

MỤC LỤC

BÀI TẬP THỰC HÀNH 1: LẬP TRÌNH CĂN BẢN VỚI C#.....	2
BÀI TẬP THỰC HÀNH 2: LẬP TRÌNH CĂN BẢN VỚI C#.....	13
BÀI TẬP THỰC HÀNH 3: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG TRONG C#...18	
BÀI TẬP THỰC HÀNH 4: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG VỚI C# (tiếp) 22	
BÀI TẬP THỰC HÀNH 5: LÀM VIỆC VỚI WINDOWS FORM.....	27
BÀI TẬP THỰC HÀNH 6: LÀM VIỆC VỚI WINDOWS FORM (tiếp).....	39
BÀI TẬP THỰC HÀNH 7: TRUY CẬP DỮ LIỆU VỚI ADO.NET.....	47
BÀI TẬP THỰC HÀNH 8: TRUY CẬP DỮ LIỆU VỚI ADO.NET (tiếp)	55
BÀI TẬP THỰC HÀNH 9: MỘT SỐ KỸ THUẬT LẬP TRÌNH NÂNG CAO TRONG .NET (LÀM QUEN VỚI LINQ).....	72
BÀI TẬP THỰC HÀNH 10: LẬP TRÌNH NÂNG CAO TRONG C# (LÀM VIỆC VỚI LINQ)	80

BÀI TẬP THỰC HÀNH 1: LẬP TRÌNH CĂN BẢN VỚI C#

1. Mục tiêu kiến thức

- Giúp sinh viên làm quen với ngôn ngữ C#: qua việc viết các ứng dụng console đơn giản, các cấu trúc điều khiển, các câu lệnh cơ bản tuần tự, rẽ nhánh, vòng lặp trong C#
- Làm quen với môi trường phát triển tích hợp Visual studio: các công cụ hỗ trợ soạn thảo mã nguồn, các công cụ biên dịch, debug...

2. Yêu cầu:

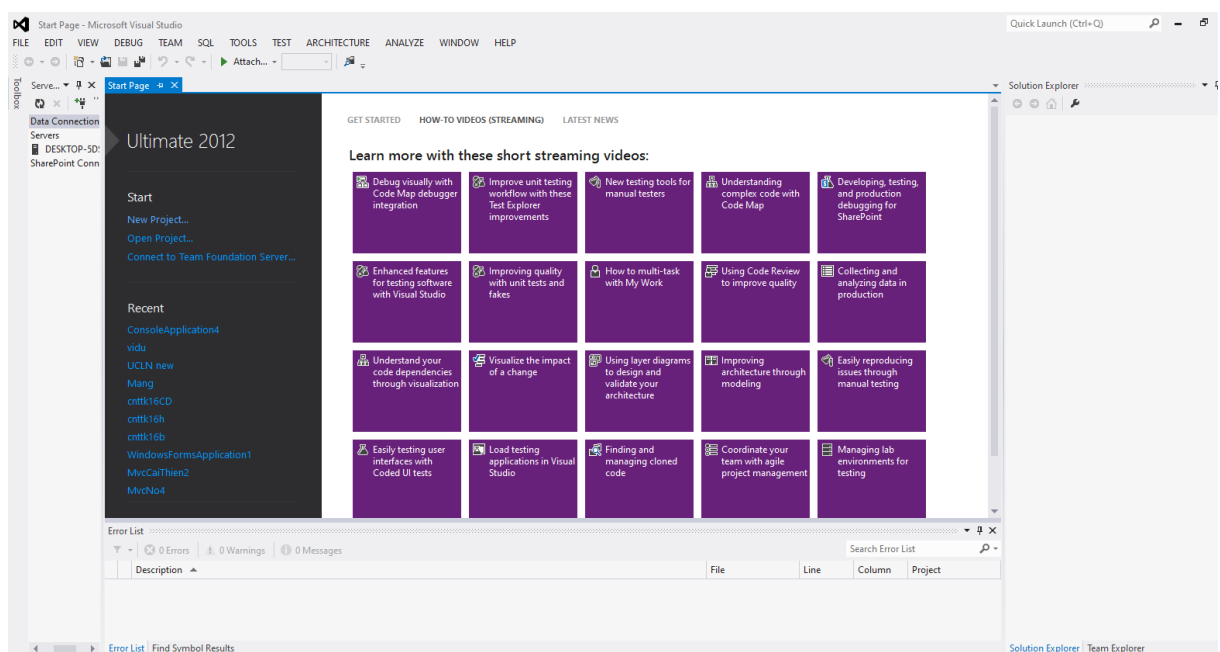
- + Yêu cầu về điều kiện thực hành: máy tính, phần mềm visual studio
- + Yêu cầu sinh viên: nắm được lý thuyết về ngôn ngữ C#, cách tạo ra một project, biên dịch và chạy chương trình. Các câu lệnh đơn giản như khai báo biến, hằng, vòng lặp, điều kiện....

3. Nội dung

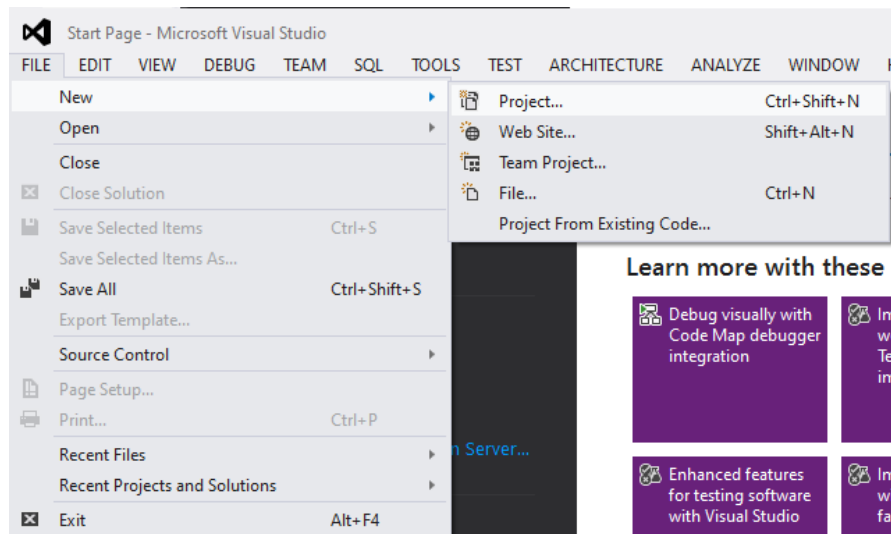
3.1. Bài thực hành mẫu

Bài 1: Tạo ứng dụng HelloWorld

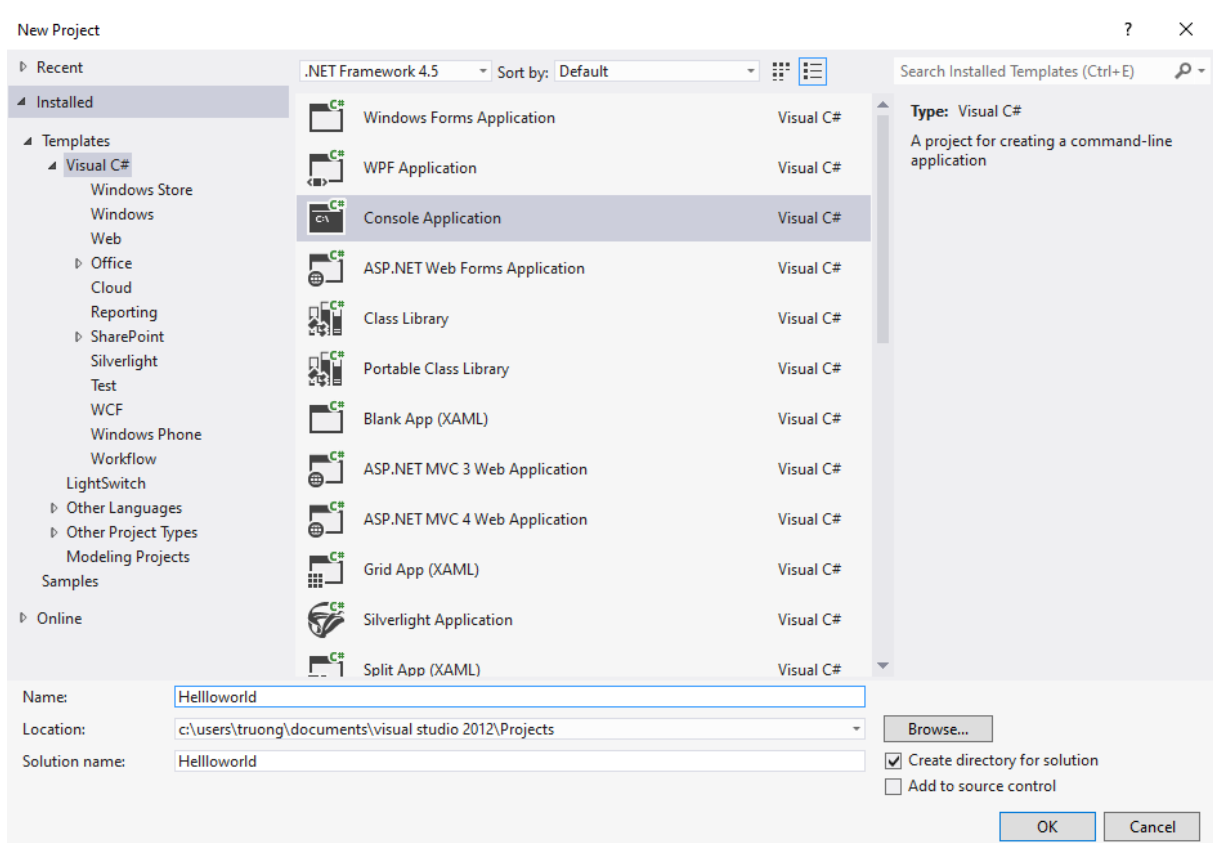
1. Để sử dụng VS. NET thực hiện như sau: **Start | Programs | Microsoft Visual Studio .NET | Microsoft Visual Studio .NET**. Start Page xuất hiện như hình dưới đây.

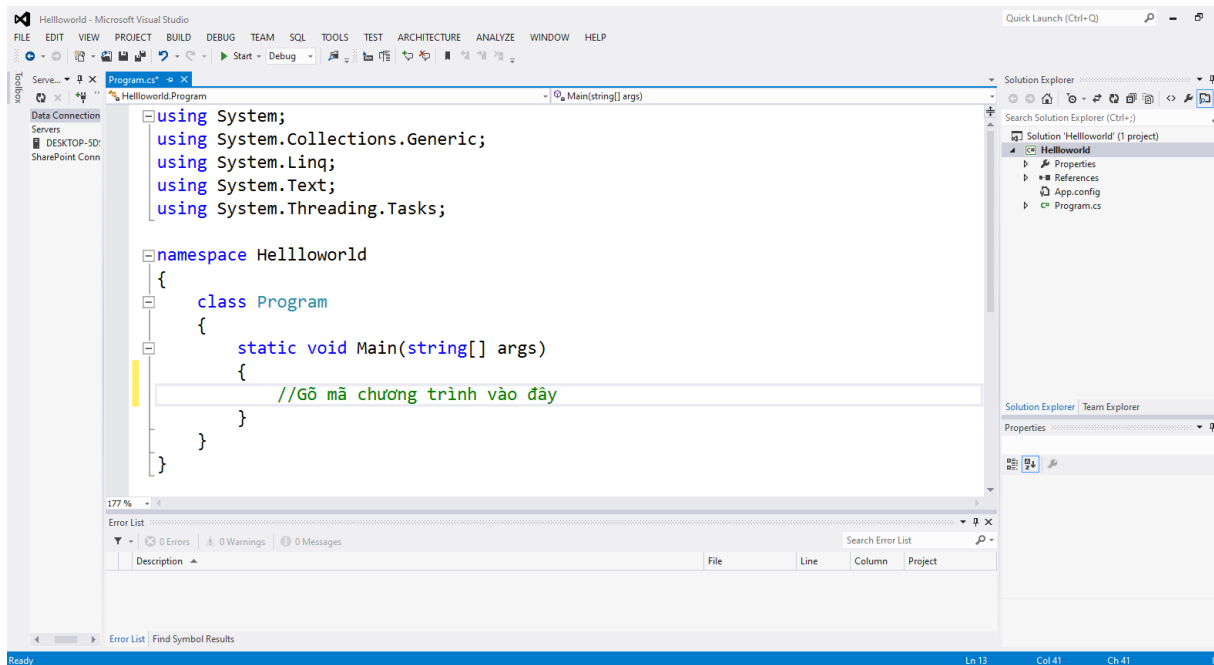


2. Từ menu **File | New | Project**. Cửa sổ **New project** xuất hiện như hình dưới đây.



Trong hộp thoại **New Project**, ở vùng **Recent Template**, click **Visual C#**. Ở vùng **Templates**, click **Console Application**, đặt tên của project là “HelloWorld” sau đó click OK để tạo một project mới.

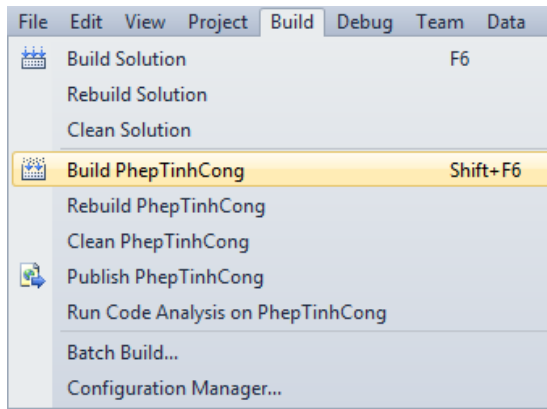




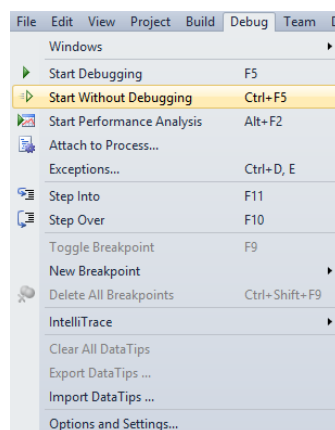
Mã nguồn chương trình:

```
using System;
namespace HelloWorld
{
    class Program
    {
        static void Main(string[] args)
        {
            System.Console.WriteLine("Hello Character World");
            System.Console.ReadLine();
        }
    }
}
```

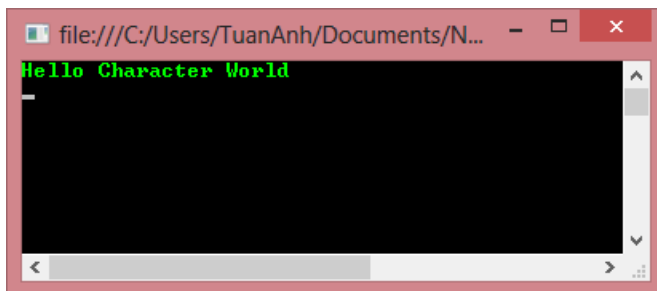
- Dịch và chạy chương trình
 - Build chương trình bằng cách Build -> Build Solution. Hoặc sử dụng phím tắt **Ctrl+Shift+B**.



- Chạy chương trình sử dụng Debug->Start Without Debugging. Hoặc có thể sử dụng phím tắt **Ctrl+F5**.



- Kết quả hiển thị ra màn hình như sau:



Bài 2: Viết chương trình nhập vào tên của mình và xuất ra màn hình “Xin chào + Tên”.

Chú ý:

- Khai báo biến chuỗi:
`string ten;`
- Nhập chuỗi từ bàn phím:
`ten = Console.ReadLine();`
- Xuất ra màn hình:

```
Console.WriteLine("Chuỗi" + biến);
```

Mã nguồn chương trình:

```
using System;
namespace HelloWorld1
{
    class Program
    {
        static void Main(string[] args)
        {
            string ten;
            ten = Console.ReadLine();
            Console.WriteLine("Xin chào " + ten);
        }
    }
}
```

Bài 3: Viết chương trình nhập 2 số nguyên, xuất tổng, hiệu, tích, thương.

Mã nguồn chương trình:

```
using System;
namespace Demo
{
    class Program
    {
        static void Main(string[] args)
        {
            int a, b;    //Toán hạng
            char ch;     //Phép toán
            Console.Write("Nhập A: ");
            a = Convert.ToInt16(Console.ReadLine());
            Console.Write("Nhập B: ");
            b = Convert.ToInt16(Console.ReadLine());
            Console.Write("Nhập phép toán: ");
            ch = Console.ReadKey().KeyChar;
```

```

        Console.WriteLine("\n" + TinhToan(a, b, ch));
        Console.ReadLine();

    }
    static string TinhToan(int a, int b, char ch)
    {
        switch (ch)
        {
            case '+': return "a + b = " + (a + b);
            case '-': return "a - b = " + (a - b);
            case '*': return "a * b = " + (a * b);
            case '/':
                if (b != 0) return "a / b = " + (a / b);
                else return "Nhap b khác khong";
            default: return "Nhap phep toan khac";
        }
    }
}

```

Bài 4: Viết chương trình nhập số nguyên N, kiểm tra và xuất kết quả N là số chẵn/lẻ.

Hướng dẫn:

- Nhập N từ bàn phím: khai báo là int N;
- Dùng câu điều kiện If kiểm tra N là số chẵn hay lẻ
- Hiện thị kết quả.

Code

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Tinh2So
{
    class Program

```

```

{
    static void Main(string[] args)
    {
        int N;          // khai báo biến N
        // Nhập N từ bàn phím
        Console.Write("Nhập số N:");
        N = Convert.ToInt16(Console.ReadLine());
        // Dùng câu lệnh kiểm tra có điều kiện If để kiểm tra N là số chẵn hay lẻ
        if (N % 2 == 0)
            Console.WriteLine("{0} là số chẵn", N);

        else
            Console.WriteLine("{0} là số lẻ", N);

        Console.ReadLine();
    }
}

```

Bài 5: Sử dụng C# viết chương trình Nhập vào ba cạnh của một tam giác và kiểm tra xem tam giác đó là tam giác đều, tam giác cân hay tam giác thường.

Hướng dẫn:

- Cần kiểm tra xem 3 số nhập vào có phải là 3 cạnh của tam giác hay không (tổng 2 cạnh phải lớn hơn cạnh còn lại)
- Là tam giác đều nếu 3 cạnh bằng nhau, tam giác cân trong trường hợp có 2 cạnh bất kì bằng nhau

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace Demo
{
    class Program
    {
        public static void Main()
        {

            int canha, canhb, canhc;
            Console.Write("\n");
            Console.Write("Kiểm tra tam giác đều, cân, thường trong C#:\n");

```



```

Console.Write("Nhập cạnh a: ");
canha = Convert.ToInt32(Console.ReadLine());

Console.Write("\nNhập cạnh b: ");
canhb = Convert.ToInt32(Console.ReadLine());

Console.Write("\nNhập cạnh c: ");
canhc = Convert.ToInt32(Console.ReadLine());
if (canha + canhb <= canhc || canha + canhc <= canhb || canhb + canhc <=
canha)
{
    Console.WriteLine("Không phải tam giác");
}
else
{
    if (canha == canhb && canhb == canhc)
    {
        Console.WriteLine("Đây là tam giác đều.\n");
    }
    else if (canha == canhb || canha == canhc || canhb == canhc)
    {
        Console.WriteLine("Đây là tam giác cân.\n");
    }
    else
    {
        Console.WriteLine("Đây là tam giác thường.\n");
    }
}
Console.ReadKey();
}
}
}

```

Bài 6: Viết chương trình nhập vào 2 số a và b thực hiện tìm ước số chung lớn nhất của 2 số đó.

Hướng dẫn:

- Nhập 2 số a và b khai báo int a,b
- Áp dụng thuật Euclid và sử dụng vòng lặp để triển khai

```

using System;

using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace UCLN_new
{

```

```

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

        int a, b;
        Console.WriteLine("a =");
        a = int.Parse(Console.ReadLine());
        Console.WriteLine("b =");
        b = int.Parse(Console.ReadLine());
        // lấy trị tuyệt đối
        a = Math.Abs(a);
        b = Math.Abs(b);

        while (a % b != 0)
        {
            int r = a % b;
            a = b;
            b = r;
        }
        Console.WriteLine("UCLN = " + b);

        Console.ReadKey();
    }
}

```

3.2. Các bài thực hành cơ bản

Bài 1: Viết chương trình giải phương trình bậc nhất có dạng $ax + b = 0$, với các hệ số a, b được nhập từ bàn phím.

Bài 2: Viết chương trình nhập chiều dài, chiều rộng hình chữ nhật, xuất chu vi, diện tích của hình chữ nhật đó.

Bài 3: Viết chương trình nhập bán kính hình tròn, xuất chu vi, diện tích của hình tròn đó.

Hướng dẫn:

- Đặt `const double pi = 3.14`
- Tính chu vi và diện tích hình tròn

Bài 4: Viết chương trình nhập vào một số kiểm tra xem số đó có phải số chính phương hay không ?

