câu 08: (Form6 – Menu Quản lý danh mục đơn \ Danh mục Nhân Viên)

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục đơn \ Danh mục Nhân Viên” sẽ mở Form6.
- Trên Form6: thiết kế như sau

Mã NV	Họ Lót	Tên	Nữ	Ngày NV	Địa Chỉ	Điện Thoại	H
2	Hà	Vĩnh	<input type="checkbox"/>	7/12/1991	89 Đặng Khôi Q1	8352074	
3	Trần	Tuyết	<input checked="" type="checkbox"/>	2/27/1991	26 Lê Quý Đôn P6 Q3	8557798	
4	Nguyễn	Kim	<input checked="" type="checkbox"/>	3/30/1992	178 Hậu Giang P6 Q6	8553278	
5	Trương	Duy	<input type="checkbox"/>	9/13/1992	77 Trương Định P6 Q3	8940295	
6	Lương	Bá	<input type="checkbox"/>	9/13/1992	92 Lê Thánh Tôn Q1	8940549	

Buttons at the bottom: Reload,Thêm,Sửa,Lưu,Hủy BỎ,Xóa,Trở Về.

#### Câu 09: (Form7 – Menu Quản lý danh mục đơn \ Danh mục Sản Phẩm)

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục đơn \ Danh mục Sản Phẩm” sẽ mở Form7.
- Trên Form7: thiết kế như sau

Quản lý Danh mục Sản Phẩm

Mã SP:		Đơn vị tính:	
Tên SP:		Đơn giá:	

	MaSP	TenSP	DonViTinh	DonGia	Hình
▶	1	Ruou	Chai	230.5000	<input type="checkbox"/>
	2	Gia vị	Thùng	40.0000	<input type="checkbox"/>
	3	Bánh kem	Cái	10.0000	<input type="checkbox"/>
	4	Bơ	Kg	38.0000	<input type="checkbox"/>
	5	Bánh mì	Cái	8.0000	<input type="checkbox"/>
	6	Nem	Kg	23.7900	<input type="checkbox"/>
	7	Táo	Kg	5.0000	<input type="checkbox"/>
	8	Cà phê	Túi	60.5000	<input type="checkbox"/>

Reload  Thêm   Sửa   Lưu   Hủy Bỏ   Xóa   Trở Về

#### Câu 10: (Form8 – Menu Quản lý danh mục đơn \ Danh mục Hóa Đơn)

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục đơn \ Danh mục Hóa Đơn” sẽ mở Form8.
- Trên Form8: thiết kế như sau

Quản lý Danh mục Hóa Đơn

Mã HD:		Ngày Lập Hóa Đơn:	
Mã KH:		Ngày Lập Nhận Hàng:	
Mã NV:			

	Mã HD	Tên Cty	Mã NV	Ngày Lập HD	Ngày Nhận Hàng
▶	10145	SXKD Dịch Vụ Tổng Hợp	Vương Ngọc Lan	1/9/1992	2/6/1992
	10148	Hóa Nhựa Vĩnh Tiến	Nguyễn Ngọc Nga	1/14/1992	2/11/1992
	10150	Cơ Khí Dân Dụng	Nguyễn Kim Ngọc	1/17/1992	2/28/1992
	10156	Công Nghiệp Mới	Nguyễn Kim Ngọc	1/28/1992	2/25/1992
	10157	Đại Hồng Phát	Nguyễn Kim Ngọc	1/29/1992	2/26/1992

Reload  Thêm   Sửa   Lưu   Hủy Bỏ   Xóa   Trở Về

#### Câu 11: (Form9 – Menu Quản lý danh mục đơn \ Danh mục Chi Tiết Hóa Đơn)

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục đơn \ Danh mục Chi Tiết Hóa Đơn” sẽ mở Form9.

- Trên Form9: thiết kế như sau

	Mã HD	Tên SP	Số Lượng
▶	10156	Ruou	25
	10156	Gia vị	25
	10157	Ruou	35
	10158	Ruou	12
	10158	Bơ	30
	10158	Nem	20
	10158	Cá hộp	12

### Câu 11: (Form10 – Menu Quản lý danh mục theo nhóm \ Khách hàng theo thành phố)

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục theo nhóm \ Khách hàng theo Thành Phố” sẽ mở Form10.
- Trên Form10: thiết kế như sau

	Mã KH	Tên Cty	Địa Chỉ	Thành Phố	Điện Thoại
▶	ASC	Du Lịch An Phú	233 Nguyễn Trãi P2	Nha Trang	0588124780
	ATC	Sàn Xuất Hàng Mỹ Thuật	7 Trang Tử P14	Nha Trang	0588512230
	CODACO	Cơ Khí Dân Dụng	534 Lê Văn Sỹ P14	Nha Trang	0588647207
	LIPHACO	Liên Phát	200 Bến Chuong Dương	Nha Trang	0588321047
	MOVIMEX	Vật Tư Điện Ánh	410 Hầm Tử P5	Nha Trang	0588321808
	SAFICO	Thủy Sản Xuất Khẩu	47 Bãi Sậy P1	Nha Trang	0588650126
	TDE	Thang Máy Tự Động	56 Nguyễn Biểu P2	Nha Trang	0588974562
	VAFACO	Vật Phẩm Văn Hóa	105A Ngõ Quyền P11	Nha Trang	0588654201
	VITICO	Hóa Nhựa Vĩnh Tiến	11 Vạn Tường P13	Nha Trang	0588796540

