Trường ĐH CT TP. HCM Khoa: CNTT Bộ môn: CNPM Môn: Lập trình .NET	**BÀI 5** **GIAO DIỆN** **(tiếp theo)**	

A. MỤC TIÊU:

- Thiết kế được giao diện Windows Form từ các control trong C#.
- Sử dụng được các Controls Textbox, Label, Button, Listbox, Combobox, TreeView, TabControl.

B. DỤNG CỤ - THIẾT BỊ THỰC HÀNH CHO MỘT SV:

STT	Chủng loại – Quy cách vật tư	Số lượng	Đơn vị	Ghi chú
1	Computer	1	1	

C. NỘI DUNG THỰC HÀNH

1. Cơ sở lý thuyết

1.1 Kiến thức cần nhớ

- **Listbox:**
  - + Thuộc tính: Items, SelectionMode, SelectedIndex, SelectedItem.
  - + Sự kiện: SelectedIndexChanged.
  - + Phương thức: Add, AddRange, RemoveAt, Remove, IndexOf.

Một số thao tác với ListBox:

- **Add()**: Thêm một mục chọn vào cuối danh sách **ListBox**.
- **Insert()**: Chèn thêm mục vào vào vị trí **i**.
- **Count**: Trả về số mục chọn hiện đang có.
- **Item()**: Trả về mục chọn ở vị trí **i**.
- **Remove()**: Bỏ mục chọn.
- **RemoveAt()**: Bỏ mục chọn tại vị trí **i**.
- **Contains()**: Trả về True nếu có mục chọn trong danh sách và trả về False nếu không có mục chọn trong danh sách.
- **Clear**: Xóa tất cả các mục chọn.
- **IndexOf()**: Trả về vị trí mục chọn trong danh sách, nếu không tìm thấy sẽ trả về -1.

Một số thuộc tính của ListBox thường dùng:

Thuộc tính	Mô tả
DataSource	Chọn tập dữ liệu điền vào ListBox. Tập dữ liệu có thể là mảng, chuỗi,

Thuộc tính	Mô tả
	ArrayList,...
DisplayMember	Dữ liệu thành viên sẽ được hiển thị trên ListBox
ValueMember	Thuộc tính này chỉ định dẽ liệu thành viên sẽ cung cấp giá trị cho ListBox
SelectedValue	Trả về giá trị của mục chọn nếu ListBox có liên kết dữ liệu. Nếu không liên kết với dữ liệu hoặc thuộc tính ValueMember không được thiết lập thì giá trị thuộc tính SelectedValue là giá trị chuỗi của thuộc tính SelectedItem
Items	Các mục chứa trong ListBox
SelectedItem	Trả về mục được chọn
SelectedIndex	Lấy chỉ số mục được chọn, chỉ số mục chọn đầu tiên là 0
SelectionMode	Cho phép chọn một hoặc nhiều dòng dữ liệu trên ListBox, bao gồm: - *One*: Chỉ chọn một giá trị - *MultiSimple*: Cho phép chọn nhiều, chọn bằng cách Click vào mục chọn, bỏ chọn bằng cách Click vào mục đã chọn - *MultiExtended*: Chọn nhiều bằng cách nhấn kết hợp với Shift hoặc Ctrl
SelectedItems	Được sử dụng khi SelectionMode là MultiSimple hoặc MultiExtended. Thộc tính SelectedItems chứa các chỉ số của các dòng dữ liệu được chọn
SelectedItems	Được sử dụng khi SelectionMode là MultiSimple hoặc MultiExtended. Thuộc tính SelectedItems chứa các chỉ số của các dòng dữ liệu được chọn

Trong ListBox có một sự kiện được sử dụng rất nhiều đó chính là ***SelectedIndexChanged***, sự kiện này xảy ra khi thay đổi mục chọn trong ListBox.

- **ComboBox:**
  - + Thuộc tính: Items, DropDownStyle, SelectedIndex, SelectedItems, Text.
  - + Sự kiện: SelectedIndexChanged.