Bài 5: Sử dụng ngôn ngữ lập trình C# viết chương trình giải bài toán sau:

Tìm số gà và số chó, biết:
Vừa gà vừa chó
Bó lại cho tròn
Ba mươi sáu con
Một trăm chân chẵn
Hỏi có bao nhiêu gà, bao nhiêu chó

Bài 6 : Sử dụng ngôn ngữ lập trình C# viết chương trình xét xem một số n có phải là số nguyên tố không?

Hướng dẫn: Nếu n không chia hết mọi số i có giá trị từ 2 đến $n - 1$ thì n là số nguyên tố. Sử dụng biến ok có kiểu int và có giá trị ban đầu là 1. Cho biến i chạy từ 2 đến $n - 1$. Xét $n \% i$, nếu bằng 0 thì gán $ok = 0$. Ngược lại vẫn để nguyên ok .

Bài 7: Viết chương trình tính $n!$ với $n!$ được định nghĩa như sau:

- $n! = 1$ với $n = 0$

- $n! = 1.2.3...n$ (Tích của n số từ 1 đến n).

Yêu cầu: Sử dụng vòng lặp với số lần chưa biết trước:

Hướng dẫn: Có thể viết lại: $n! = n \cdot (n-1) \dots 3.2.1$. Lặp $gt = gt * n$; $n = n-1$ với điều kiện $n > 0$.

3.3. Các bài thực hành nâng cao

Bài 1: Viết chương trình nhập vào đơn giá 1 mặt hàng, và số lượng bán của mặt hàng. Tính tiền khách phải trả, với thông tin như sau:

- ✓ Thành tiền: đơn giá * số lượng
- ✓ Giảm giá: Nếu thành tiền > 100 , thì giảm 3% thành tiền, ngược lại không giảm
- ✓ Tổng tiền phải trả: thành tiền – giảm giá.

Bài 2: Viết chương trình tính tiền điện sử dụng trong tháng:

- ✓ Từ 1 – 100KW: 5\$
- ✓ Từ 101 – 150KW: 7\$
- ✓ Từ 151 – 200KW: 10\$
- ✓ Từ 201 – 300KW: 15\$
- ✓ Từ 300KW trở lên: 20\$

Bài 3: Sử dụng ngôn ngữ C# Viết chương trình tính tổng S, với n nguyên dương được nhập vào từ bàn phím.

$$S = 1 - \frac{1}{1^2 + 2^2} + \frac{1}{1^2 + 2^2 + 3^2} - \dots + \frac{(-1)^{n+1}}{1^2 + 2^2 + 3^2 \dots + n^2}$$

Bài 4: Sử dụng ngôn ngữ C# viết chương trình Nhập x, n để tính tổng chuỗi số sau (làm tròn 3 chữ số):

$$S(x, n) = x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^n}{n!}$$

Bài 5: Sử dụng ngôn ngữ C# viết chương trình tính tổng sau:

$$\frac{1}{1.2.3} + \frac{1}{2.3.4} + \frac{1}{3.4.5} + \dots + \frac{1}{n(n+1)(n+2)}$$

BÀI TẬP THỰC HÀNH 2: LẬP TRÌNH CĂN BẢN VỚI C#

1. Mục tiêu:

- Giúp sinh viên làm quen với ngôn ngữ C#: qua việc viết các ứng dụng console đơn giản, làm quen với cấu trúc dữ liệu mảng, khai báo sử dụng mảng một chiều, mảng 2 chiều...

- Làm quen với môi trường phát triển tích hợp VS .NET: các công cụ hỗ trợ soạn thảo mã nguồn, các công cụ biên dịch, debug...

2. Yêu cầu:

+ Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio

+ Yêu cầu sinh viên: nắm vững kiến thức về các ngôn ngữ C# như mảng, dãy, vòng lặp, điều kiện,...

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1: Viết chương trình C# cho phép nhập các phần tử mảng một chiều, sau đó tìm tổng các phần tử của mảng và hiển thị kết quả trên màn hình.

Mục đích: Giúp sinh viên làm quen với các khái niệm: khai báo mảng, khởi tạo mảng, và cách truy cập các phần tử của mảng trong C#.

Hướng dẫn :

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

namespace Demo
{
    class Program
    {
        public static void Main()
        {
            int[] a = new int[100];
            int i, n, sum = 0;

            Console.WriteLine("Nhập số phần tử cần lưu trữ vào trong mảng: ");
            n = Convert.ToInt32(Console.ReadLine());
```

```

    Console.WriteLine("Nhập {0} phần tử vào trong mảng: \n", n);
    for (i = 0; i < n; i++)
    {
        Console.WriteLine("Phần tử - {0}: ", i);
        a[i] = Convert.ToInt32(Console.ReadLine());
    }

    for (i = 0; i < n; i++)
    {
        sum += a[i];
    }

    Console.WriteLine("Tổng các phần tử trong mảng là: {0}\n\n", sum);

    Console.ReadKey();
}
}
}

```

Bài 2: Viết chương trình C# cho phép nhập một mảng một chiều, thực hiện chèn thêm phần tử mới vào trong mảng sau đó in mảng trên màn hình.

Mục tiêu:

Bài tập chèn phần tử vào mảng là bài tập rất phổ biến trong mọi ngôn ngữ lập trình và có nhiều ứng dụng. Bài tập C# này giúp làm quen với các khái niệm: khai báo mảng, khởi tạo mảng, và cách truy cập các phần tử của mảng trong C#.

Hướng dẫn:

- Để chèn một phần tử vào vị trí X ta thực hiện di chuyển các phần tử mảng ban đầu từ vị trí X sang bên phải một đơn vị
- Gán vị trí ô nhớ X bằng phần tử người dùng nhập vào

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```

namespace Demo
{
    class Program
    {
        public static void Main()
        {

```

```

int[] arr1 = new int[20];
int i, n, p, x;

Console.WriteLine("\nChen phan tu vao mang trong C#: \n");
Console.WriteLine("Nhap kích co mang: ");
n = Convert.ToInt32(Console.ReadLine());
/* nhap cac phan tu vao trong mang*/
Console.WriteLine("Nhap {0} phan tu vao trong mang:\n", n);
for (i = 0; i < n; i++)
{
    Console.WriteLine("Phan tu - {0}: ", i);
    arr1[i] = Convert.ToInt32(Console.ReadLine());
}

Console.WriteLine("Nhap gia tri phan tu moi can chen: ");
x = Convert.ToInt32(Console.ReadLine());
Console.WriteLine("Nhap vi tri can chen phan tu moi nay: ");
p = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("In mang ban dau:\n");
for (i = 0; i < n; i++)
    Console.WriteLine("{0} ", arr1[i]);
/* di chuyen vi tri cac phan tu ben phai cua mang */
for (i = n; i >= p; i--)
    arr1[i] = arr1[i - 1];
/* chen gia tri vao vi tri da cho */
arr1[p - 1] = x;

Console.WriteLine("\n\nSau khi chen phan tu, mang co dang:\n");
for (i = 0; i <= n; i++)
    Console.WriteLine("{0} ", arr1[i]);
Console.WriteLine("\n\n");

Console.ReadKey();
}
}
}

```

Bài 3: Viết chương trình C# cho phép nhập vào 2 ma trận và tính tổng hai ma trận và sau đó in ma trận kết quả trên màn hình.

Hướng dẫn:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace Demo

```

```

{
    class Program
    {
        public static void Main()
        {
            int i, j, n;
            int[,] arr1 = new int[50, 50];
            int[,] arr2 = new int[50, 50];
            int[,] ma_tran_tong = new int[50, 50];

            Console.WriteLine("Nhap kích co của hai ma tran vuong (nhỏ hơn 5): ");
            n = Convert.ToInt32(Console.ReadLine());

            /* Nhap các phần tử vào trong mảng đã chieu */
            Console.WriteLine("Nhap các phần tử vào trong ma tran đầu tiên:\n");
            for (i = 0; i < n; i++)
            {
                for (j = 0; j < n; j++)
                {
                    Console.WriteLine("Phần tử - [{0},{1}]: ", i, j);
                    arr1[i, j] = Convert.ToInt32(Console.ReadLine());
                }
            }

            Console.WriteLine("Nhap các phần tử vào trong ma tran thứ hai:\n");
            for (i = 0; i < n; i++)
            {
                for (j = 0; j < n; j++)
                {
                    Console.WriteLine("Phần tử - [{0},{1}]: ", i, j);
                    arr2[i, j] = Convert.ToInt32(Console.ReadLine());
                }
            }

            Console.WriteLine("\nIn ma tran thứ nhất:\n");
            for (i = 0; i < n; i++)
            {
                Console.WriteLine("\n");
                for (j = 0; j < n; j++)
                    Console.WriteLine("{0}\t", arr1[i, j]);
            }

            Console.WriteLine("\nIn ma tran thứ hai:\n");
            for (i = 0; i < n; i++)
            {
                Console.WriteLine("\n");
                for (j = 0; j < n; j++)
                    Console.WriteLine("{0}\t", arr2[i, j]);
            }

            /* cộng hai ma tran */
            for (i = 0; i < n; i++)
                for (j = 0; j < n; j++)
                    ma_tran_tong[i, j] = arr1[i, j] + arr2[i, j];
            Console.WriteLine("\nMa tran tong của hai ma tran trên là: \n");
            for (i = 0; i < n; i++)
            {
                Console.WriteLine("\n");
                for (j = 0; j < n; j++)
                    Console.WriteLine("{0}\t", ma_tran_tong[i, j]);
            }
            Console.WriteLine();

            Console.ReadKey();
        }
    }
}

```


}
}

3.2. Các bài thực hành tương tự

Bài 1 : *Viết chương trình nhập vào một mảng một chiều và thực hiện:*

- 1) Liệt kê các phần tử dương
- 2) Liệt kê các phần tử lẻ ở vị trí chẵn.
- 3) Liệt kê các số nguyên tố, số chính phương, số hoàn hảo trong mảng.
- 4) Tìm phần tử lớn nhất, nhỏ nhất trong mảng một chiều
- 5) Xóa một phần tử khỏi mảng
- 6) Xóa tất cả phần tử = x.
- 7) Sắp xếp mảng theo chiều tăng , giảm dần.

Bài 2: *Viết chương trình nhập vào một ma trận và thực hiện:*

- 1) Tính tổng các phần tử trên cùng một dòng
- 2) Tính tổng các phần tử trên cùng một cột
- 3) Tính tổng các phần tử trên đường chéo chính
- 4) Tìm ma trận chuyển vị của ma trận vừa nhập
- 5) Sắp xếp các phần tử của ma trận theo chiều tăng, giảm dần

3.3. Các bài thực hành nâng cao

Bài 1: *Viết chương trình nhập vào một mảng một chiều và thực hiện:*

- 1) Đếm số phần tử xuất hiện nhiều nhất trong mảng
- 2) Tìm số lẻ nhỏ nhất lớn hơn mọi số chẵn có trong mảng
- 3) Xóa tất cả những phần tử trùng nhau trong dãy chỉ giữ lại một phần tử trong đó.

Ví dụ: 1 6 **2** 3 **2** 4 **2** 6 5 => 1 6 2 3 4 5

Bài 2: *Viết chương trình nhập vào 2 mảng một chiều và thực hiện:*

- ✓ Tìm dãy con chung dài nhất của 2 mảng đã nhập

Bài 3: *Viết chương trình nhập vào 2 ma trận và thực hiện:*

- 1) Tính tích 2 ma trận đã nhập
- 2) In ra ma trận tam giác trên của ma trận kết quả
- 3) Tính định thức của ma trận kết quả

BÀI TẬP THỰC HÀNH 3: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG TRONG C#

1. Mục tiêu:

- Giúp sinh viên làm quen và nâng cao kỹ năng lập trình hướng đối tượng trong C# như: Khai báo lớp, khởi tạo và sử dụng đối tượng, các đặc trưng của lập trình hướng đối tượng trong C#.

2. Yêu cầu:

- + Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- + Yêu cầu sinh viên: nắm vững kiến thức về các ngôn ngữ C# như lớp, đối tượng, constructor, hàm static, thuộc tính truy cập,....

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1. Sử dụng C# viết chương trình xây dựng lớp phân số cho phép nhập , xuất phân số, cộng 2 phân số và trả về là một phân số tối giản.

Hướng dẫn:

- Ta đi xây dựng lớp phân số với 2 thuộc tính là tử số và mẫu số
- Các phương thức bao gồm phương thức nhập, phương thức in phân số, cộng phân số
- Để lấy được phân số tối giản sau khi cộng phân số ta chia tử số mẫu số của phân số kết quả cho UCLN của chúng.

Lớp Phân số:

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

namespace Phansochuan
{
    class Phanso
    {
        public int tuso { set; get; }
        public int mauso { set; get; }

        public void NhapPhanso()
        {
            Console.WriteLine("Tu so = ");
```

```

        this.tuso = int.Parse(Console.ReadLine());
        Console.WriteLine("mau so = ");
        do
            this.mauso = int.Parse(Console.ReadLine());
        while (this.mauso == 0);
    }
    public void InPhanso(int tuso, int mauso)
    {
        Console.Write("Phan so = " + tuso + "/" + mauso);
    }
    public Phanso congphanso(Phanso ph1, Phanso ph2)
    {
        Phanso ph3 = new Phanso();
        ph3.tuso = ph1.tuso * ph2.mauso + ph2.tuso * ph1.mauso;
        ph3.mauso = ph1.mauso * ph2.mauso;

        int uc = ucln(ph3.tuso, ph3.mauso);
        ph3.tuso = ph3.tuso / uc;
        ph3.mauso = ph3.mauso / uc;

        return ph3;
    }
    public int ucln(int a, int b)
    {
        int r = 1;
        while (r != 0)
        {
            r = a % b;
            a = b;
            b = r;
        }
        return a;
    }
}

```

Hàm Main

```

using System.Text;
using Phansochuan;
namespace Phansochuan
{
    class Program
    {
        static void Main(string[] args)

```

```

{
    Console.WriteLine("Nhập phân số 1");
    Phanso p1 = new Phanso();
    p1.NhapPhanso();
    Console.WriteLine("Nhập phân số 2");
    Phanso p2 = new Phanso();
    p2.NhapPhanso();
    Console.WriteLine("tong 2 phân số là");
    Phanso p3 = new Phanso();
    p3 = p3.congphanso(p1, p2);
    p3.InPhanso(p3.tuso, p3.mauso);
    Console.ReadKey();
}
}
}

```

3.2. Các bài tập tương tự

Bài 1. Viết một chương trình để cài đặt một hệ thống quản lý kho.

Hãy lưu trữ mã số, tên hàng, giá và số lượng đang có của mỗi món hàng trong một lớp. Nhập chi tiết của N (N nhập từ bàn phím) món hàng hiển thị tên từng món hàng và tổng giá trị của nó.

Bài 2. Viết một chương trình để lưu trữ các sinh viên gồm: mã sinh viên, họ và tên và điểm trung bình của N (N nhập từ bàn phím) sinh viên.

Hãy sắp xếp danh sách sinh viên này theo thứ tự điểm trung bình giảm dần. Hiển thị 3 sinh viên có điểm trung bình cao nhất.

3.3. Các bài tập nâng cao

Bài 1. Tạo một lớp CD gồm: CDName, CDType và CDPrice và các phương thức cần thiết khác.

Viết một ứng dụng có dạng menu như sau:

Menu

1. Add CD
2. Search CD
3. Display catalog
4. Exit

Your choice: _

Thêm CD: Cho phép thêm một đĩa CD vào danh mục các đĩa CD hiện có. Hãy cung cấp khả năng quản lý khoảng 1000 đĩa CD. Cần có sự kiểm soát nếu số lượng đĩa CD được nhập vượt quá 1000.

Search CD: Nhập vào tên của đĩa CD, nếu không tìm thấy thì báo lỗi, nếu tìm thấy thì in các thông tin liên quan đến đĩa CD đó.

CD No.	CD Name	CD Type	CD Price
1	Ngay khong em	Ca nhac	70K(VND)
2	Thoi xa vang	Phim	320K(VND)
3	De che	Tro choi	6K(VND)

Display catalog: Hiển thị tất cả các đĩa CD hiện có trong danh mục, danh mục đĩa CD được hiển thị ở dạng bảng, có chứa cột tiêu đề.

Bài 2:

1. Xây dựng các lớp đối tượng hình học như: điểm, đoạn thẳng, đường tròn, hình chữ nhật, hình vuông, tam giác, hình bình hành, hình thoi. Mỗi lớp có các thuộc tính riêng để xác định được hình vẽ biểu diễn của nó như đoạn thẳng thì có điểm đầu, điểm cuối....

2. Mỗi lớp thực thi một phương thức Draw() ghi đè (overriding) phương thức Draw() của lớp cơ sở gốc của các hình mà nó dẫn xuất. Hãy xây dựng lớp cơ sở của các lớp trên và thực thi đa hình với phương thức Draw(), sau đó tạo lớp Tester cùng với hàm Main() để thử nghiệm.

BÀI TẬP THỰC HÀNH 4: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG VỚI C# (tiếp)

1. Mục tiêu:

- Giúp sinh viên nâng cao kỹ năng xây dựng lớp đối tượng trong C#
- Xây dựng giao diện, kế thừa và thực thi giao diện.
- Nạp chồng phương thức
- Thuộc tính và thủ tục thuộc tính
- Thành phần tĩnh và cách sử dụng

2. Yêu cầu:

- + Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- + Yêu cầu sinh viên: nắm vững kiến thức về các ngôn ngữ C# như lớp, đối tượng, constructor, hàm static, thuộc tính truy cập,....

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1: Cho thiết kế lớp Employee (nhân viên) như sau:

Các thành phần dữ liệu:

- id: Định danh, kiểu int. Định danh này được sinh tự động và tăng dần bắt đầu từ 1.
- name: Họ tên nhân viên, kiểu String.
- yearOfBirth: Năm sinh nhân viên, kiểu int.
- salaryLevel: Bậc lương, kiểu double.
- basicSalary: Lương cơ bản, kiểu double. (Chú ý lương cơ bản là thuộc tính được sử dụng chung cho mọi đối tượng của lớp Employee).

Các phương thức:

- GetId(): trả lại định danh của nhân viên.
- GetName(): trả lại tên của nhân viên.
- GetYearOfBirth(): trả lại năm sinh của nhân viên.
- GetIncome(): trả lại thu nhập của nhân viên. Thu nhập được tính bằng bậc lương nhân lương cơ bản (salaryLevel * basicSalary).
- Input(): nhập thông tin nhân viên.

- Display(): hiển thị thông tin về nhân viên. Bao gồm các thông tin: định danh, tên, năm sinh, lương cơ bản, thu nhập.
- SetSalaryLevel(): thiết lập bậc lương cho nhân viên.
- SetBasicSalary(): thiết lập lương cơ bản.

Hãy viết chương trình cài đặt lớp Employee và lớp sử dụng Employee.