### Câu 12: (Form11 – Menu Quản lý danh mục theo nhóm \ Hóa đơn theo Khách hàng)

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục theo nhóm \ Hóa đơn theo Khách hàng” sẽ mở Form11.

- Trên Form11: thiết kế như sau

	Mã HD	Mã KH	Mã NV	Ngày Lập HD	Ngày Nhận Hàng
▶	10273	Xây Dựng Cáp Thoát Nước Q2	Trần Tuyết Oanh	6/29/1992	7/27/1992
	10285	Xây Dựng Cáp Thoát Nước Q2	Nguyễn Ngọc Nga	7/14/1992	8/11/1992
	10345	Xây Dựng Cáp Thoát Nước Q2	Hà Vĩnh Phát	9/28/1992	10/26/1992
	10418	Xây Dựng Cáp Thoát Nước Q2	Nguyễn Kim Ngọc	12/11/1992	1/8/1993
	10451	Xây Dựng Cáp Thoát Nước Q2	Nguyễn Kim Ngọc	1/13/1993	1/27/1993
	10515	Xây Dựng Cáp Thoát Nước Q2	Hà Vĩnh Phát	3/17/1993	3/31/1993
	10540	Xây Dựng Cáp Thoát Nước Q2	Trần Tuyết Oanh	4/12/1993	5/10/1993

#### **Câu 13: (Form12 – Menu Quản lý danh mục theo nhóm \ Hóa đơn theo Sản phẩm)**

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục theo nhóm \ Hóa đơn theo Sản phẩm” sẽ mở Form12.
- Trên Form12: thiết kế như sau  
 → tự thực hiện

#### **Câu 14: (Form13 – Menu Quản lý danh mục theo nhóm \ Hóa đơn theo Nhân viên)**

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục theo nhóm \ Hóa đơn theo Nhân viên” sẽ mở Form13.
- Trên Form13: thiết kế như sau  
 → tự thực hiện

#### **Câu 15: (Form14 – Menu Quản lý danh mục theo nhóm \ Chi tiết hóa đơn theo Hóa đơn)**

\* Yêu cầu:

- Trên Form1: khi chọn menu “Quản lý danh mục theo nhóm \ Chi tiết hóa đơn theo Hóa đơn” sẽ mở Form14.
- Trên Form14: thiết kế như sau  
 → tự thực hiện

#### **Câu 16: (Form15 – Menu Quản lý danh mục theo nhóm \ Đa cấp)**

\* Yêu cầu:

- Trên Form1: *bỏ sang* menu “Quản lý danh mục theo nhóm \ Quản lý Đa cấp”, và khi chọn sẽ mở Form15.
- Trên Form15: thiết kế như sau

Quản lý Đa cấp

Chọn Thành Phố: TP HCM Ok

Danh sách khách hàng: 13

Mã KH	Tên Cty	Địa Chỉ	Thành Phố	Điện
DHP	Đại Hồng Phát	406 Ngô Gia Tự P4 Q10	TP HCM	08854
FAHASA	Phát Hành Sách Sài Gòn	12 Thuận Kiều	TP HCM	08842
HUNSAN	Hùng Sáng	275 Lý Thường Kiệt P15 Q11	TP HCM	08821
INEXIM	XNK Hàng Công Nghiệp	27A Minh Phụng P5 Q6	TP HCM	08896
LIPEXIM	Đầu Tư & XNK	8A Đoàn Kết Q1	TP HCM	08824
MINEXIM	Vật Tư Thiết Bị Công Nghiệp	226 Phú Hòa P8 QTB	TP HCM	08845

Danh sách hóa đơn: 4

Mã HD	Tên Cty	Mã NV	Ngày Lập HD	Ngày Nhận Hàng
10360	Phát Hành Sách Sài Gòn	Nguyễn Kim Ngọc	10/16/1992	11/13/1992
10436	Phát Hành Sách Sài Gòn	Trần Tuyết Oanh	12/30/1992	1/27/1993
10449	Phát Hành Sách Sài Gòn	Trần Tuyết Oanh	1/12/1993	2/9/1993
10566	Phát Hành Sách Sài Gòn	Vương Ngọc Lan	5/6/1993	6/3/1993

Chi tiết hóa đơn: 3

Mã HD	Tên SP	Số Lượng
10360	Rouou	10
10360	Nem	28
10360	Táo	30

Trở Về

-- Hết --