Hướng dẫn:

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

namespace Employee_Class
{
    class Employee
    {
        //Khai bao cac thuoc tinh/thanh phan dữ liệu
        private int id;
        private string name;
        private int yearOfBirth;
        private double salaryLevel;
        double basicSalary;

        #region Tao lap doi tuong

        public Employee() { }
        #endregion

        // Phuong thuc GetId
        int GetID()
        {
            return id;
        }
        // phuong thuc GetName
        public string GetName
        {
            get { return name; }
        }

        // phuong thuc Get Year Of Birth
        public int GetYearOfBirth
        {
            get { return this.yearOfBirth; }
        }
    }
}
```

```

// phương thức getIncome
public double GetIncome
{
    get { return salaryLevel * basicSalary; }
}
//Xây dựng phương thức nhập
public void Input()
{
    Console.WriteLine("Enter id: ");
    this.id = Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("Enter name: ");
    this.name = Console.ReadLine();
    Console.WriteLine("Enter year of birth: ");
    this.yearOfBirth = Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("Enter salary level: ");
    this.salaryLevel = Convert.ToDouble(Console.ReadLine());
    Console.WriteLine("Enter basic salary: ");
    this.basicSalary = Convert.ToDouble(Console.ReadLine());
}

//Xây dựng phương thức hiển thị
public void Display()
{
    Console.WriteLine("Employee Info");
    Console.WriteLine("    ID:        " + this.GetID());
    Console.WriteLine("    Name:      " + this.GetName());
    Console.WriteLine("    Year Of Birt: " + this.GetYearOfBirth());
    Console.WriteLine("    Basic salary: " + this.basicSalary);
    Console.WriteLine("    Income:    " + this.GetIncome);
}

// Xây dựng phương thức SetSalaryLevel
public double SetSalaryLevel
{
    set { salaryLevel = value; }
}
public double SetBasicSalary
{
    set { basicSalary = value; }
}
}

```

3.2. Các bài tập tương tự

Bài 1: Xây dựng lớp Stack để mô phỏng một stack bao gồm:

- Phương thức khởi tạo không tham số với việc khởi tạo stack ngầm định có 20 phần tử và stack rỗng
- Phương thức khởi tạo một tham số với việc khởi tạo số phần tử của stack được truyền vào thông qua đối số của phương thức và stack rỗng
- Phương thức IsEmpty kiểm tra xem stack có rỗng không
- Phương thức IsFull kiểm tra xem stack có đầy không
- Phương thức Push và Pop để thêm vào, lấy ra một phần tử

Hãy viết chương trình cài đặt lớp Stack và lớp sử dụng Stack.

Bài 2: Viết chương trình quản lý sinh viên của một trường. Biết rằng mỗi sinh viên gồm có các thuộc tính sau: mã sinh viên, Họ tên, ngày tháng năm sinh, Điểm toán, điểm văn, điểm anh, chuyên ngành đào tạo

- 1) Chương trình cho phép nhập danh sách sinh viên, sau đó in danh sách sinh viên cùng với điểm trung bình của họ ra màn hình điểm trung bình được tính là $(\text{điểm toán} + \text{điểm văn} + \text{điểm anh}) / 3$
- 2) In ra danh sách những sinh viên có điểm trung bình cao trên 5.0 ra màn hình. Thông tin hiển thị có dạng Họ tên, Chuyên ngành đào tạo, Điểm trung bình.

Bài 3: Xây dựng lớp tam giác với các thuộc tính là 3 cạnh của tam giác xây dựng các phương thức:

- 1) Nhập tam giác, In tam giác
- 2) Xây dựng các phương thức tính chu vi, diện tích tam giác đó
- 3) Xây dựng phương thức kiểm tra xem tam giác đó là tam giác gì (tam giác thường, cân, đều, vuông)

3.3. Các bài tập nâng cao

Bài 1: Xây dựng chương trình quản lý lương cho công ty ABC. Thông tin để tính lương cho mỗi Nhân viên trong công ty bao gồm: Họ tên, quê quán, hệ số lương, lương cơ bản. Hãy nhập vào một danh sách các Nhân viên của công ty sau đó thực hiện các yêu cầu sau:

- Tính và hiển thị lương của các Nhân viên có trong danh sách
- Liệt kê những nhân viên có hệ số lương cao nhất
- Sắp xếp danh sách theo thứ tự tăng dần của trường hệ số lương

Hướng dẫn:

- Xây dựng lớp có tên NhanVien với các thành phần

- Dữ liệu: Họ tên, quê quán, hệ số lương, lương cơ bản(là thành phần dữ liệu tĩnh)
- Phương thức: Các thuộc tính để truy xuất tới các thành phần dữ liệu họ tên, hệ số lương, phương thức nhập, hiển thị, tính lương,...

- Xây dựng lớp QuanLy nhân viên bao gồm

- Dữ liệu: ds là một mảng các nhân viên

Phương thức: Phương thức nhập, hiển thị, sắp xếp,...

Bài 2: Xây dựng lớp có tên là TienDien với các thông tin bao gồm:

+ Dữ liệu:

- Họ tên chủ hộ
- Địa chỉ
- Số công tơ tháng trước
- Số công tơ tháng này

+ Phương thức

- Phương thức thiết lập không tham số và 4 tham số
- Phương thức nhập dữ liệu
- Phương thức hiển thị dữ liệu
- Thuộc tính tính số công tơ điện đã dùng (=Số công tơ tháng này- Số công tơ tháng trước)
- Phương thức tính tiền điện được tính theo công thức: Số điện đã dùng*1240
Sau đó xây dựng lớp TienDienMoi bằng việc kế thừa lớp TienDien để tính tiền điện theo một quy định mới. Việc tính tiền điện lúc này căn cứ vào định mức quy định. Nếu trong định mức là 1240, ngoài định mức là 1600

Hướng dẫn:

- Xây dựng lớp TienDien theo như đã mô tả

BÀI TẬP THỰC HÀNH 5: LÀM VIỆC VỚI WINDOWS FORM

1. Mục tiêu:

- + Tạo ứng dụng trên Windows Form.
- + Sử dụng các thuộc tính, phương thức và sự kiện của các control: Label, Textbox, Button, ErrorProvider, RadioButton, CheckBox, GroupBox, Panel, PictureBox, ListBox, CheckedListBox, ComboBox.

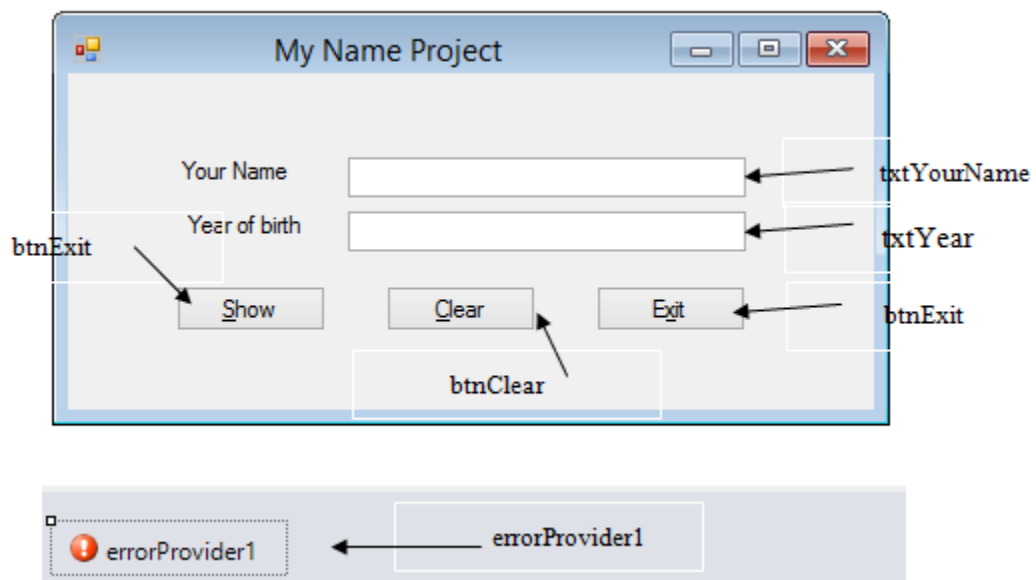
2. Yêu cầu:

- + Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- + Yêu cầu sinh viên: nắm vững kiến thức về lập trình trên Window Form như tạo Form, thêm các điều khiển, thay đổi các thuộc tính, viết các sự kiện.

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1: Thiết kế Form sau:



Yêu cầu:

- Chương trình cho phép nhập tên, năm sinh vào Textbox YourName và Year of birth tương ứng. Nếu YourName không nhập đủ liệu, Year of birth không phải là số thì phải thông báo lỗi (dùng ErrorProvider). Người dùng nhấn nút Show sẽ hiển thị thông tin nhập vào MessageBox bao gồm: tên, tuổi (năm hiện tại – năm sinh).

- Người dùng nhấn nút Clear sẽ xóa hết thông tin đã nhập trên các Textbox, đồng thời đặt con trỏ văn bản vào Textbox YourName.

- Nút Exit xác nhận người dùng có thực sự muốn thoát khỏi chương trình không? (Yes: thoát, No: không).

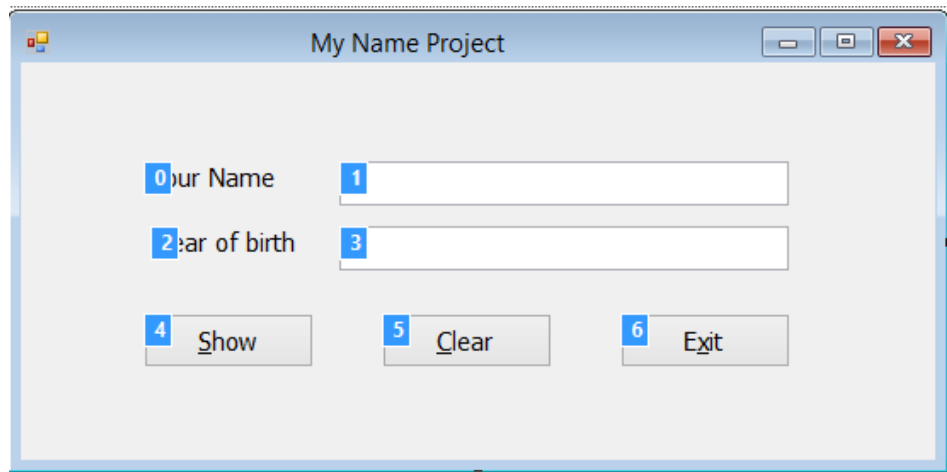
Hướng dẫn:

Danh sách các thuộc tính của các object:

Object	Properties	Events
frmMain	Name: frmMain Text: My name Project FontName: Tahoma FontSize: 11 AcceptButton: btnShow (nhận sự kiện click chuột khi nhấn Enter) CancelButton: btnExit (nhận sự kiện click chuột khi nhấn Esc)	FormClosing
txtYourName	Name: txtYourName BorderStyle: FixSingle	Leave (mất tiêu điểm)
TxtYear	Name: txtYear BorderStyle: FixSingle	TextChanged
btnShow	Name: btnShow Text: &Show	Click
btnClear	Name: btnClear Text: &Clear	Click
BtnExit	Name: btnExit Text: E&xit	Click
errorProvider	Name: errorProvider1	

Thứ tự nhận tiêu điểm trên Form: chọn menu View → Tab Order

Lần lượt thực hiện click chọn từng phần tử trên Form theo thứ tự nhận tiêu điểm:



Các sự kiện:

```
private void btnClear_Click(object sender, EventArgs e)
```

```
{
    txtYourName.Clear();
    txtYear.Clear();
    txtYourName.Focus();
}
```

```
private void btnShow_Click(object sender, EventArgs e)
```

```
{
    int age = DateTime.Now.Year - Convert.ToInt32(txtYear.Text);
    string s = "My name is: " + txtYourName.Text + "\n" + age.ToString();
    MessageBox.Show(s);
}
```

```
private void txtYourName_Leave(object sender, EventArgs e)
```

```
{
    Control ctr = (Control)sender;
    if (ctr.Text.Trim().Length == 0)
        this.errorProvider1.SetError(txtYourName, "You must enter Your name");
    else
        this.errorProvider1.Clear();
}
```

```

private void txtYear_TextChanged(object sender, EventArgs e)
{
    Control ctr = (Control)sender;
    if (ctr.Text.Trim().Length > 0 && !char.IsDigit(ctr.Text, ctr.Text.Length - 1))
        this.errorProvider1.SetError(txtYear, "This is not invalid number");
    else
        this.errorProvider1.Clear();
}

```

```

private void frmMain_FormClosing(object sender, FormClosingEventArgs e)
{
    DialogResult r;
    r = MessageBox.Show("Do you want to close?", "Exit",
        MessageBoxButtons.YesNo,
        MessageBoxIcon.Question,
        MessageBoxDefaultButton.Button1);
    if (r == DialogResult.No)
        e.Cancel = true;
}

```

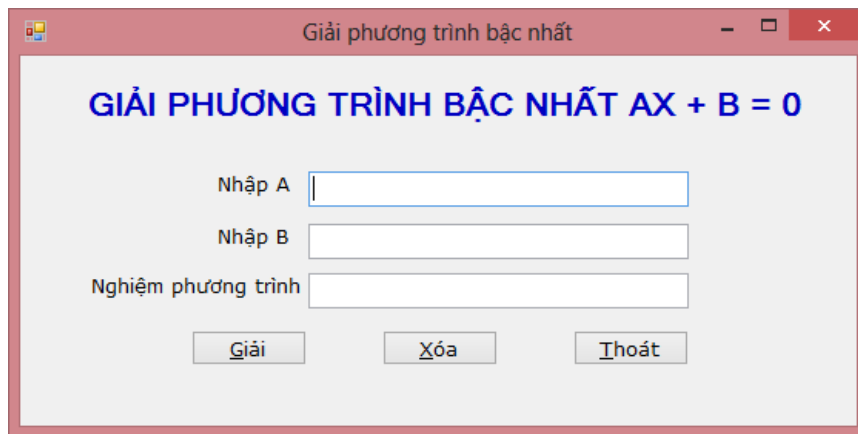
```

private void btnExit_Click(object sender, EventArgs e)
{
    this.Close();
}

```

3.2. Các bài tập tương tự

Bài 1: Thiết kế chương trình hiện thực bài toán giải phương trình bậc 1:



Yêu cầu:

- Khi form hiện lên thì nút Giải và nút Xóa bị mờ (Enabled=false). Nếu dữ liệu nhập không hợp lệ thì thông báo lỗi (dùng errorProvider). Sau khi nhập dữ liệu hợp lệ và đầy đủ thì nút Giải có tác dụng (Enabled=true).

- Khi nhấn nút Tính: tính nghiệm phương trình (xét tất cả các trường hợp xảy ra: PT có 1 nghiệm, vô nghiệm, vô số nghiệm) và hiện kết quả vào Nghiệm PT. Khi đó nút Xóa có tác dụng, nút Tính bị mờ.

- Khi nhấn nút Xóa: xóa các Textbox và Label, đặt con trỏ vào Textbox A, nút Xóa bị mờ

- Khi nhấn nút Thoát: xác nhận người dùng có chắc chắn thoát khỏi ứng dụng hay không?

Bài 2: Thiết kế giao diện như hình. Khi nhấn chọn vào phép tính nào thì sẽ hiện kết quả của phép tính đó vào ô Kết quả.

Trước khi tính cần kiểm tra dữ liệu nhập phải là số.

Bài 3: Thiết kế giao diện như hình sau:

Yêu cầu:

- Khi chương trình hiện lên:
 - Radiobutton Red được chọn mặc định (đổi màu chữ ô lblLapTrinh và ô txtNhapTen)
 - Con trỏ văn bản xuất hiện ngay tại ô txtNhapTen.
- Khi gõ vào ô txtNhapTen thì Label lblLapTrinh chạy song song cùng nội dung.
- Nhấn nút "Thoát" hoặc Esc thì thoát chương trình.

- Nhấn Radiobutton Red, Green, Blue, Black thì đổi màu chữ tương ứng trong ô lblLapTrinh và ô txtNhapten.

- Nhấn các checkbox chữ đậm, nghiêng, gạch chân thì đổi style chữ trong ô lblLapTrinh và ô txtNhapten tương ứng.

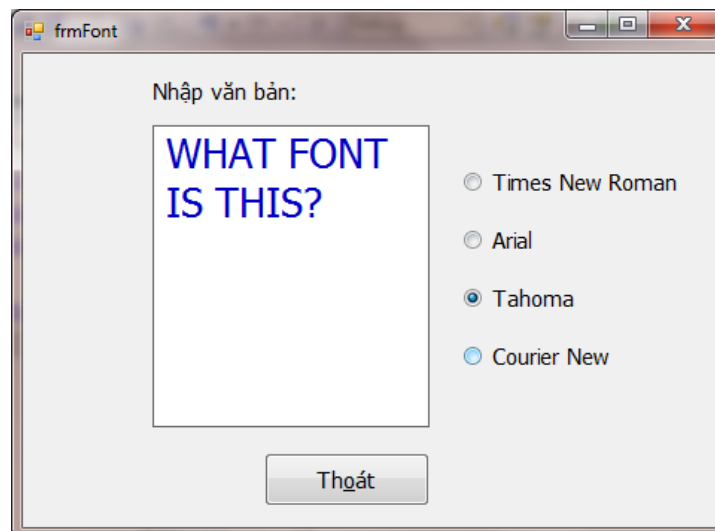
Hướng dẫn: đổi style chữ trong ô lblLapTrinh

Viết trong sự kiện CheckedChanged của từng checkbox:

```
private void chkdam_CheckedChanged(object sender, EventArgs e)
{
    lblLapTrinh.Font = new Font(lblLapTrinh.Font.Name,
                                lblLapTrinh.Font.Size,
                                lblLapTrinh.Font.Style ^ FontStyle.Bold);
}
```

3.3. Các bài tập nâng cao

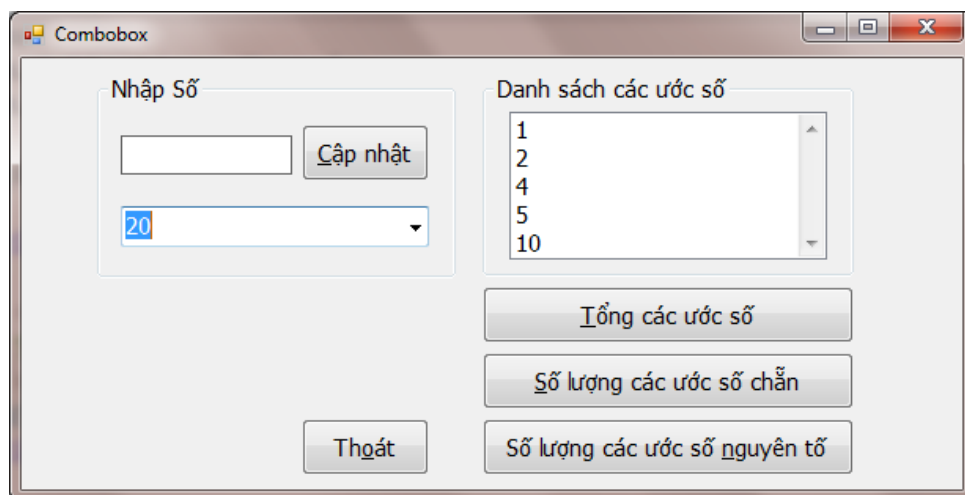
Bài 1: *Viết chương trình định dạng Textbox theo các font chữ tương ứng với từng Radiobutton*



Bài 2: Viết chương trình hiển thị vào PictureBox lá cờ tương ứng với nước được chọn trên Radio Button tương ứng.



Bài 3. Thiết kế giao diện như sau:



Yêu cầu:

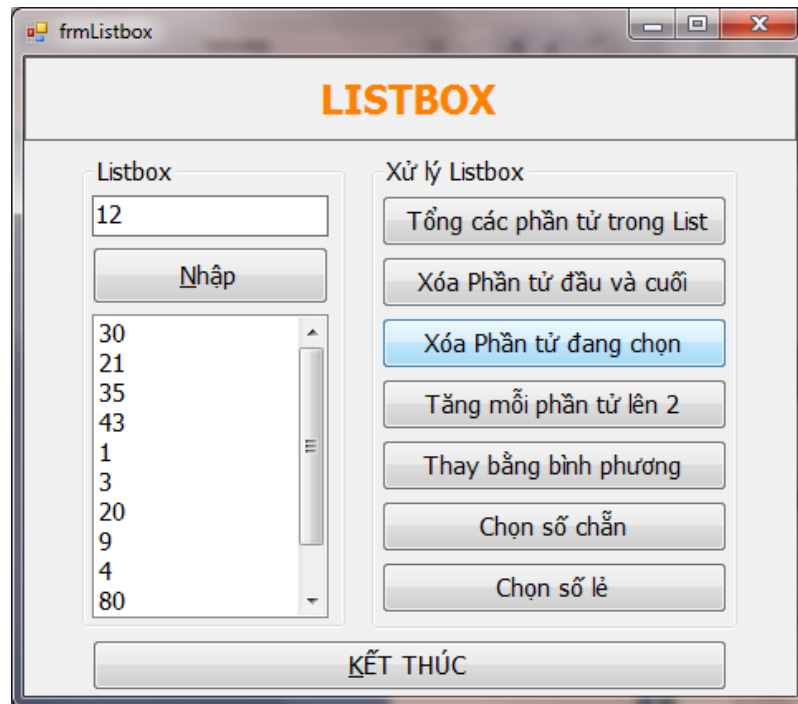
Khi Form vừa hiện lên, các Textbox, Combobox, Listbox chưa có dữ liệu, con trỏ đặt tại Textbox (thiết lập Tab Order hợp lý).

Nhấn nút “Cập nhật” hoặc Enter: thêm số vừa nhập ở Textbox vào Combobox (nhớ kiểm tra dữ liệu nhập), đồng thời xóa nội dung Textbox và đặt con trỏ lại Textbox.

Khi chọn 1 số trên Combobox thì danh sách các ước số của số này sẽ hiển thị vào Listbox bên phải tương ứng.

Khi nhấn các nút: “Tổng các ước số”, “Số lượng các ước số chẵn”, “Số lượng các ước số nguyên tố” thì sẽ hiển thị thông tin tương ứng vào MessageBox dựa vào các ước số trên Listbox.

Bài 4. Thiết kế giao diện như sau:



Yêu cầu:

Khi Form vừa hiện lên, các Textbox, Listbox để trống, con trỏ đặt tại Textbox (thiết lập Tab Order hợp lý).

Khi người sử dụng nhập một số vào Textbox rồi Enter hoặc nhấn nút "Nhập" thì số đó được thêm vào Listbox, đồng thời nội dung trong Textbox bị xóa và con trỏ được chuyển về Textbox.

Người dùng nhấn vào nút nào thì thực hiện chức năng tương ứng của nút đó. Hiện kết quả ra MessageBox (nếu có).

Thiết lập thuộc tính Anchor hợp lý cho các control.

Thiết lập MinimumSize cho form.



Bài 5. Viết chương trình nhập danh sách sinh viên theo yêu cầu sau: (xem hình bên dưới).



Quy định Form hiển thị giữa màn hình. Không cho người sử dụng thay đổi kích thước Form.

Quy định việc di chuyển tab hợp lý.

Các Listbox được phép chọn nhiều mục (kết hợp giữa phím Shift, Ctrl và chuột)

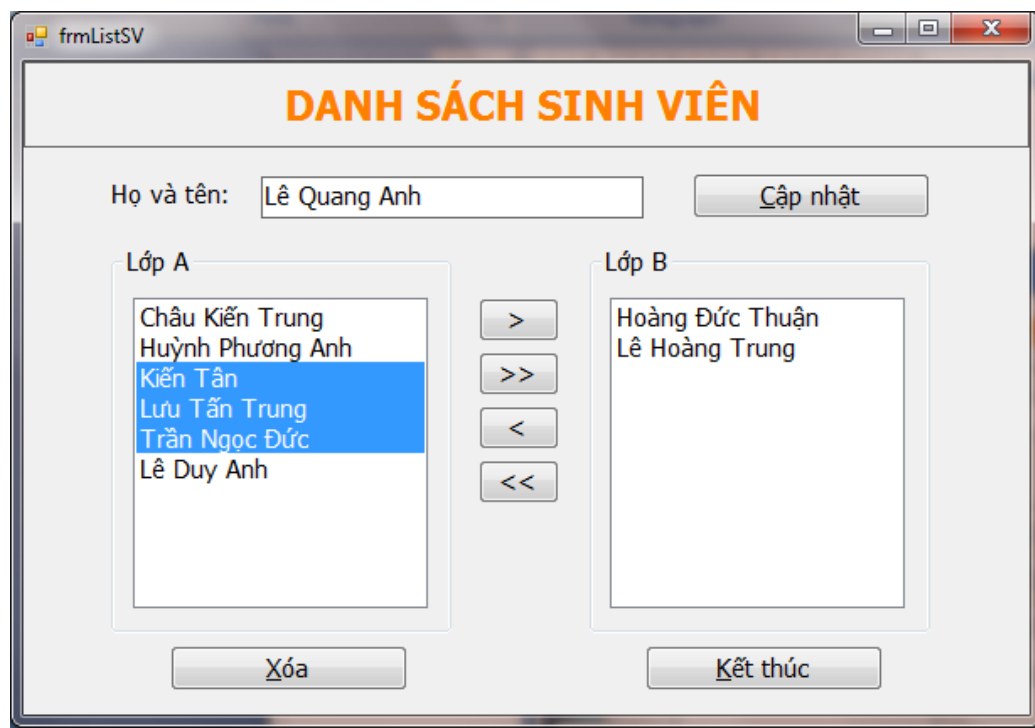
Khi người dùng nhập Họ và tên của sinh viên vào Textbox, click nút Cập Nhật (hoặc Enter) thì tên sinh viên đó sẽ được đưa vào danh sách lớp A (không chấp nhận dữ liệu rỗng).

,  chuyển các tên đang chọn từ Listbox trái sang Listbox phải và ngược lại.

,  chuyển hết toàn bộ các tên từ Listbox trái sang Listbox phải và ngược lại.

Nút Xóa: cho phép xóa các tên đang chọn trong danh sách lớp A.

Thêm vào giao diện 1 combobox Lớp, trong đó có 2 lớp: Lớp A, Lớp B, theo đó người sử dụng có thể chọn lớp để cập nhật sinh viên vào lớp mong muốn.



Hướng dẫn:

// copy selected strings in the source list to the destination list

```
for (int i = 0; i < SourceListbox.SelectedItems.Count; i++)
```

```
{
```

```
    DestinationListbox.Items.Add(SourceListbox.SelectedItems[i]);
```

```
}
```

```
// remove selected strings from the source list
for(int j = SourceListbox.SelectedItems.Count - 1; j >= 0; j --)
{
    SourceListbox.Items.Remove(SourceListbox.SelectedItems[j]);
}
}
```

Bài 6. Viết chương trình cho phép sinh viên đăng ký học các môn học trong học kỳ:

Yêu cầu:

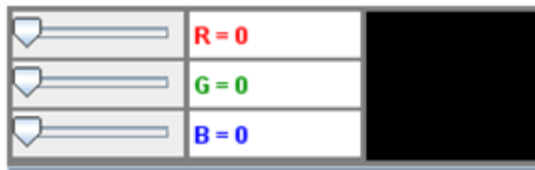
Khi Form hiện lên, các ô nhập đều để trống (thiết lập tab hợp lý).

Nút Đăng ký: Hiện thị các thông tin mà sinh viên đã đăng ký lên MessageBox như hình:

Nút Hủy: trả lại trạng thái ban đầu của Form.

Nút Thoát: thoát khỏi ứng dụng.

Bài 7. Thiết kế giao diện cho phép đổi màu Panel như sau:



Để đổi màu Panel, người dùng có thể kéo các TrackBar để thay đổi các giá trị màu red(R), green(G) và blue(B). Yêu cầu khi TrackBar nào kéo đến đâu thì giá trị của nó phải được hiện ra tương ứng trên các Label. Các TrackBar có giá trị trong khoảng 0 đến 255.

BÀI TẬP THỰC HÀNH 6: LÀM VIỆC VỚI WINDOWS FORM (tiếp)

1. Mục tiêu

+ Sử dụng các thuộc tính, phương thức và sự kiện của các control: TrackBar, NumericUpDown, MaskedTextBox, DateTimePicker, MonthCalendar, Timer, ProgressBar, ToolTip, MenuStrip.

+ Viết ứng dụng dạng MDI

+ ListView, ImageList, TreeView

2. Yêu cầu:

+ Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio

+ Yêu cầu sinh viên: nắm vững kiến thức về các lập trình trên Window Form và sử dụng các điều khiển trên Window Form Application

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1. Thiết kế giao diện như sau:

The image shows a screenshot of a Windows Form titled "Employee Details". The form contains the following fields and controls:

- Employee Name:
- Date of Birth: (mm/dd/yyyy)
- Address:
- City: (Ho Chi Minh, Nha Trang, Ha Noi)
- Country: (Select)
- Qualification: (University, Master, Ph D)
- Phone:
- EMail:
- Date of Joining: (11/ 6/2009)

Annotations with arrows point from the form fields to two boxes on the right:

- A box labeled "TextBox" has an arrow pointing to the "Employee Name" field.
- A box labeled "MaskedTextBox" has arrows pointing to the "Date of Birth", "Phone", and "E-Mail" fields.

At the bottom of the form, there is a link "Link To VnExpress" and two buttons: "Submit" and "Exit".

Hướng dẫn:

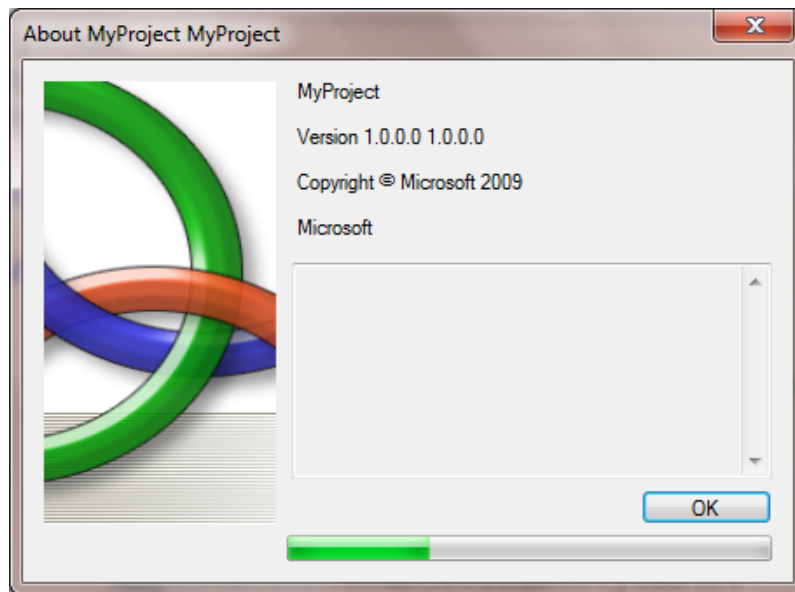
Quy định của Masktextbox Phone là 000-0000000.

ComboBox Country chỉ chứa 2 nước VietNam và Thailan (chứa 3 thành phố Pattaya, Chiang Mai và Bangkok).

Khi đang nhập 1 ô mà bỏ trống và focus đến ô khác thì sẽ có thông báo lỗi và cho focus về ô cần nhập.

Khi nhấn Submit sẽ có một MessageBox hiển thị đầy đủ thông tin vừa nhập.

Bài 2. Tạo giao diện Form Splash như hình: chứa Progressbar và một nút OK.



Khi khởi động chương trình thì Form Splash xuất hiện. Form này dừng trong thời gian là 15s.

Thanh Progressbar sẽ thể hiện tiến trình load form này, sau 15s Form Splash sẽ tự động tắt và khởi động Form trong bài 1.

Trong khi Form Splash đang hoạt động, nếu người dùng nhấn vào nút OK thì Form này ngưng hoạt động, đồng thời Form bài 1 được gọi hoạt động.

Hướng dẫn: kéo thả thanh Progressbar vào Form. Bắt sự kiện cho nút OK

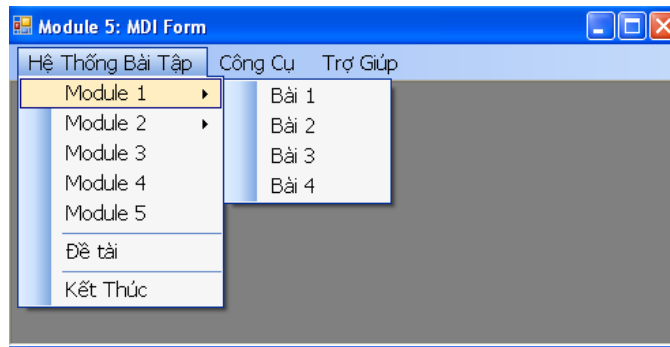
3.2. Các bài thực hành tương tự

Bài 1:

- + Tạo một ứng dụng liên kết các ứng dụng đã tạo ra từ các project trước.
- + Tạo form Splash
- + Tạo form About

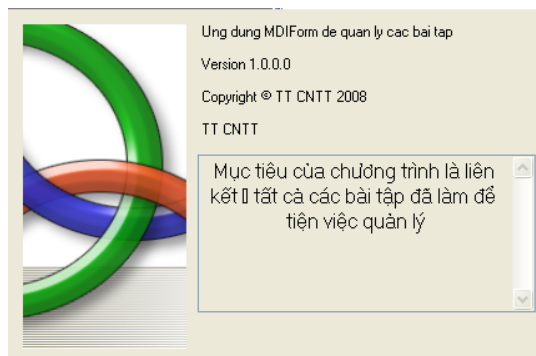
Yêu cầu:

Tổ chức Form chính như mẫu sau:



Yêu cầu liên kết các project đã có sẵn vào menu.

Thiết kế Form Splash cho chương trình (tùy ý):



Thiết kế Form About cho chương trình, tùy ý nhưng phải mang thông tin về chương trình như: tên chương trình, phiên bản, tác giả,...

Hướng dẫn:

Thiết lập một số thuộc tính của Form About:

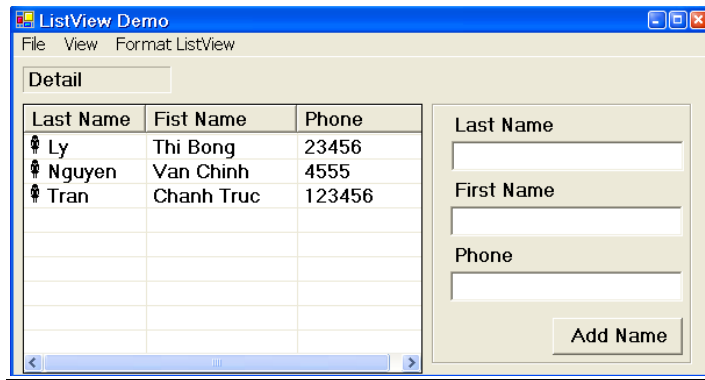
ControlBox → False

FormBorderStyle → FixedDialog

ShowInTaskbar → False

Bài 2. Thêm vào bài 1 chức năng cho phép người dùng mở dialog chọn màu để chọn màu cho Panel.

Viết chương trình nhập dữ liệu vào Listview như hình:



Yêu cầu:

Người sử dụng nhập thông tin Last name, First name, Phone và sử dụng nút Add Name để nhập vào Listview.

Các dòng trong Listview có biểu tượng (icon) hiển thị như hình.

Người sử dụng có thể thay đổi chế độ view của Listview bằng menu View.

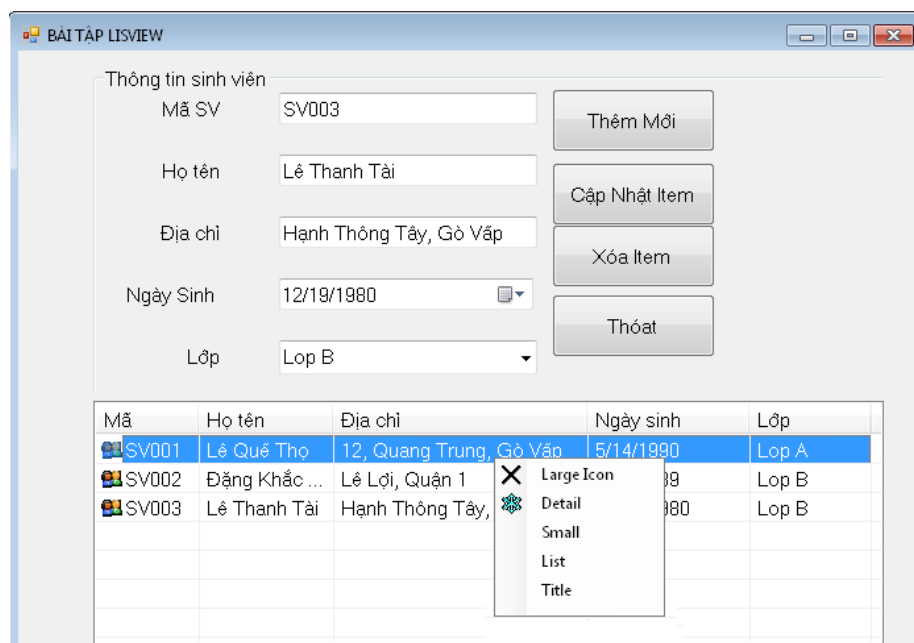
Menu FormatListView hiển thị hộp thoại chọn màu dùng để thay đổi dạng grid của Listview.

Gợi ý:

Sử dụng ListViewItem để thêm một dòng mới cho Listview.

Sử dụng ImageList để chứa thư viện icon cho Listview. Kết nối Listview với ImageList.

Bài 3. Thiết kế chương trình quản lý SV, cho phép nhập thông tin SV vào các Textbox như hình:



Yêu cầu:

Thêm vào Form hai Imagelist là *ilsNho* có kích thước mặc định 16 x 16, *ilsLon* có kích thước mặc 48 x 48 phục vụ cho ListView.

Nhấn nút Cập Nhật Item thì đưa thông tin sinh viên vào Listview theo các cột như hình.

Nhấn nút Xóa Item là xóa item đang chọn trên Listview (có thể chọn nhiều). Trước khi xóa cần xác nhận đã chọn Item nào chưa, xác nhận có chắc xóa không.

Nhấn nút Thêm Mới thì xóa thông tin sinh viên đang nhập và cho phép nhập thông tin sinh viên mới.

Nếu chọn một sinh viên nào trong Listview thì hiện lại thông tin Sinh Viên đó lên các Textbox tương ứng.

Click phải vào Listview cho phép hiện menu ngữ cảnh để chọn chức năng view.

Bài 4. Thiết kế giao diện như sau:

The screenshot shows a Windows application window titled "Tính tiền điện". The main area is divided into two sections. On the left, there is a form titled "BÁO CÁO TIÊU THU ĐIỆN" (Electricity Consumption Report) with the following fields: "HỌ TÊN KH:" (Customer Name), "KHU VỰC:" (Area) with a dropdown arrow, "ĐỊNH MỨC:" (Rate), "SỐ CŨ:" (Old Meter Reading), "SỐ MỚI:" (New Meter Reading), "TIÊU THU:" (Consumption), and "THÀNH TIỀN:" (Amount). Below these fields are three buttons: "TÍNH TIỀN" (Calculate), "NHẬP MỚI" (New Entry), and "THOÁT" (Exit). On the right, there is a list view with a header row containing five columns: "Họ tên", "Khu vực", "Định mức", "Tiêu thụ", and "Thành tiền". Below the header, the list view is currently empty. At the bottom right of the window, there is a label "TỔNG TIỀN:" (Total Amount) followed by a text box containing the value "0" and a button labeled "XÓA" (Delete).

Thực hiện các yêu cầu sau:

Thiết lập thuộc tính cho phép chọn nhiều dòng trên Listview.

Combobox có 3 khu vực: Khu vực 1 (định mức là 50), khu vực 2 (định mức là 100), khu vực 3 (định mức là 150). Khi chọn khu vực nào thì hiện định mức tương ứng.

Nút TÍNH TIỀN (hoặc Enter trên các textbox): kiểm tra dữ liệu nhập, nếu hợp lệ thì tính và xuất kết quả ra ô *Tiêu thụ* và *Thành tiền*, đồng thời thêm một dòng tương ứng vào Listview và cập nhật ô tổng tiền.

Đơn giá điện: trong định mức là 500, ngoài định mức là 1000.

Nút NHẬP MỚI: Xóa nội dung các textbox và label, đồng thời đặt con trỏ vào textbox đầu tiên

Nút XÓA: cho phép xóa 1 dòng đang chọn trong Listview, phải xác nhận lại trước khi xóa và cập nhật lại ô tổng tiền.

Nút THOÁT (hoặc nhấn Esc): thoát chương trình

Quy định Form hiển thị giữa màn hình.

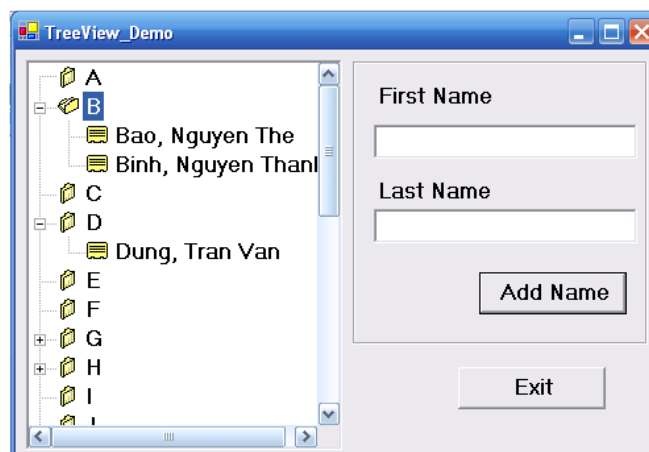
Quy định việc di chuyển tab hợp lý.

Thiết lập thuộc tính Anchor hợp lý cho các control.

Thiết lập MinimumSize cho form.

3.3.Các bài thực hành nâng cao

Bài 1. Viết chương trình nhập danh bạ với yêu cầu giao diện như hình dưới.



Yêu cầu:

Khi chương trình vừa hiển thị, Treeview chứa tất cả các chữ cái từ A->Z.

Nhằm mục đích tiện lợi cho người sử dụng khi tìm tên, khi người sử dụng nhập tên của một người nào đó, chương trình sẽ đưa tên người này vào Treeview ở vị trí node có tương ứng với chữ cái đầu của tên (xem hình).

Bài 2. Viết chương trình xem danh sách SV của Khoa Tin học như hình:

[illegible]

Yêu cầu:

Khi Form hiện lên, Treeview hiển thị danh sách các lớp – sinh viên như hình, chưa có nút nào được chọn. Con trỏ đặt tại ô Nhập tên.

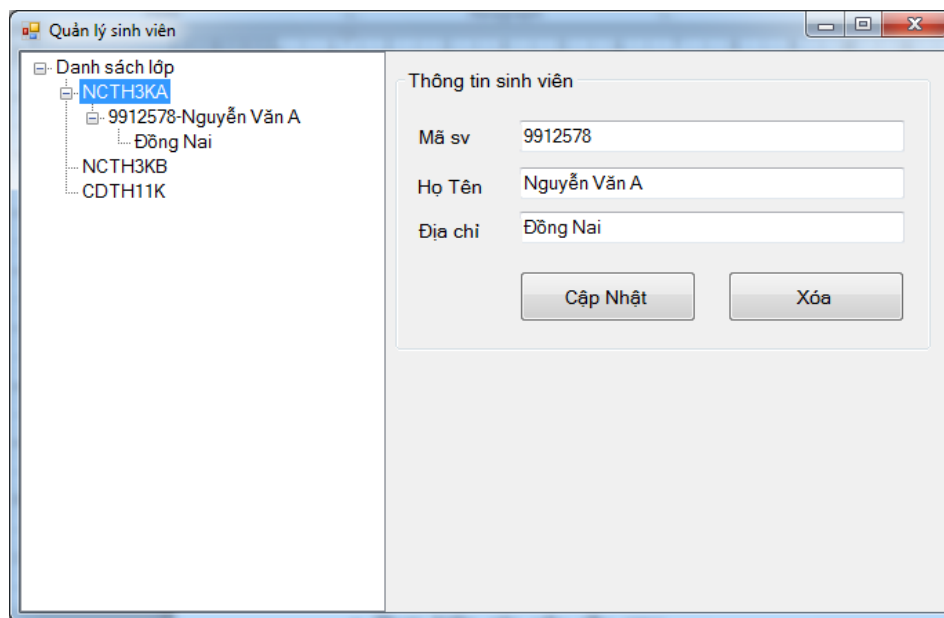
Khi người dùng chọn nút cấp Khoa, chương trình hiện toàn bộ danh sách SV thuộc Khoa vào Listview.

Khi người dùng chọn một lớp bất kỳ thì chương trình hiện toàn bộ danh sách SV thuộc lớp đang chọn vào Listview.

Khi chọn 1 SV bất kỳ thì chỉ hiện thị SV đó vào Listview.

Nút Tìm: cho phép tìm SV (trong cấp đang chọn trên Treeview) có họ tên chứa chuỗi nhập trong Textbox. Hiện kết quả ra Listview.

Bài 3. Thiết kế giao diện như sau:



Thực hiện các yêu cầu sau:

Thiết lập HideSelection = False.

Khi Form hiện lên, đã có sẵn 1 số lớp trong danh sách lớp ở Treeview.

Nút Cập Nhật: Thêm 1 SV vào lớp đang chọn trên Treeview với nội dung các nút như hình. Trước khi thêm phải kiểm tra thông tin nhập gồm: các ô nhập không được để trống, không được trùng mã SV. Ngoài ra còn phải kiểm tra nút chọn trên Treeview có phải là nút lớp không (chỉ được thêm vào nút lớp).

Nút Xóa: cho phép xóa nút đang chọn trong Treeview, phải xác nhận lại trước khi xóa và chỉ được xóa khi chọn nút chứa mã SV.

Khi click chọn nút mã SV hoặc địa chỉ thì hiện thông tin sv đó qua các Textbox.

Quy định Form hiển thị giữa màn hình.

Quy định việc di chuyển tab hợp lý.

Thiết lập thuộc tính Dock hợp lý cho Treeview.

Thiết lập MinimumSize cho form.

BÀI TẬP THỰC HÀNH 7: TRUY CẬP DỮ LIỆU VỚI ADO.NET

1. Mục tiêu

- Tìm hiểu một số khái niệm liên quan đến ADO.NET: Data Provider, Kiến trúc của ADO.NET.
- Sử dụng các đối tượng ADO.NET: SqlConnection, SqlCommand, SqlDataReader, SqlDataAdapter, DataSet, DataTable

2. Yêu cầu:

- + Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- + Yêu cầu sinh viên: nắm vững kiến thức về các cách kết nối cơ sở dữ liệu,

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1: Lập trình phần mềm Quản lý thông tin sách có thêm các chức năng thêm mới, xóa thông tin sách từ CSDL

Mã sách	Tiêu đề	Giá	Số lượng
CSH8	Murach's C# 2008	54.5000	5136
DB1R	DB2 for the COBOL Programmer, Part 1 (2n...	42.0000	4825
DB2R	DB2 for the COBOL Programmer, Part 2 (2n...	45.0000	621
JSE6	Murach's JAVA SE 6	52.5000	3455
JSP2	Murach's JAVA Servlets and JSP (2nd Editi...	52.5000	4999

Chức năng của ứng dụng được mô tả như sau:

- Khi load ứng dụng
- Hiển thị thông tin về tất cả các loại sách trong ứng dụng trên ListView
- Các TextBox đều bị vô hiệu hóa
- Button Thêm mới được kích hoạt cho phép thêm thông tin sách mới
- Button Lưu, Xóa, Bỏ qua bị vô hiệu hóa
- Khi một cuốn sách trên ListView được chọn

- Thông tin về cuốn sách được hiển thị trên các TextBox vẫn đang bị vô hiệu hóa
- Button Thêm mới, Xóa được kích hoạt
- Button Lưu, Bỏ qua bị vô hiệu hóa

Mã sách	Tiêu đề	Giá	Số lượng
CSH8	Murach's C# 2008	54.5000	5136
DB1R	DB2 for the COBOL Programmer, Part 1 (2n...	42.0000	4825
DB2R	DB2 for the COBOL Programmer, Part 2 (2n...	45.0000	621
JSE6	Murach's JAVA SE 6	52.5000	3455
JSP2	Murach's JAVA Servlets and JSP (2nd Editi...	52.5000	4999

Hình 2. Form khi một item trên ListView được chọn .

- Khi button Thêm mới được nhấn
- Các TextBox được kích hoạt cho phép nhập thông tin
- Button Lưu được kích hoạt cho phép lưu thông tin
- Button Bỏ qua được kích hoạt cho phép bỏ qua thao tác Thêm mới
- Button Thêm mới, Xóa, bị vô hiệu hóa
- Khi button Xóa được nhấn
- Hiện thị thông báo hỏi người dùng có chắc chắn xóa không
- Xóa thông tin

Yêu cầu: Thực hiện thao tác thêm hàng, xóa hàng trên DataSet, sau đó cập nhật lại CSDL

1. DATABASE

```
use master
go
drop database sachdb
go
```

```
CREATE DATABASE SachDB
```



```

go
use SachDB
go
create table categories
(
categoryid int identity(1,1) primary key,
categoryname varchar(50)
)
go

insert into categories(categoryname) values ('Sách .net')
go
insert into categories(categoryname) values ('Sách java')
go
CREATE TABLE [dbo].[Products](

[ProductCode] [char](10) NOT NULL PRIMARY KEY,

[Description] [varchar](50) NOT NULL,

[UnitPrice] [money] NOT NULL,

[OnHandQuantity] [int] NOT NULL,

categoryid int

constraint fk_c_p foreign key(categoryid) references categories(categoryid)

)

GO

INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],
[OnHandQuantity],categoryid) VALUES (N'A3CS', N'Murach"s ASP.NET 3.5
Web Programming with C# 2008', 54.5000, 4637,1)

INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],
[OnHandQuantity],categoryid) VALUES (N'A3VB', N'Murach"s ASP.NET 3.5
Web Programming with VB 2008', 54.5000, 3974,1)

INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],
[OnHandQuantity],categoryid) VALUES (N'ADC3', N'Murach"s ADO.NET 3.5,
LINQ, and EF with C# 2008', 54.5000, 5244,2)

INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],
[OnHandQuantity],categoryid) VALUES (N'ADV3', N'Murach"s ADO.NET 3.5,
LINQ, and EF with VB 2008', 54.5000, 4538,2)

```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'CRFC    ', N'Murach''s CICS Desk Reference',  
50.0000, 1865)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'CSH8    ', N'Murach''s C# 2008', 54.5000, 5136)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'DB1R    ', N'DB2 for the COBOL Programmer,  
Part 1 (2nd Edition)', 42.0000, 4825)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'DB2R    ', N'DB2 for the COBOL Programmer,  
Part 2 (2nd Edition)', 45.0000, 621)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'JSE6    ', N'Murach''s JAVA SE 6', 52.5000, 3455)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'JSP2    ', N'Murach''s JAVA Servlets and JSP (2nd  
Edition)', 52.5000, 4999)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'MCBL    ', N'Murach''s Structured COBOL',  
62.5000, 2386)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'MCCP    ', N'Murach''s CICS for the COBOL  
Programmer', 54.0000, 2368)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'MDOM    ', N'Murach''s JavaScript and DOM  
Scripting', 54.5000, 6937)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'SQL8    ', N'Murach''s SQL Server 2008',  
52.5000, 2465)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'VB08    ', N'Murach''s Visual Basic 2008',  
54.5000, 2193)
```

```
INSERT [dbo].[Products] ([ProductCode], [Description], [UnitPrice],  
[OnHandQuantity]) VALUES (N'ZJLR    ', N'Murach''s OS/390 and z/os JCL',  
62.5000, 677)
```

```
go  
select * from categories
```

```

go
select * from products
go

```

2. CODING

```

public partial class Form1 : Form
{
public Form1()
{
InitializeComponent();
}

private void load_data()
{
    SqlConnection con = new SqlConnection("server=.;database=sachdb;integrated
security=true;");
    SqlDataAdapter da = new SqlDataAdapter("select * from products",con);
    DataTable tb = new DataTable();
    da.Fill(tb);
    dataGridView1.DataSource = tb;

    //databinding
    textBox1.DataBindings.Clear();
    textBox2.DataBindings.Clear();
    textBox3.DataBindings.Clear();
    textBox4.DataBindings.Clear();

    textBox1.DataBindings.Add("Text", dataGridView1.DataSource,
"productcode");
    textBox2.DataBindings.Add("Text", dataGridView1.DataSource,
"Description");
    textBox3.DataBindings.Add("Text", dataGridView1.DataSource, "UnitPrice");
    textBox4.DataBindings.Add("Text", dataGridView1.DataSource,
"OnHandQuantity");

}

private void Form1_Load(object sender, EventArgs e)
{
    load_data();
    textBox1.Enabled = false;
    textBox2.Enabled = false;
    textBox3.Enabled = false;
    textBox4.Enabled = false;

    button2.Enabled = false;

```

```

        button3.Enabled = false;
        button4.Enabled = false;
    }

    private void button1_Click(object sender, EventArgs e)
    {
        textBox1.Enabled = true;

        textBox2.Enabled = true;
        textBox3.Enabled = true;
        textBox4.Enabled = true;
        textBox1.Text = "";
        textBox2.Text = "";
        textBox3.Text = "";
        textBox4.Text = "";
        textBox1.Focus();

        button1.Enabled = false;
        button3.Enabled = false;
        button2.Enabled = true;
        button4.Enabled = true;
    }

    private void button2_Click(object sender, EventArgs e)
    {
        SqlConnection con = new SqlConnection("server=.;database=sachdb;integrated
        security=true;");
        SqlCommand cmd = new SqlCommand("insert into products values('" +
        textBox1.Text + "',''" + textBox2.Text + "',''" + textBox3.Text + "',''" +
        textBox4.Text + "')", con);
        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();
        load_data();
    }

    private void button3_Click(object sender, EventArgs e)
    {
        DialogResult kq = MessageBox.Show("ban muon xoa khong ?", "tieu
        de", MessageBoxButtons.YesNo);

        if (kq == System.Windows.Forms.DialogResult.Yes)
        {
            SqlConnection con = new SqlConnection("server=.;database=sachdb;integrated
            security=true;");
            SqlCommand cmd = new SqlCommand("delete from products where
            productcode = '"+textBox1.Text+"'", con);

```

```

        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();
        load_data();
    }
}

```

```

private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)
{
    button3.Enabled = true;
    button4.Enabled = true;
}

```

```

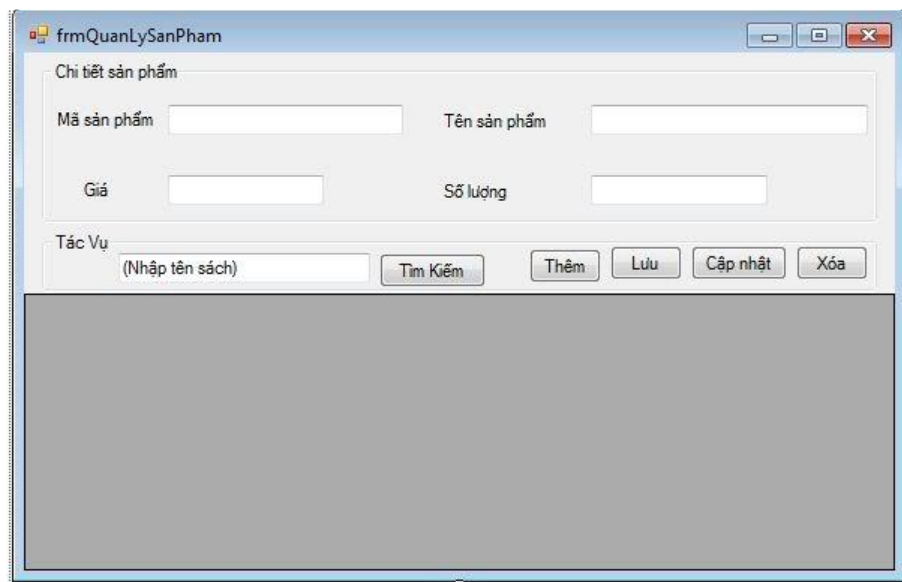
private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
{
    dataGridView1_CellClick(sender, e);
}
}

```

3.2. Các bài thực hành tương tự

Bài 1. Thực hiện các thao tác với DataGridView xây dựng chương trình quản lý sản phẩm sách: hiển thị dữ liệu sản phẩm từ CSDL (SQL), thêm dữ liệu, xóa dữ liệu, cập nhật dữ liệu, tìm kiếm dữ liệu.

Thực hiện việc bắt lỗi nếu số lượng sản phẩm không phải là số nguyên và >100. Demo chương trình bằng hình ảnh.



3.3. Các bài thực hành nâng cao

Bài 1: Viết chương trình quản lý Sinh Viên C# với database SQL trong đó

- Load dữ liệu từ file SQL vào Visual Studio
- Click hiện thông tin từ bảng DataGridView lên các TextBox
- Sử dụng được các nút Thêm, Xóa, Sửa, Lưu, Hủy, Thoát
- Có sử dụng việc bắt lỗi

Form1

QUẢN LÝ THÔNG TIN SINH VIÊN Cách 2

Mã SV: T05 Họ SV: Tam Tên SV: Ga
Mã Khoa: CN Ngày sinh: 09/23/1985 Giới tính: Nam

Mã SV	Họ SV	Tên SV	Ngày Sinh	Giới tính	Mã Khoa
C04	Nguyễn Hoàng	Hưng	03/19/1990	Nam	CN
T00	Lê	Tuấn	02/15/1991	Nam	TO
T01	Bùi Minh	Khánh	04/09/1990	Nam	TO
T02	Trần Thị	Lan	03/04/1990	Nữ	TO
T03	Lê	Thiện	05/18/1990	Nam	TO
T04	Lê Thị	Thảo	03/27/1990	Nữ	TO
T05	Tam	Ga	09/23/1985	Nam	CN

Thêm Sửa Xóa Lưu Hủy Thoát

BÀI TẬP THỰC HÀNH 8: TRUY CẬP DỮ LIỆU VỚI ADO.NET (tiếp)

1. Mục tiêu

Sinh viên vẫn tiếp tục làm các bài thực hành liên quan tới cơ sở dữ liệu nhưng tập trung vào phần thống kê đưa ra danh sách dựa vào các điều kiện có trước. Ngoài ra, hướng dẫn một số cách làm khác để sinh viên học tập.

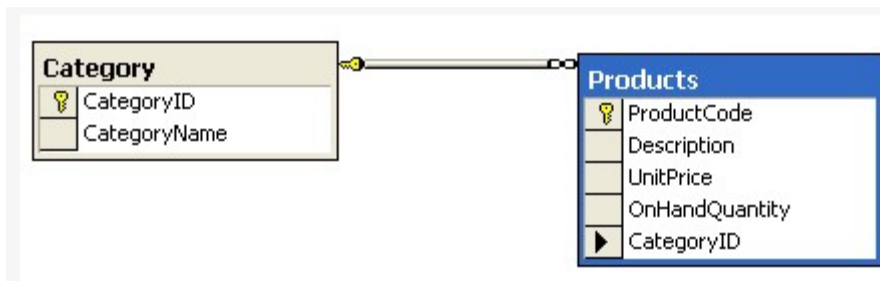
2. Yêu cầu:

- + Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- + Yêu cầu sinh viên: nắm vững kiến thức về các cách kết nối cơ sở dữ liệu, như hiển thị dữ liệu, thêm, sửa, xóa dữ liệu, tìm kiếm.

3. Nội dung

3.1. Bài thực hành mẫu

Bài 1:



Thông tin sách

Loại sách:

Mã sách:

Tiêu đề:

Giá:

Số lượng:

Mã sách	Tiêu đề	Giá	Số lượng
A3CS	Murach's ASP.NET 3.5 Web Programming ...	54.5000	4637
A3VB	Murach's ASP.NET 3.5 Web Programming ...	54.5000	3974
ADC3	Murach's ADO.NET 3.5, LINQ, and EF with...	54.5000	5244
ADV3	Murach's ADO.NET 3.5, LINQ, and EF with...	54.5000	4538
CRFC	Murach's CICS Desk Reference	50.0000	1865
CSH8	Murach's C# 2008	54.5000	5136

Thêm mới Sửa Lưu Xóa Bỏ qua

Hãy lập trình để thực hiện sau

- a.** Đưa dữ liệu từ bảng Category vào Combobox tên là loại sách
- b.** Thực hiện chức năng nhập mới sách theo Category.
- c.** Hiển thị dữ liệu của bảng Product ra DataGridView

Hướng dẫn:

```
public partial class Form2 : Form
```

```
{
```

```
    public Form2()
```

```
    {
```

```
        InitializeComponent();
```

```
    }
```

```
    private void load_data()
```

```
    {
```

```
        SqlConnection con = new
```

```
        SqlConnection("server=.;database=sachdb;integrated security=true;");
```

```
        SqlDataAdapter da = new SqlDataAdapter("select * from products",  
        con);
```

```
        DataTable tb = new DataTable();
```

```
        da.Fill(tb);
```

```
        dataGridView1.DataSource = tb;
```

```
        //databinding
```

```
        textBox1.DataBindings.Clear();
```

```
        textBox2.DataBindings.Clear();
```

```
        textBox3.DataBindings.Clear();
```

```
        textBox4.DataBindings.Clear();
```



```

        textBox1.DataBindings.Add("Text", dataGridView1.DataSource,
        "productcode");

        textBox2.DataBindings.Add("Text", dataGridView1.DataSource,
        "Description");

        textBox3.DataBindings.Add("Text", dataGridView1.DataSource,
        "UnitPrice");

        textBox4.DataBindings.Add("Text", dataGridView1.DataSource,
        "OnHandQuantity");
    }

    private void Form2_Load(object sender, EventArgs e)
    {
        load_data();

        textBox1.Enabled = false;

        textBox2.Enabled = false;

        textBox3.Enabled = false;

        textBox4.Enabled = false;

        button2.Enabled = false;

        button3.Enabled = false;

        button4.Enabled = false;

        // load dữ liệu lên comboBox

        SqlConnection con = new
        SqlConnection("server=.;database=sachdb;integrated security=true;");

        SqlDataAdapter da = new SqlDataAdapter("select * from categories",
        con);

        DataTable tb = new DataTable();

        da.Fill(tb);

        comboBox1.DataSource = tb;

        comboBox1.DisplayMember = "categoryname";
    }

```

```

        comboBox1.ValueMember = "categoryid";
    }

    private void button1_Click(object sender, EventArgs e)
    {
        textBox1.Enabled = true;

        textBox2.Enabled = true;

        textBox3.Enabled = true;

        textBox4.Enabled = true;

        textBox1.Text = "";

        textBox2.Text = "";

        textBox3.Text = "";

        textBox4.Text = "";

        textBox1.Focus();

        button1.Enabled = false;

        button3.Enabled = false;

        button2.Enabled = true;

        button4.Enabled = true;
    }

    private void button2_Click(object sender, EventArgs e)
    {
        SqlConnection con = new
        SqlConnection("server=.;database=sachdb;integrated security=true;");

        SqlCommand cmd = new SqlCommand("insert into products values('" +
        textBox1.Text + "',''" + textBox2.Text + "',''" + textBox3.Text + "',''" +
        textBox4.Text + "',''+comboBox1.SelectedValue.ToString()+')", con);

        con.Open();
    }

```

```

        cmd.ExecuteNonQuery();

        con.Close();

        load_data();

    }

    private void button3_Click(object sender, EventArgs e)

    {

        DialogResult kq = MessageBox.Show("ban muon xoa khong ?", "tieu
de", MessageBoxButtons.YesNo);

        if (kq == System.Windows.Forms.DialogResult.Yes)

        {

            SqlConnection con = new
            SqlConnection("server=.;database=sachdb;integrated
            security=true;");

            SqlCommand cmd = new SqlCommand("delete from products
            where productcode = " + textBox1.Text + "", con);

            con.Open();

            cmd.ExecuteNonQuery();

            con.Close();

            load_data();

        }

    }

    private void dataGridView1_CellClick(object sender, DataGridViewCellEventArgs e)

    {

        button3.Enabled = true;

        button4.Enabled = true;

    }

```

```

private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)

{

    dataGridView1_CellClick(sender, e);

}

private void button5_Click(object sender, EventArgs e)

{

    SqlConnection con = new
    SqlConnection("server=.;database=sachdb;integrated security=true;");

    //MessageBox.Show(textBox1.Text);

    //MessageBox.Show(textBox2.Text);

    //MessageBox.Show(textBox3.Text);

    //MessageBox.Show(textBox4.Text);

    //MessageBox.Show(comboBox1.SelectedValue.ToString());

    //string sql = "update products set description=" + textBox2.Text +
    "unitprice=" + textBox3.Text +
    ",OnHandQuantity="+textBox4.Text+"categoryid="+comboBox1.S
    electedValue.ToString()+" where productcode="+textBox1.Text+"";

    //MessageBox.Show(sql);

    SqlCommand cmd = new SqlCommand("update products set
description=@description,unitprice=@unitprice,OnHandQuantity=@onh
andquantity,categoryid=@categoryid where
productcode=@productcode", con);

    cmd.Parameters.AddWithValue("@description", textBox2.Text);

    cmd.Parameters.AddWithValue("@unitprice", textBox3.Text);

    cmd.Parameters.AddWithValue("@OnHandQuantity", textBox4.Text);

    cmd.Parameters.AddWithValue("@categoryid",
comboBox1.SelectedValue.ToString());

```

```

        cmd.Parameters.AddWithValue("@productcode", textBox1.Text);

        con.Open();

        cmd.ExecuteNonQuery();

        con.Close();

        load_data();

    }

    private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
    {

        //SqlConnection con = new
        SqlConnection("server=.;database=sachdb;integrated security=true;");

        //SqlDataAdapter da = new SqlDataAdapter("select * from products
        where categoryid=@categoryid", con);

        //da.SelectCommand.Parameters.AddWithValue("@categoryid", comboBox1.SelectedValue);

        //DataSet ds = new DataSet();

        //da.Fill(ds);

        //dataGridView1.DataSource = ds.Tables[0];

    }

}

```

3.2. Các bài thực hành tương tự

Bài 1: Viết chương trình thống kê sinh viên theo Môn học bằng C# với database SQL trong đó:

- Load dữ liệu từ file SQL vào Visual Studio
- Click hiển thị thông tin từ ComboBox sẽ hiện lên Textbox và DataGridView
- Có sử dụng Class để kết nối

Mã Môn Học Số tiết

Tên Môn Học

	Mã SV	Họ SV	Tên SV	Ngày sinh	Điểm
▶	s001	Tran Minh	Son	01/05/1985	4
	s001	Tran Minh	Son	01/05/1985	6
	s007	Phan Thi	Ha	03/07/1988	2
	s007	Phan Thi	Ha	03/07/1988	9
	s008	Tran The	Dung	21/10/1985	7

Hướng dẫn:

- Xây dựng lớp kết nối

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.SqlClient;
using System.Data;

namespace _39_SQLTamGa_ThongKeTheoMonHoc
{
    public class KetNoiDuLieu
    {
        public SqlConnection cnn = new SqlConnection
            ("Data Source=(local);Initial Catalog=QLSV;Integrated Security=True");

        public void myconnect()
        { cnn.Open(); }

        public void myclose()
        { cnn.Close(); }

        // www.tamga.tk www.c10mt.tk www.c10maytinh.tk
        public DataTable taobang(string sql)
        {
            DataTable dt = new DataTable();
            SqlDataAdapter ds = new SqlDataAdapter(sql, cnn);
            ds.Fill(dt);
            return (dt);
        }
    }
}

```

Các hàm trong Form

```
public Form1()
{
    InitializeComponent();
}

KetNoiDulieu kn = new KetNoiDulieu();

private void Form1_Load_1(object sender, EventArgs e)
{
    kn.myconnect();
    string sql = "SELECT * FROM MONHOC";
    cbMaMon.DataSource = kn.taobang(sql);
    cbMaMon.DisplayMember = "MAMH";
}

private void btThoat_Click(object sender, EventArgs e)
{
    this.Close();
    kn.myclose();
}

// www.tamga.tk www.c10mt.tk www.c10maytinh.tk
private void cbMaMon_SelectedIndexChanged_1(object sender, EventArgs e)
{
    // load dữ liệu từ combobox xuống text
    string s = "select * from monhoc where mamh='" + cbMaMon.Text + "'";
    DataTable d = kn.taobang(s);
    foreach (DataRow hang in d.Rows)
        KhungTenMon.Text = hang["TENMH"].ToString();
    foreach (DataRow hang in d.Rows)
        KhungSoTiet.Text = hang["SOTIET"].ToString();

    // load dữ liệu lên DataGridView
    string s2 = "select sv.masv, hosv, tensv, ngaysinh, diem " +
        "from SinhVien SV, KetQua KQ " +
        "where (SV.masv = KQ.masv) and (KQ.mamh= '" + cbMaMon.Text + "')";
    dataGridViewMH.DataSource = kn.taobang(s2);
}
```

Bài 2: Viết chương trình thống kê sinh viên theo Khoa bằng C# với database SQL trong đó:

- Load dữ liệu từ file SQL vào Visual Studio
- Click hiện thông tin từ ComboBox sẽ hiện lên Textbox và DataGridView
- Có sử dụng lớp để kết nối dữ liệu

Form1

THỐNG KÊ SỐ SINH VIÊN THEO KHOA

Mã Khoa: Tên Khoa:

	Mã SV	Họ SV	Tên SV	Ngày Sinh
▶	s001	Tran Minh	Son	01/05/1985
	s002	Nguyen Quoc	Bao	16/05/1986
	s006	Nguyen Thi	Lam	11/11/1984
	s007	Phan Thi	Ha	03/07/1988
	s008	Tran The	Dung	21/10/1985
	s001	Tam	Ca	24/10/2012

Tổng số Sinh Viên:

```

/*
TamGa
www.tamga.tk www.c10mt.tk www.c10maytinh.tk
www.tamga85.multiply.com
Phone: 01283.98.69.98 Email : tamgaalbum@yahoo.com
*/

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Data.SqlClient;
using System.Data;

namespace _40_SQLTamGa_ThongKeTheoKhoa
{
    public class KetNoiDuLieu
    {
        public SqlConnection cnn = new SqlConnection
            ("Data Source=(local);Initial Catalog=QLSV;Integrated Security=True");

        public void myconnect()
        { cnn.Open(); }
    }
}

```



```

Form1.cs  KetNoiDuLieu.cs  Form1.cs [Design]
_40_SQLTamGa_ThongKeTheoKhoa.KetNoiDuLieu  taobang(string sql)

public void myclose()
{ cnn.Close(); }

// www.tamga.tk www.c10mt.tk www.c10maytinhtk
public DataTable taobang(string sql)
{
    DataTable dt = new DataTable();
    SqlDataAdapter ds = new SqlDataAdapter(sql, cnn);
    ds.Fill(dt);
    return (dt);
}
}

```

```

namespace _40_SQLTamGa_ThongKeTheoKhoa
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();

            KetNoiDuLieu kn = new KetNoiDuLieu();

            private void btThoat_Click(object sender, EventArgs e)
            {
                this.Close();
                kn.myclose();
            }

            // www.tamga.tk www.c10mt.tk www.c10maytinhtk
            private void Form1_Load(object sender, EventArgs e)
            {
                kn.myconnect();
                string sql = "select * from Khoa";
                cbMaKhoa.DataSource = kn.taobang(sql);
                cbMaKhoa.DisplayMember = "MAKHOA";
            }

            private void cbMaKhoa_SelectedIndexChanged(object sender, EventArgs e)
            {
                // load dữ liệu từ combobox xuống text
                string s = "select * from khoa where makhoa='" + cbMaKhoa.Text + "'";
                DataTable d = kn.taobang(s);
                foreach (DataRow hang in d.Rows)
                    KhungTenKhoa.Text = hang["TENKHOA"].ToString();
            }
        }
    }
}

```

```

// load dữ liệu lên DataGridView
string s1 = "select masv, hosv, tensv, ngaysinh " +
            "from SinhVien SV, Khoa KH " +
            "where (SV.makhoa = KH.makhoa) " +
            "and (KH.makhoa= '" + cbMaKhoa.Text + "')";
dataGridViewKhoa.DataSource = kn.taobang(s1);

string s2 = "select count(*) from SinhVien " +
            "where makhoa='" + cbMaKhoa.Text + "' group by makhoa";
KhungTong.Text = (dataGridViewKhoa.Rows.Count).ToString();
}

```

3.3. Các bài thực hành nâng cao

Bài 1. Tạo 1 ứng dụng đơn giản để quản lý sinh viên. Giao diện được thiết kế như hình

Các bạn thiết kế 1 ComboBox để hiển thị tên lớp, 1 textBox hiển thị tên giáo viên chủ nhiệm, 1 textBox hiển thị số sinh viên có trong lớp.. 1 listBox hiển thị thông tin chi tiết về sinh viên bao gồm tên, ngày sinh, địa chỉ. Thiết kế cơ sở dữ liệu như sau:

Bảng Student

Column Name	Data Type	Allow Nulls
stuNo	varchar(5)	<input type="checkbox"/>
stuName	varchar(50)	<input checked="" type="checkbox"/>
classNo	varchar(50)	<input type="checkbox"/>
stuYear	datetime	<input checked="" type="checkbox"/>
address	varchar(100)	<input checked="" type="checkbox"/>
		<input type="checkbox"/>

Bảng StudentClass

THAOTRINH.Stude...bo.StudentClass		THAOTRINH.Stude...nt - dbo.Student	
	Column Name	Data Type	Allow Nulls
🔑	classNo	varchar(50)	<input type="checkbox"/>
	className	varchar(30)	<input type="checkbox"/>
	totalStudent	int	<input checked="" type="checkbox"/>
	homeroomTeacher	varchar(50)	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Bài 2: Cho CSDL quản lý sinh viên bao gồm các table:

- ✓ Khoa (makhoa, tenkhoa)
- ✓ Lop (malop, tenlop, makhoa)
- ✓ Sinhvien (masv, hoten, ngaysinh, malop)
- ✓ Monhoc (mamh, tenmh)
- ✓ Diem (masv, mamh, diem)

Thực hiện các yêu cầu sau:

1) Xây dựng Form quản lý Khoa

a. Giao diện như sau:

b. Yêu cầu

Formload:

- + Datagrid: Hiển thị tất cả khoa trong bảng Khoa
- + Tất cả textbox bị vô hiệu hóa
- + Các nút Sửa, xóa, Lưu bị vô hiệu hóa

Khi chọn vào nút Thêm:

- + Các textbox có hiệu lực

- + Nút Lưu có hiệu lực
- + Dấu nháy xuất hiện ở textbox Mã khoa.

Khi chọn vào Datagrid

- + Hiển thị thông tin tương ứng lên các textbox
- + Nút Sửa và Xóa có hiệu lực

Chọn nút Sửa

- + Nút Lưu có hiệu lực
- + Các textbox có hiệu lực trừ textbox Mã khoa
- + Cho phép sửa các thông tin còn lại

Khi chọn nút “Lưu”

- + Kiểm tra thông tin vừa nhập hoặc sửa cho phù hợp
- + Lưu vào Cơ sở dữ liệu (lưu ý đang lưu Thêm hay Sửa)
- + Thông báo thành công hoặc báo lỗi nếu có
- + Nút Lưu bị vô hiệu hóa

Khi nhấn nút Xóa.

- + Hiển thị thông báo xác nhận
- + Nếu đồng ý
 - ✓ Kiểm tra mã khoa định xóa có tồn tại trong bảng lớp hay không. Nếu có
 - ✓ thì hỏi xác nhận “có xóa luôn trong bảng lớp hay không?”. Nếu đồng ý,
 - ✓ xóa dữ liệu trong table Lớp tương ứng với mã khoa đang chọn (nếu có);
 - ✓ xóa khoa vừa chọn.
 - ✓ Nếu mã khoa không tồn tại bên bảng lớp thì xóa bên bảng khoa
- + Hiển thị thông báo nếu xóa thành công hoặc báo lỗi (nếu có)

2) Xây dựng Form quản lý Sinh viên hoàn chỉnh

- a. Giao diện như sau

b. Yêu cầu

Formload:

- + Combobox mã khoa: Chứa tên khoa trong bảng khoa
- + Combobox mã lớp: Chứa tên lớp trong bảng lop
- + Datagrid sinh viên: Hiện thị tất cả sinh viên trong bảng sinh viên và chỉ đọc
- + Tất cả textbox, combobox bị vô hiệu hóa
- + Các nút Sửa, xóa, Lưu bị vô hiệu hóa

Khi chọn vào nút Thêm:

- + Nút Lưu có hiệu lực
- + Cho phép thêm các dòng tiếp theo trên Datagrid

Lưu ý: không được sửa đổi các dòng trên Datagrid đã có dữ liệu

Khi chọn vào Datagrid

- + Hiện thị thông tin tương ứng lên các textbox, combobox
- + Nút Sửa và Xóa có hiệu lực

Chọn nút Sửa

- + Nút Lưu có hiệu lực
- + Cho phép sửa các thông tin trên Datagrid
- + Lưu ý: không cho phép gõ thêm các dòng mới

Khi chọn nút “Lưu”

- + Kiểm tra thông tin vừa nhập hoặc sửa trên lưới cho phù hợp
- + Lưu vào Cơ sở dữ liệu (lưu ý đang lưu Thêm hay Sửa)
- + Thông báo thành công hoặc báo lỗi nếu có
- + Nút Lưu bị vô hiệu hóa

Khi nhấn nút Xóa.

- + Hiện thị thông báo xác nhận
- + Nếu đồng ý

Xóa dữ liệu trong table diem tương ứng với mã số sinh viên đang chọn (nếu có) (nhớ cảnh báo nhắc nhở)

Xóa dữ liệu trong table sinh viên tương ứng với mã số sinh viên đang chọn

Hiện thị thông báo nếu xóa thành công hoặc báo lỗi (nếu có)

3) Xây dựng Form Điểm

a. Giao diện như sau

a. Yêu cầu

FormLoad:

- + Hiện thị Khoa lên Combobox Khoa
- + Hiện thị Lớp lên Combox Lớp
- + Hiện thị Mã sinh viên lên Combobox Sinh viên
- + Hiện thị môn học lên combobox môn học
- + Datagrid hiện thị dữ liệu trong bảng Điểm
- + Hiện thị dòng đầu tiên trong Datagrid lên các control

Chọn Combobox Khoa

+ Hiện thị lớp tương ứng với khoa vừa chọn

Chọn combobox Lop:

+ Hiện thị sinh viên thuộc lớp đó

Chọn combobox sinh viên

+ Hiện thị Điểm của sinh viên đó

Chọn combobox Môn học

+ Hiện thị điểm của môn học đó (lưu ý xem có chọn sinh viên hay không? Nếu sinh viên được chọn thì hiện thị điểm của sinh viên được chọn tương ứng)

BÀI TẬP THỰC HÀNH 9: MỘT SỐ KỸ THUẬT LẬP TRÌNH NÂNG CAO TRONG .NET (LÀM QUEN VỚI LINQ)

1. Mục tiêu

Sinh viên thực hành xây dựng một số bài tập sử dụng thành thạo các kỹ thuật lập trình nâng cao như Generic, Lambda Expressions, Anonymous Types... để thực hành lại phần lý thuyết đã học và củng cố kiến thức.

2. Yêu cầu:

- Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- Yêu cầu sinh viên: nắm vững kỹ thuật lập trình nâng cao trong C# sử dụng Windows Form để thể hiện các kỹ thuật này.

3. Nội dung

- Hiểu được Generic
- Hiểu được Implicitly Typed Variables
- Hiểu được Anonymous Types
- Hiểu được Extension Methods
- Hiểu được Lambda Expressions

3.1. Bài thực hành mẫu

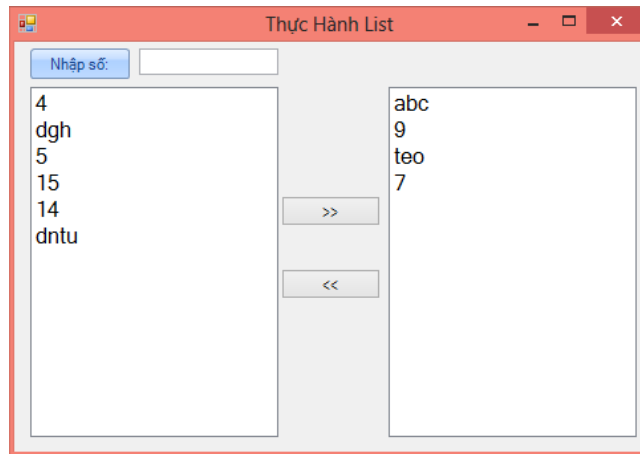
Bài tập 1: Generic

Mục đích:

- Hiểu được Generic
- Trong **System.Collections.Generic**; cung cấp rất nhiều class: List, Dictionary, SortedList, SortedDictionary, LinkedList, HashSet....
- Nhưng ở đây chúng ta chỉ quan tâm tới **List** và **Dictionary**, các class khác sinh viên phải có trách nhiệm tự nghiên cứu.

Yêu cầu:

- Sử dụng **List** để viết chương trình như bên dưới:



- Mỗi lần click “Nhập số”: Sẽ đưa số trong TextBox vào ListBox bên trái.
- Click “>>” sẽ chuyển tất cả các phần tử đang được chọn trong ListBox bên trái sang ListBox bên phải
- Click “<<” sẽ chuyển tất cả các phần tử đang được chọn trong ListBox bên phải sang ListBox bên trái
- Cải tiến lại chương trình: cho phép đưa bất kỳ kiểu dữ liệu nào vào ListBox (xem hình)

Hướng dẫn:

- Thiết lập các ListBox có chế độ cho phép chọn nhiều phần tử: SelectionMode là **MultiSimple** hoặc **MultiExtended**.
- Để List có thể chứa bất kỳ kiểu dữ liệu nào thì ta dùng List<**object**>. Mọi class sinh ra đều kế thừa từ object, nên khi ta để object thì nó sẽ có thể lưu bất kỳ kiểu dữ liệu nào.

Bài tập 2:

Mục đích:

- Thực hành và hiểu được **Implicitly Typed Variables**

Yêu cầu:

- Hãy khai báo và gán giá trị bất kỳ cho một biến có kiểu **var**

`var x = 113;`

`var y = "1/1/2012";`

`var z = 1.7;`

`var k = new DateTime(2012, 1, 1);`

`string msg = "x type="+x.GetType() + "\n"+`

`"y type = "+y.GetType() + "\n" +`

```
"z type =" + z.GetType() + "\n" +
```

```
"k type = " + k.GetType();
```

```
MessageBox.Show(msg);
```

- Hãy thử trường hợp không gán giá trị mặc định cho biến có kiểu **var**, cho nhận xét
- Hãy thử gán giá trị có kiểu dữ liệu khác cho biến đã khai báo kiểu **var**, cho nhận xét
- Hãy cho biết **var** thường được ứng dụng trong trường hợp nào?
- Trong trường hợp đã biết chính xác kiểu dữ liệu thì có nên khai báo kiểu **var** hay không?

Hướng dẫn:

- **var** được .net framework hỗ trợ cơ chế nội suy kiểu dữ liệu
- do đó nó sẽ tự động lấy đúng kiểu dữ liệu khi ta gán vào cho nó.
- Bất kỳ kiểu dữ liệu nào, kể cả kiểu object, var cũng tự động nội suy ra đúng kiểu dữ liệu của nó:

var s=new **sinhvien**(); thì s cũng được nội suy ra là đối tượng sinh viên, do đó ta có thể sử dụng các Properties, method .. của sinhvien một Cách bình thường.

Bài tập 3:

Mục đích:

- Thực hành và hiểu được **Anonymous Types**

Yêu cầu:

- Hãy kiểm tra một số đoạn lệnh sau:

```
var teo = new { ID=1234, Name="Tèo Hả Tèo"};  
MessageBox.Show(teo.ID + "-" + teo.Name);
```
- Sinh viên tự tạo thêm nhiều Anonymous Types để hiểu thêm về nó.
- Anonymous Types được sử dụng trong nhiều tình huống khác nhau.

Hướng dẫn:

- Anonymous Types được sử dụng khi:
 - ✓ Ta cần một đối tượng tạm thời để lưu trữ dữ liệu
 - ✓ Khi ta không cần định nghĩa phương thức
 - ✓ Khi chúng ta muốn tạo thêm các Properties khác ngoài class đã định nghĩa
 - ✓ Khi ta muốn thay đổi thứ tự các properties...

Bài tập 4:

Mục đích:

- Thực hành và hiểu được Extension Methods:
 - ✓ Quy tắc tạo extension methods như thế nào?
 - ✓ Cách sử dụng chúng ra sao?

Yêu cầu:

- Hãy cài một hàm tính tổng các số từ 1 tới N vào kiểu số nguyên
- Hãy cài một hàm kiểm tra số nguyên tố vào kiểu số nguyên
- Hãy cài một hàm xuất danh sách các số nguyên tố vào kiểu số nguyên
- Hãy cài một hàm xuất danh sách dãy số Fibonacci vào kiểu số nguyên
- Hãy cài một hàm cho phép nối 2 chuỗi vào kiểu chuỗi
- Hãy cài một hàm cho phép tô màu đỏ vào Button
- Hãy cài một hàm cho phép tô đen các số chẵn vào ListBox
- Hãy cài một hàm cho phép tô đen các số lẻ vào ListBox
- Hãy cài một hàm cho phép tô đen các số nguyên tố vào ListBox
- Dùng delegate làm mặt nạ để thực hiện tô số chẵn, tô số lẻ, tô số nguyên tố cho ListBox

Hướng dẫn:

```
namespace StudyLinq
{
    public static class MyExtensionMethod
    {
        public static int SumFrom1toN(this int n)
        {
            int sum = 0;
            for (int i = 1; i <= n; i++)
                sum += i;
            return sum;
        }
    }
}
```

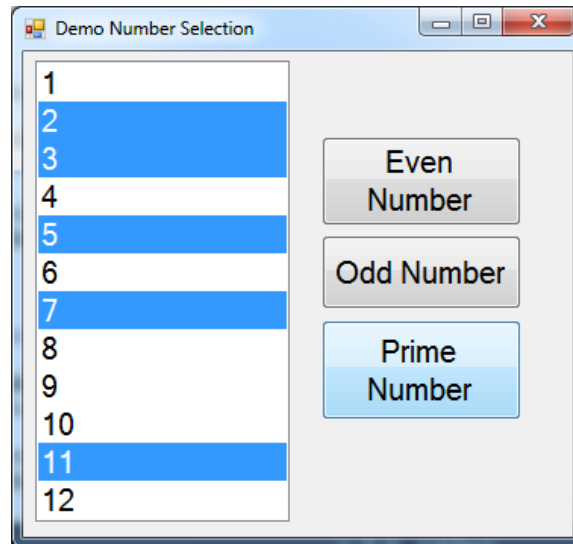
Bài tập 5:

Mục đích:

- Thực hành và hiểu được Lambda Expressions

Yêu cầu:

- Sử dụng Lambda Expressions để làm bài tập như hình dưới đây:



- Thực hiện 3 chức năng:
 - ✓ Tô đen số chẵn
 - ✓ Tô đen số lẻ
 - ✓ Tô đen số nguyên tố

Hướng dẫn:

- Kết hợp Extension methods với Lambda Expression để làm bài tập này:

`Listbox1.todensochan(x => x % 2 == 0);`

- Với hàm **todensochan**, viết theo kỹ thuật Extension methods

```
public static class MyExt
{
    public static void todensochan(this ListBox lb)
    {
        //viết code tô đen ở đây
    }
}
```

3.2. Các bài tập thực hành tương tự

Bài tập 1 :

Mục đích:

- Hiểu được Generic
- Trong **System.Collections.Generic**; cung cấp rất nhiều class: List, Dictionary, SortedList, SortedDictionary, LinkedList, HashSet....

- Nhưng ở đây chúng ta chỉ quan tâm tới **List** và **Dictionary**, các class khác sinh viên phải có trách nhiệm tự nghiên cứu.

Yêu cầu:

- Sử dụng **Dictionary** để viết chương trình như bên dưới:

STT	Mã sinh viên	Tên sinh viên
1	010103	Nguyễn Văn Tèo
2	020103	Trần Thị Tí
3	040302	Nguyễn Ngọc An

Thông tin chi tiết

Mã sinh viên:

Tên sinh viên:

- Mỗi lần bấm Lưu: Đưa thông tin sinh viên vào Dictionary rồi cập nhật lên ListView. Nếu mã đã tồn tại thì tự động cập nhật, nếu mã chưa tồn tại thì thêm mới.
- Mỗi lần click vào từng phần tử trong ListView thì hiển thị thông tin chi tiết của sinh viên vào phần Thông tin chi tiết
- Nút Xóa: cho phép xóa sinh viên hiện tại
- Chú ý là tất cả các thao tác phải sử dụng Dictionary.
- Khi thao tác với CSDL thì Dictionary là một class rất hữu hiệu, cải tiến tốc độ xử lý.

Hướng dẫn:

- Tạo một class tên là Sinhvien
- Khai báo Dictionary như sau:
`Dictionary<string,sinhvien> dic = new Dictionary<string, sinhvien>();`
 Với đối số thứ nhất là key, đối số thứ 2 là value
- Ta có thể đưa dữ liệu vào bằng phương thức `dic.Add("010203",sinhvien nào đó)`
- Để duyệt toàn bộ các phần tử trong Dictionary có nhiều cách, ở đây ta thường dùng:

```
foreach (KeyValuePair<string, sinhvien> item in dic)
{
    sinhvien sv = item.Value;
}
```

- Để lấy sinh viên theo đúng mã ta chỉ cần: dic["010203"]
- Để xóa sinh viên ta làm như sau: dic.Remove("010203");

3.3. Các bài thực hành nâng cao

Bài tập 1:

Mục đích:

- Thực hành và hiểu được Query Syntax và Method Syntax
- So sánh được Query Syntax với Method Syntax
- Hiểu được **Deferred execution** và **lazy loading**

Yêu cầu:

- Sinh viên hãy thực hành lại các đoạn lệnh bên dưới đây:

- Lệnh số 1:

```
string[] list = new string[] { "teo", "ty", "bin", "bo"};
```

```
var ret1 = from c in list
           where c.StartsWith("t")
           select c;
```

```
IEnumerable<string> ret2 =
    list.Where(c=>c.StartsWith("t"));
```

- Lệnh số 2:

```
string[] list = new string[] { "teo", "ty", "bin", "bo"};
```

```
var ret1 = from c in list
           where c.EndsWith("o")
           orderby c
           select new { Id=c.ElementAt(0),Name=c };
```

```
var ret2 = list
    .Where(c => c.EndsWith("o"))
    .OrderBy(c => c)
    .Select(c=>new { Id=c.ElementAt(0),Name=c});
```

- Lệnh số 3:

```

var source = new List<string> { "A", "B", "C" };
var values = from c in source
              select c;
source.Add("D");
foreach (var c in values)
{
    Console.WriteLine(c);
}

```

Out Put:

A
B
C
D

Hướng dẫn:

- Tương tự như câu truy vấn SQL, Query Syntax cũng dùng quy tắc tương tự nhưng ở đây nó đảo ngược from lên trước. Ở đâu thấy from ... select đó chính là query syntax
- Lambda expression thường được sử dụng trong Method Syntax

BÀI TẬP THỰC HÀNH 10: LẬP TRÌNH NÂNG CAO TRONG C# (LÀM VIỆC VỚI LINQ)

1. Mục tiêu

Sinh viên thực hành xây dựng một số bài tập sử dụng thành thạo các kỹ thuật lập trình nâng cao sử dụng Linq để thực hành lại phần lý thuyết đã học và củng cố kiến thức.

2. Yêu cầu:

- Yêu cầu về điều kiện thực hành: máy PC (laptop), phần mềm visual studio
- Yêu cầu sinh viên: nắm vững kỹ năng trong lập trình Linq, đặc biệt là sử dụng Linq to sql

3. Nội dung

3.1. Bài thực hành mẫu

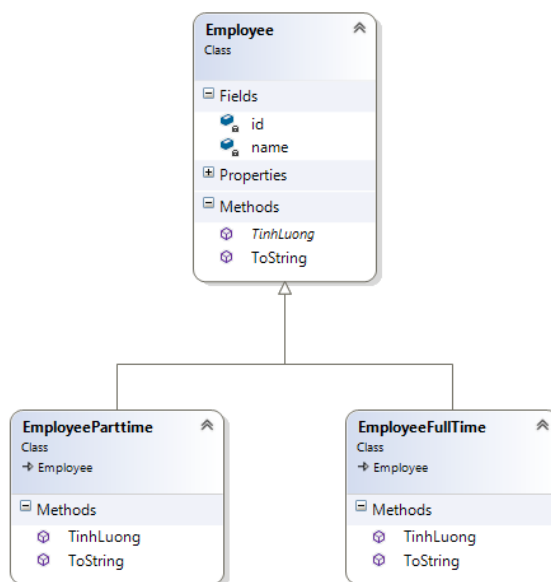
Bài tập 1:

Mục đích:

- Thực hành và hiểu được Linq to Object, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **Select, Where**
 - ✓ Thực hành và hiểu được **OrderBy/OrderByDescending**
 - ✓ Thực hành và hiểu được **OfType, All, Any, Max, Min**
 - ✓ Và các hàm khác...

Yêu cầu:

- Viết chương trình quản lý nhân viên theo mô hình class dưới đây:



- Giao diện chương trình:

- Có 2 loại nhân viên: Nhân viên chính thức và nhân viên thời vụ
- Nhân viên chính thức thì mức lương là 1000.
- Nhân viên thời vụ thì dựa vào số ngày công *30
- Thực hiện các chức năng:
 - ✓ Người sử dụng checked vào “**Là thời vụ**” thì mới hiển thị mục ngày công, nếu không checked thì ẩn mục ngày công, mặc định khi khởi động chương trình thì unchecked “**Là thời vụ**”
 - ✓ Bấm nút “Lưu”, tự động lưu mới khi mã chưa tồn tại, tự động cập nhập khi mã đã tồn tại. Khi lưu thành công thì cập nhập vào ListView
 - ✓ Bấm nút “Xóa”, cho phép xóa nhân viên hiện tại đang chọn
 - ✓ Chỉ duyệt theo tên nhân viên: Phần ListViewEx sẽ chỉ có 2 cột là cột Thứ Tự và cột Tên nhân viên
 - ✓ Thực hiện “sắp nhân viên theo tên tăng dần” → cập nhập lại ListViewEx
 - ✓ Thực hiện “sắp nhân viên theo lương giảm dần” → cập nhập lại ListViewEx
 - ✓ Thực hiện lọc nhân viên có lương từ 300 tới 500 → cập nhập lại ListViewEx
 - ✓ Trong mục xem danh sách nhân viên, chọn loại nhân viên nào thì lọc các nhân viên đó vào ListViewEx

Hướng dẫn:

- Tất cả phải sử dụng **Generic List** và các phương thức của **List**
- Dùng tính đa hình trong hướng đối tượng để xử lý coding thật khoa học

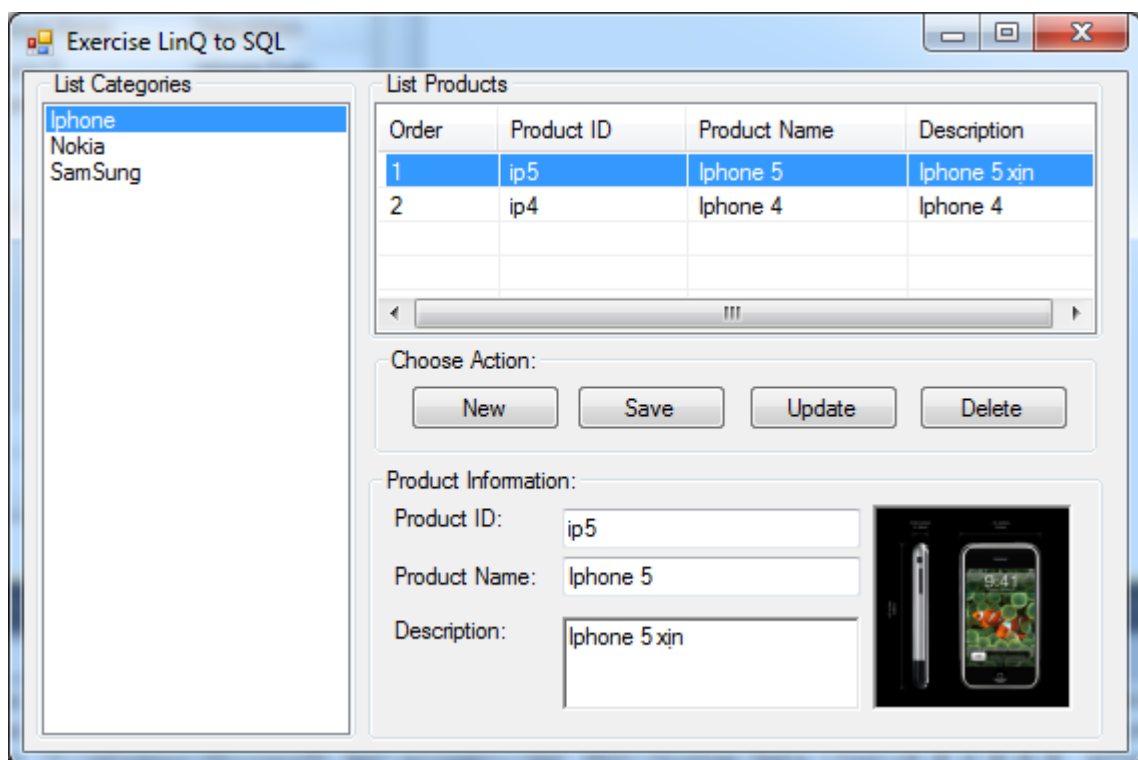
Bài tập 2:

Mục đích:

- Thực hành và hiểu được LinQ to SQL, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **Mapping object và Datacontext**
 - ✓ Thực hành và hiểu được **Query**
 - ✓ Thực hành và hiểu được **Insert, update, delete**
 - ✓ Và các hàm khác...

Yêu cầu:

- Dùng LinQ để viết chương trình quản lý sản phẩm được mô tả đơn giản như sau: Danh mục sản phẩm gồm Mã và tên. Mỗi một danh mục sẽ có nhiều sản phẩm, thông tin mỗi sản phẩm bao gồm: Mã sản phẩm, tên sản phẩm, mô tả và hình ảnh.
- Thiết kế giao diện như hình bên dưới và thực hiện các yêu cầu:



- Khi khởi động chương trình sẽ tải toàn bộ danh mục sản phẩm vào listbox bên trái
- Khi người sử dụng Click chuột vào từng danh mục thì hiển thị danh sách sản phẩm của danh mục đó vào ListView.
- Khi người sử dụng Click chuột vào từng sản phẩm trong Listview thì hiển thị chi tiết thông tin sản phẩm vào phần bên dưới.
- Double click vào PictureBox để cho phép thay đổi hình ảnh sản phẩm
- Thực hiện các thao tác: New, Save, Update, Delete

Hướng dẫn:

- Dùng **MemoryStream** và **Image.FromStream** để đọc hình nhị phân từ CSDL lên giao diện

3.2. Các bài tập thực hành tương tự

Bài tập 1

Mục đích:

- Thực hành và hiểu được LinQ to Object, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **ForEach, Exists, TrueForAll**
 - ✓ Thực hành và hiểu được **Find, FindAll, FindIndex, FindLast, FindLastIndex**
 - ✓ Thực hành và hiểu được **RemoveAll**
 - ✓ Và các hàm khác...
- Hiểu được Generic **List**

Yêu cầu:

- Một sản phẩm cần có các thông tin (mã sản phẩm, tên sản phẩm, số lượng, đơn giá, xuất xứ, ngày hết hạn)
- Hãy viết chương trình quản lý sản phẩm đáp ứng các yêu cầu sau:
 - ✓ Cho phép thêm/ sửa/ xóa sản phẩm
 - ✓ Cho phép duyệt danh sách sản phẩm
 - ✓ Kiểm tra xem trong kho có chứa bất kỳ sản phẩm nào quá hạn hay không?
 - ✓ Tìm 1 sản phẩm có đơn giá cao nhất
 - ✓ Tìm 1 sản phẩm có xuất xứ từ Nhật Bản
 - ✓ Xuất tất cả các sản phẩm bị quá hạn trong kho
 - ✓ Xuất tất cả các sản phẩm có đơn giá trong đoạn [a...b]
 - ✓ Xóa các sản phẩm có xuất xứ bất kỳ
 - ✓ Xóa toàn bộ sản phẩm trong kho
- Giao diện tương tự như bên dưới:

LINQ to OBJECT - Quản lý sản phẩm

Nhập thông tin sản phẩm:

Mã SP:

Tên SP:

Số lượng:

Đơn giá:

Xuất xứ:

Ngày hết hạn:

Chọn chức năng tìm kiếm:

Mã SP	Tên SP	Số lượng	Đơn giá	Xuất xứ	Ngày hết hạn...
sp2	Kem Chiên	12	800	Lào	12/5/2013
sp3	Cá viên nướng	100	500	Việt Nam	30/7/2013

Danh sách sản phẩm sau khi nhập:

Mã SP	Tên SP	Số lượng	Đơn giá	Xuất xứ	Ngày hết hạn
sp1	Sữa Chua Vinamilk	10	15000	Trung Qu...	15/8/2013
sp2	Kem Chiên	12	800	Lào	12/5/2013
sp3	Cá viên nướng	100	500	Việt Nam	30/7/2013

Chọn thao tác:

Hướng dẫn:

Sử dụng Generic List để thao tác với toàn bộ yêu cầu đặt ra.

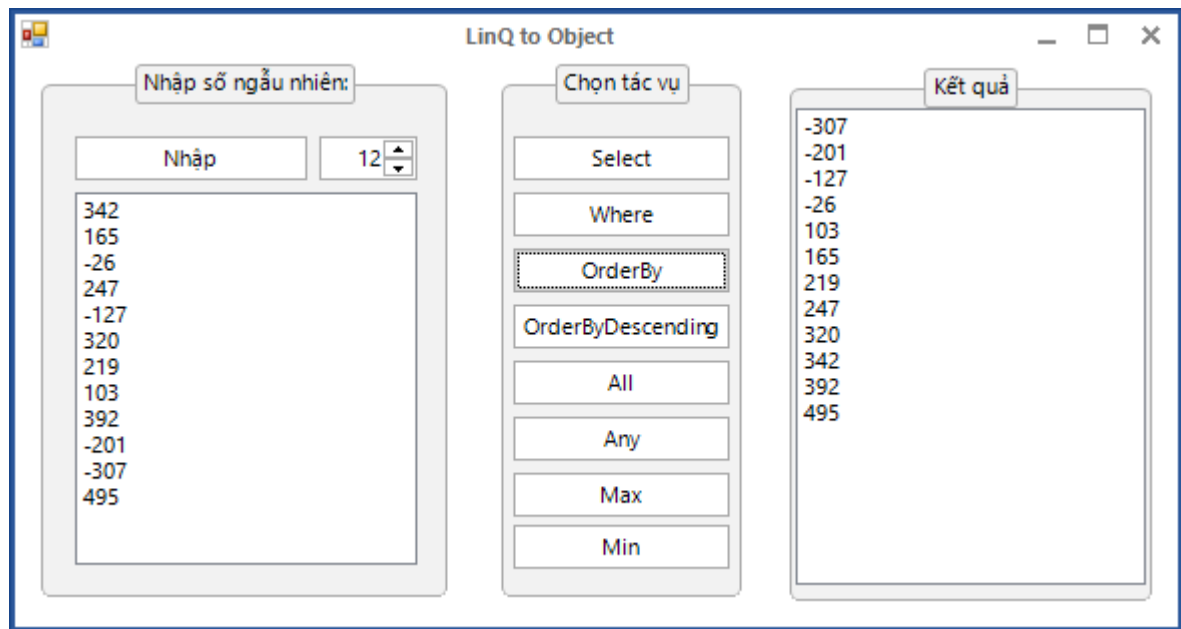
Bài tập 2 :

Mục đích:

- Thực hành và hiểu được LinQ to Object, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **Select**
 - ✓ Thực hành và hiểu được **Where**
 - ✓ Thực hành và hiểu được **OrderBy/OrderByDescending**
 - ✓ Thực hành và hiểu được **All, Any, Max, Min**
 - ✓ Sử dụng **Metro Form** và một số control khác của **DotnetBar**

Yêu cầu:

- Viết chương trình cho phép nhập vào một danh sách N số ngẫu nhiên có giá trị từ -500 tới 500 rồi thực hiện các tác vụ giống như giao diện dưới đây:



- ✓ Select: Lấy ra số hàng đơn vị của mỗi số trong danh sách
- ✓ Where: Trích ra danh sách các số nguyên tố trong danh sách
- ✓ OrderBy: Sắp xếp danh sách tăng dần
- ✓ OrderDescending: Sắp xếp danh sách giảm dần
- ✓ All: Kiểm tra xem có phải tất cả các phần tử trong danh sách là số âm hay không?
- ✓ Any: Kiểm tra xem danh sách có bất kỳ một phần tử nào là số hoàn thiện hay không? (số hoàn thiện là số có tổng các ước số không kể nó thì bằng chính nó, ví dụ $6=1+2+3$, $28=1+2+4+7+14$)
- ✓ Max: xuất ra số lớn nhất trong danh sách
- ✓ Min: xuất ra số nhỏ nhất trong danh sách

Bài tập 3:

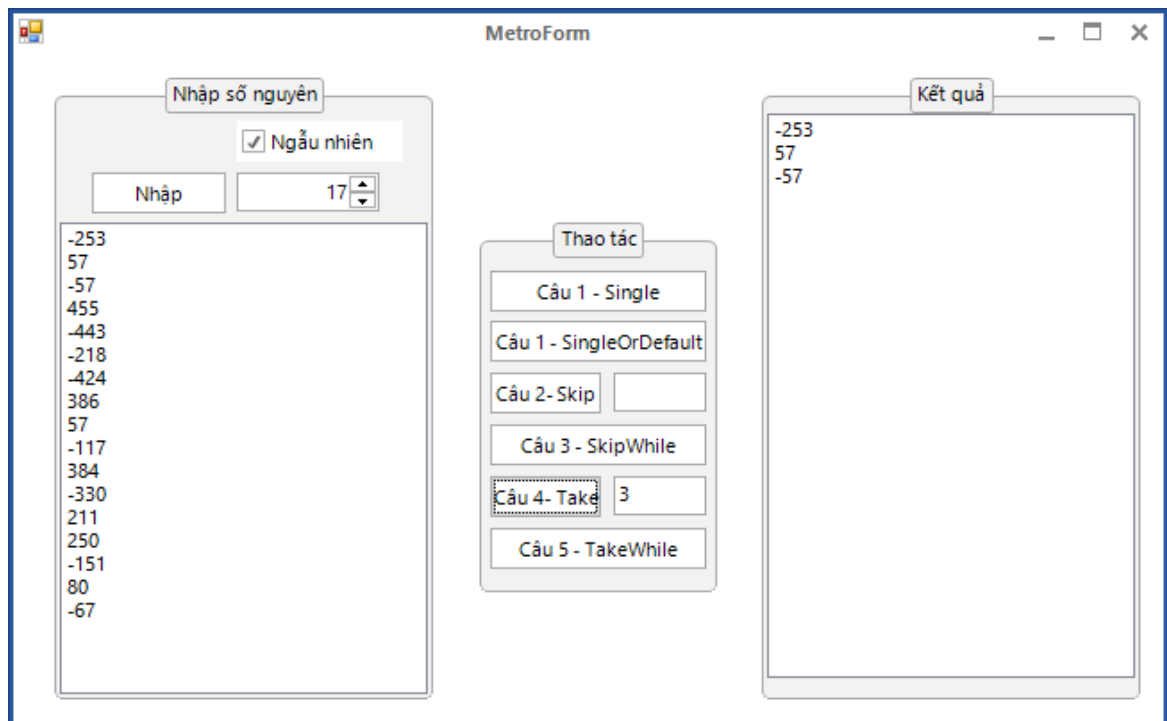
Mục đích:

- Thực hành và hiểu được LinQ to Object, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **Single, SingleOrDefault**
 - ✓ Thực hành và hiểu được **Skip, SkipWhile**
 - ✓ Thực hành và hiểu được **Take, TakeWhile**
 - ✓ Và các hàm khác...

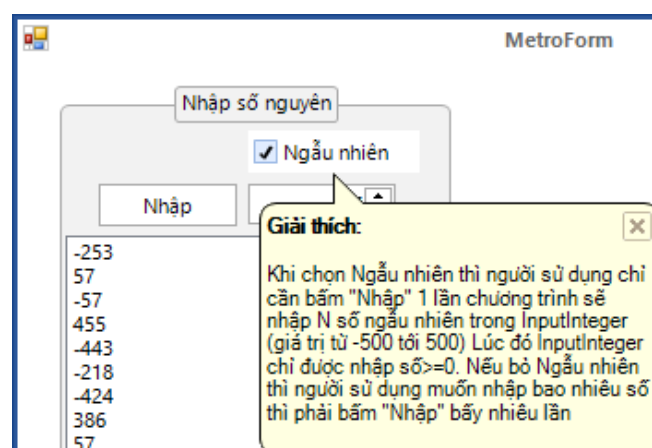
Yêu cầu:

- Viết chương trình cho phép nhập vào một danh sách các số nguyên rồi thực hiện các yêu cầu sau:
 - ✓ Lấy ra một số hoàn thiện trong danh sách, yêu cầu dùng hàm **Single** và **SingleOrDefault** (hãy cho biết chuyện gì xảy ra nếu danh sách có từ 2 số hoàn thiện trở lên)

- ✓ Viết hàm cho phép bỏ qua **n** phần tử đầu tiên trong danh sách (dùng **Skip**)
 - ✓ Viết hàm cho phép loại bỏ các phần tử là số âm liên tiếp đầu tiên trong danh sách (dùng **SkipWhile**)
 - ✓ Viết hàm cho phép trích ra n phần tử đầu tiên trong danh sách (dùng **Take**)
 - ✓ Viết hàm cho phép trích ra các phần tử lớn hơn 50 liên tiếp đầu tiên trong danh sách (Dùng **TakeWhile**)
- Giao diện tương tự như hình dưới đây:



- Dùng **BalloonTip** để gắn cho checkbox Ngẫu nhiên:



Hướng dẫn:

- Tất cả phải sử dụng **Generic List** và các phương thức của **List**

Bài tập 4:

Mục đích:

- Thực hành và hiểu được LinQ to Object, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **ToArray**
 - ✓ Thực hành và hiểu được **ToList**
 - ✓ Thực hành và hiểu được **ToDictionary**
 - ✓ Và các hàm khác...

Yêu cầu:

- Thực hành lại các lệnh sau:
- Kiểu int:

```
List<Int32> myList = new List<int>();  
myList.AddRange(new int[] { 2, 6, 7, 8, 0, 4, 9, 7 });  
var myenumerable = myList.Where(x => x % 2 != 0);  
int[] arr1 = myenumerable.ToArray();  
List<int> list2 = myenumerable.ToList();
```

- Kiểu Object:

```
public class Person  
{  
    public string Id { get; set; }  
    public string Name { get; set; }  
}  
  
List<Person> listPerson = new List<Person>();  
listPerson.Add(new Person() { Id="1",Name="Tèo"});  
listPerson.Add(new Person() { Id = "2", Name = "Hùng" });  
listPerson.Add(new Person() { Id = "3", Name = "Bin" });  
  
//Lấy Id làm key, Person làm value  
Dictionary<string,Person> dic= listPerson.ToDictionary(x => x.Id);  
  
foreach (KeyValuePair<string, Person> item in dic)  
{  
    string Id = item.Key;  
    Person p = item.Value;
```

```

        //Xử lý Id, p
    }
    foreach (Person p in dic.Values)
    {
        //Xử lý từng p
    }
    foreach (string id in dic.Keys)
    {
        //xử lý từng id
    }
}

```

Hướng dẫn:

- Một số Extension Method trả về kiểu **IEnumerable<>** , ta muốn sử dụng một số hàm tiện lợi của List, Dictionary, Array... thì ta dùng các hàm ở trên: **toList**, **toArray**, **toDictionary**

Một số trường hợp ta muốn chuyển qua Dictionary để xử lý theo Key, Value.

Dictionary rất hữu dụng ở việc tối ưu tốc độ xử lý. Khi xử lý liên quan tới CSDL hay đối tượng ta thường dùng Dictionary.

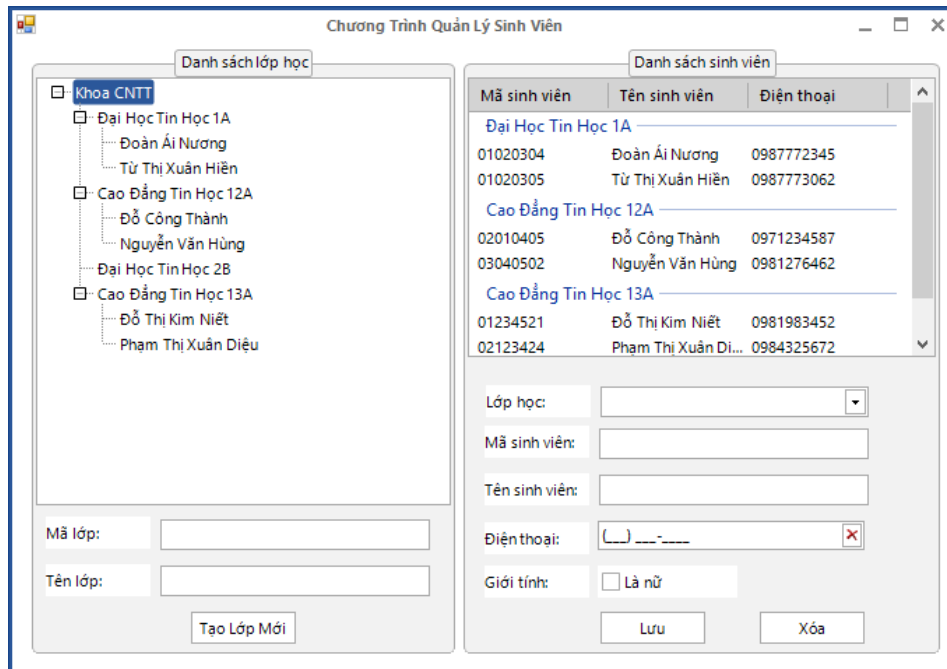
Bài tập 5:

Mục đích:

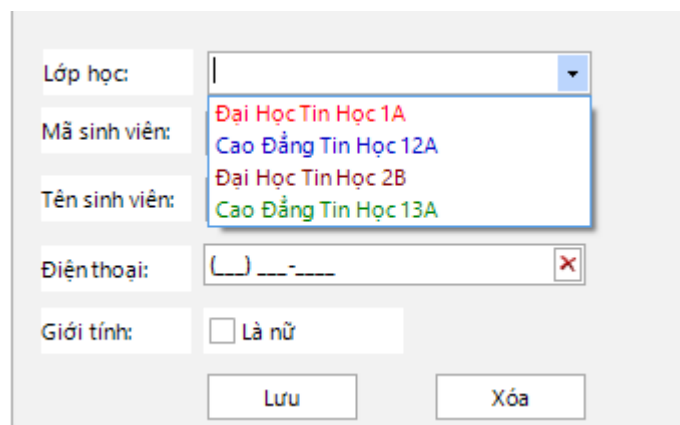
- Thực hành và hiểu được LinQ to SQL, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **Mapping object và Datacontext**
 - ✓ Thực hành và hiểu được **Query**
 - ✓ Thực hành và hiểu được **Insert, update, delete**
 - ✓ Và các hàm khác...

Yêu cầu:

- Viết chương trình quản lý sinh viên, giao diện như hình bên dưới:



- Nút “Tạo Lớp Mới” cho phép tạo lớp mới và đưa vào Adv Tree
- ListViewEx cho phép gom nhóm sinh viên theo lớp, mỗi lần click vào từng phần tử trong ListViewEx thì sẽ hiển thị thông tin chi tiết của sinh viên vào mục bên dưới
- Nút Lưu cho phép lưu thông tin sinh viên theo đúng như tên lớp đã chọn trong ComboBoxX: Cập nhật đồng thời lên ListViewEx và AdvTree



- Sinh viên tự tạo ImageList để gán hình ảnh tương ứng với giới tính.
- Mỗi lần click chuột vào từng phần tử trong Adv Tree :
 - ✓ Nếu chọn vào Node gốc là “Khoa CNTT” thì liệt kê toàn bộ lớp và sinh viên gom nhóm theo lớp vào ListViewEx
 - ✓ Nếu chọn vào Node là Lớp nào đó thì liệt kê toàn bộ sinh viên của lớp đó vào ListViewEx mà thôi
 - ✓ Nếu chọn vào Node là sinh viên thì chỉ hiển thị thông tin của sinh viên đó vào ListViewEx mà thôi
- Nút “Xóa” cho phép xóa sinh viên hiện tại, cập nhật lại ListViewEx và Adv Tree

Hướng dẫn:

- Tạo 2 bảng dữ liệu trong SQL Server: Bảng Lớp (mã lớp, tên lớp).
- Bảng sinh viên (mã sinh viên, tên sinh viên, điện thoại, giới tính)
- Tất cả thao tác phải dùng LinQ to SQL

3.3. Các bài thực hành nâng cao

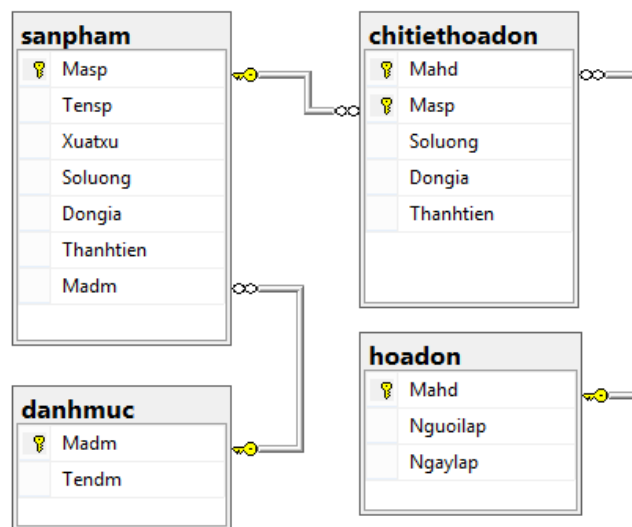
Bài tập 1:

Mục đích:

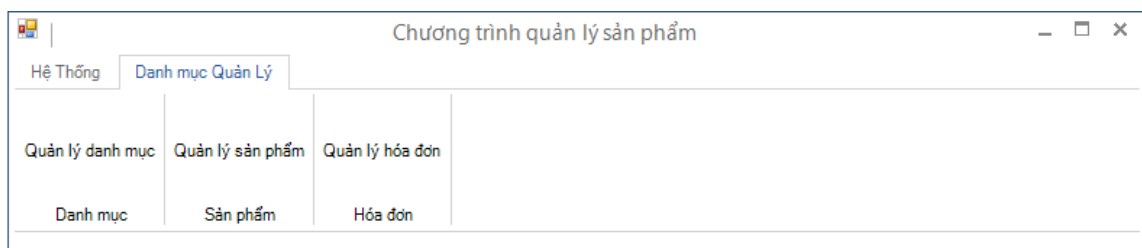
- Thực hành và hiểu được LinQ to SQL, cụ thể ở một số hàm:
 - ✓ Thực hành và hiểu được **Join**
 - ✓ Thực hành và hiểu được **Stored procedure**
 - ✓ Và các hàm khác...

Yêu cầu:

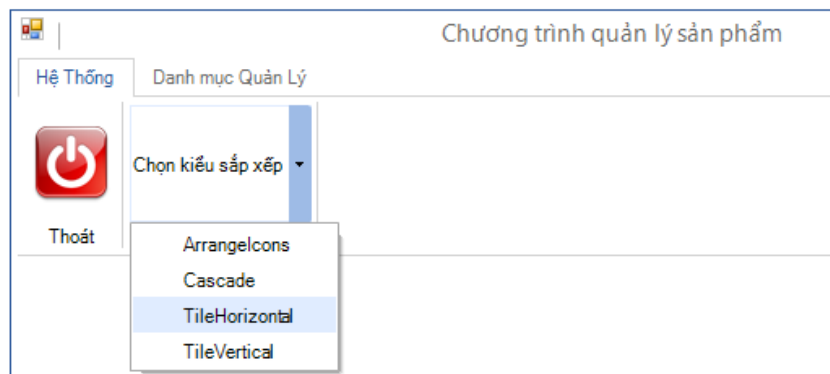
- Công ty XYZ chuyên kinh doanh trong lĩnh vực mua bán sản phẩm, trong những năm gần đây quy mô công ty ngày càng mở rộng nên ban lãnh đạo có nhu cầu viết một phần mềm quản lý sản phẩm, sau đây là một phần nhỏ của ứng dụng:



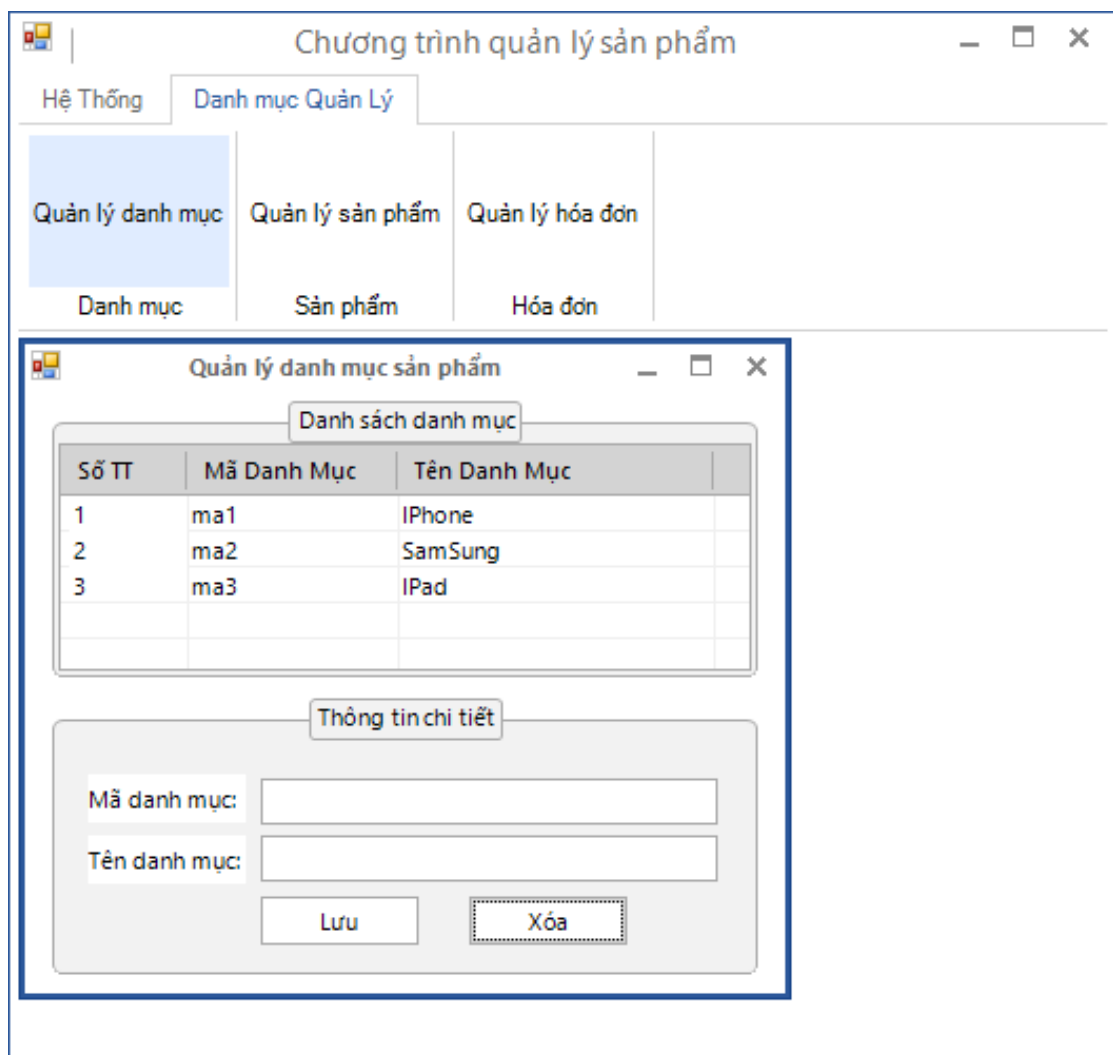
- Dưới đây là giao diện chính của chương trình:



- Tab Danh mục quản lý: Quản lý danh mục, quản lý sản phẩm, quản lý hóa đơn



- Tab Hệ Thống: Thoát, chọn kiểu sắp xếp các form con bên trong
- Khi bấm vào nút **Quản lý danh mục**:



+ Màn hình này cho phép thêm, sửa, xóa danh mục sản phẩm.

- Bấm lưu: Nếu mã danh mục đã tồn tại thì trở thành cập nhật, còn mã danh mục chưa tồn tại thì sẽ tự động thêm mới
- Bấm xóa: sẽ xóa danh mục hiện tại
- Mỗi lần bấm vào từng dòng trong ListViewEx thì sẽ hiển thị thông tin chi tiết của danh mục đó vào phần thông tin chi tiết.

- Khi bấm vào nút **Quản Lý Sản Phẩm**:

+ ListBox cho phép liệt kê toàn bộ danh mục sản phẩm đã được nhập ở phần quản lý danh mục.

+ Mỗi lần chọn từng danh mục trong ListBox thì sẽ hiển thị danh sách các sản phẩm của danh mục đó vào ListViewEx bên phải màn hình

+ Mỗi lần chọn từng sản phẩm trong ListViewEx thì sẽ hiển thị thông tin chi tiết của sản phẩm đó vào mục chi tiết sản phẩm bên dưới.

+ Nút lưu: cho phép thêm mới sản phẩm nếu như mã sản phẩm chưa tồn tại, tự động cập nhật nếu như mã sản phẩm đó đã tồn tại. Chú ý là phải lưu sản phẩm vào đúng danh mục sản phẩm đang được chọn trong ListBox

+ Nút xóa : Cho phép xóa sản phẩm hiện tại đang chọn.

STT	Mã sản phẩm	Tên sản phẩm	Thành tiền
1	i3	iPhone 3	24.0000
2	i4	iPhone 4	45.0000
3	i5	iPhone 5	16.0000

- Khi bấm vào **Quản lý hóa đơn**:

+ Danh sách hóa đơn sẽ được tự động hiển thị vào ListBox ở màn hình bên trái

- + Bấm Tạo hóa đơn cho phép tạo hóa đơn mới theo đúng với thông số nhập vào, cập nhật lại ListBox khi tạo hóa đơn thành công.
- + Nút xóa hóa đơn: cho phép xóa hóa đơn đang chọn
- + Mỗi lần chọn hóa đơn nào thì sẽ hiển thị danh sách các chi tiết của hóa đơn vào mục ListViewEx ở bên phải màn hình.
- + Mỗi lần chọn từng dòng trong ListViewEx thì hiển thị thông tin chi tiết của từng hạng mục trong hóa đơn vào mục nhập thông tin chi tiết
- + Bấm lưu để thêm sản phẩm cho hóa đơn
- + Bấm xóa để xóa chi tiết hóa đơn đang chọn

Chương trình quản lý sản phẩm

Hệ Thống | Danh mục Quản Lý

Quản lý danh mục | Quản lý sản phẩm | Quản lý hóa đơn

Danh mục | Sản phẩm | Hóa đơn

Quản Lý Hóa Đơn

Danh sách hóa đơn

HD01
HD02
HD03

Mã HD: HD01

Người lập: Trần Duy Thanh

Ngày lập: 24/04/2013

Tạo hóa đơn | Xóa hóa đơn

Chi tiết hóa đơn

STT	Mã SP	Tên SP	SL	ĐG	TT
1	i3	IPhone 3	2	3.0000	6.0000
2	ip5	IPhone 5	2	5000.0000	10000.0000

Nhập thông tin chi tiết

Mã sản phẩm: iPhone 3

Số lượng: 8

Đơn giá: 3.0000

Thành tiền: 24.0000

Lưu | Xóa

Hướng dẫn:

- Cố gắng viết các nghiệp vụ bằng Store Procedure