Một số thao tác với ComboBox:

- **Add()**: Thêm một mục chọn vào cuối danh sách ListBox.
- **Insert()**: Chèn thêm mục chọn vào vị trí **i.**
- **Count**: Trả về số muc chọn hiện đang có.
- **Item()**: Trả về mục chọn ở vị trí thứ **i.**
- **Remove()**: Bỏ mục chọn.
- **RemoveAt()**: Bỏ mục chọn ở vị trí thứ **i.**
- **Contains()**: Trả về True nếu có mục chọn trong danh sách, trả vể False nếu không có mục chọn trong danh sách.
- **Clear**: Xóa tất cả các mục chọn.
- **IndexOf()**: Trả về vị trí mục chọn trong danh sách, nếu không tìm thấy sẽ trả về -1.

Một số thuộc tính thường dùng:

Thuộc tính	**Mô tả**
Text	Trả về nội dung dòng dữ liệu đang hiển thị trên ComboBox
DropDownStyle	Quy định định dạng của ComboBox, nhận một tring các giá trị: - Simple: hiển thị theo dạng ListBox + TextBox có thể chọn dữ liệu từ ListBox hoặc nhập mới vào TextBox - DropDownList: Chỉ cho phép chọn dữ liệu trong ComboBox - DropDown: Giá trị mặc định, có thể chọn hoặc nhập mới mục dữ liệu vào ComboBox
Items	Trả về các mục chứa trong ComboBox
DropDownHeight	Thiết lập chiều cao tối đa khi sổ xuống của ComboBox
DropDownWidth	Thiết lập độ rộng của mục chọn trong ComboBox
SelectedIndex	Lấu chỉ số mục được chọn, chỉ số mục đầu tiên là 0
SelectedItem	Trả về mục được chọn
SelectedText	Lấy chuỗi hiển thị của mục chọn trên ComboBox
DataSource	Chọn tập dữ liệu điền vào ComboBox. Tập dữ liệu có thể là mảng, chuỗi, ArrayList,...
DisplayMember	Gán dữ liệu thành viên sẽ hiển thị trên ComboBox
ValueMember	Thuộc tính này chỉ định dữ liệu thành viên sẽ cung cấp giá trị cho ComboBox
SelectedValue	Trả về giá trị của mục chọn (ValueMember) nếu ComboBox có liên kết dữ liệu. Nếu không liên kết dữ liệu hoặc ValueMember không được thiết lập thì giá trị SelectedValue là giá trị chuỗi của thuộc tính

Thuộc tính	Mô tả
	SelectedItem

Trong **ComboBox** có một sự kiện là ***SelectedIndexChanged***, sự kiện này xảy ra khi thay đổi mục chọn trong ComboBox.

- **TreeView:**
  + Thuộc tính: Nodes, SelectedNode, ShowRootLine, ImageList, ShowLine, ImageIndex, SelectedImageIndex, FirstNode, LastNode, NextNode, PrevNode, Text.
  + Sự kiện: AfterSelect, Click
  + Phương thức: CollapseAll, ExpandAll.

Một số thuộc tính thường dùng của TreeView:

Thuộc tính	Mô tả
Node	Trả về một đối tượng thuộc lớp TreeNode
SelectedNode	Trả về Node đang được chọn trong TreeView
ShowPlusMinus	Hiển thị dấu + và - trước mỗi TreeNode
ShowRootLines	Hiển thị đường thẳng nối giữa các RootNode trong một TreeView
ImageList	Hiển thị hình trước mỗi Node trong TreeView. ***Lưu ý:** phải sử dụng thêm điều khiển *ImageList* và gán tên đối tượng của điều khiển *ImageList* cho thuộc tính *ImageList* của TreeView
ImageIndex	Giá trị của thuộc tính *ImageIndex* là chỉ số của hình trong điều khiển *ImageList*. Khi gán chỉ số cho thuộc tính *ImageIndex* thì hình hiển thị trước mỗi Node sẽ là hình có chỉ số tương ứng. ***Lưu ý:** Phải sử dụng thuộc tính *ImageList* trước
SelectedImageIndex	Giá trị của thuộc tính *SelectedImageIndex* là chỉ số của hình trong điều khiển *ImageList*. Khi người dùng chọn Node nào thì Node đó sẽ có hình tương ứng như thuộc tính *SelectedImageIndex* chỉ định

Một số phương thức thường dùng của TreeView:

Phương thức	Mô tả
GetNodeCount()	Đến số Node trong một TreeView
ExpandAll()	Hiển thị tất cả các Node trên TreeView
CollapseAll()	Thu gọn tất cả các Node trên TreeView
GetNodeAt(x, y)	Lấy một Node tại một vị trí có tọa độ *(x, y)* trên màn hình. ***Lưu ý**: Thương sử dụng sự kiện MouseDown hoặc NodeMouseClick

Một số sự kiện thường dùng của TreeView:

Sự kiện	Mô tả
AfterCollapse	Phát sinh khi thu gọn một TreeNode
AfterExpand	Phát sinh khi hiển thị các Node trong TreeNode
AfterSelect	Phát sinh khi chọn một TreeNode
NodeMouseClick	Phát sinh khi chọn một Node

Thuộc tính và phương thức của TreeNode

Một số thuộc tính thương dùng của TreeNode:

Thuộc tính	Mô tả
Nodes	Trả về tập các Node
Text	Đọc/gán chuỗi ký tự người dùng sẽ nhìn thấy ở mỗi Node
FirstNode	Trả về Node đầu tiên
LastNode	Trả về Node cuối cùng

Thuộc tính	Mô tả
NextNode	Chuyển đến Node tiếp theo
PrevNode	Lùi lại Node trước đó
Parent	Trả về Node cha của Node hiện tại
Index	Trả về chỉ số của Node

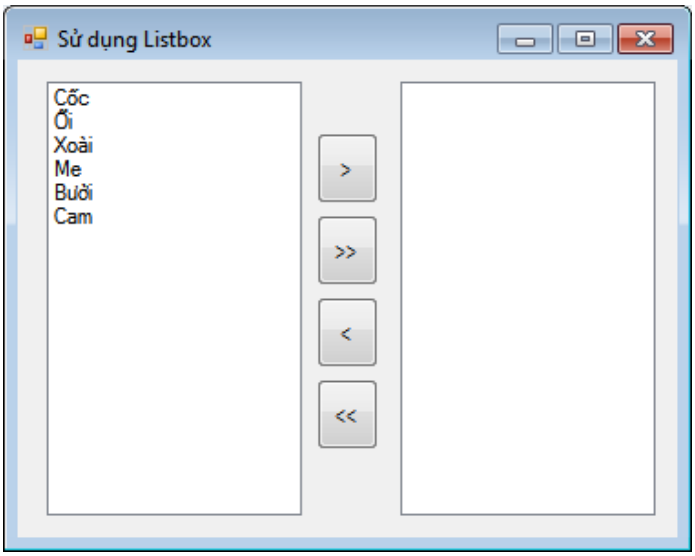
Một số phương thức thường dùng của TreeNode:

Phương thức	Mô tả
Nodes.Add	Thêm một Node
Nodes.Remove	Xóa một Node
Nodes.Insert	Chèn vào một Node
Nodes.Clear	Xóa tất cả các Node con và Node hiện tại

1.2 Giới thiệu bài tập mẫu

Bài 1: Thiết kế giao diện sau với các yêu cầu:

- Button >: Chuyển phần tử đang chọn qua listbox phải.
- Button >>: Chuyển tất cả phần tử qua listbox phải.
- Button <: Chuyển phần tử đang chọn qua listbox trái.
- Button <<: Chuyển tất cả phần tử qua listbox trái.
- Thêm button “Chuyển tùy ý” cho listbox trái: chọn item nào thì chuyển item đó qua listbox phải.
- Đóng Form có xác nhận từ người dùng.



Hướng dẫn:

– **Danh sách các đối tượng được sử dụng trong form:**

Object	Properties	Events
frmListBox	Name: frmListBox Text: "Sử dụng listbox" FontName: Tahoma FontSize: 11	FormClosing
lst_Trai	Name: lst_Trai	
lst_Phai	Name: lst_Phai	

– **Các sự kiện tương ứng với các yêu cầu:**

- Chuyển một item từ listbox trái sang listbox phải:

```
private void btnQuaPhai_Click(object sender, EventArgs e)
{
     // Chuyển giá trị chọn bên trái sang phải
     lst_Phai.Items.Add(lst_Trai.SelectedItem);
     // Xóa giá trị chọn bên trái
     lst_Trai.Items.Remove(lst_Trai.SelectedItem);
}
```

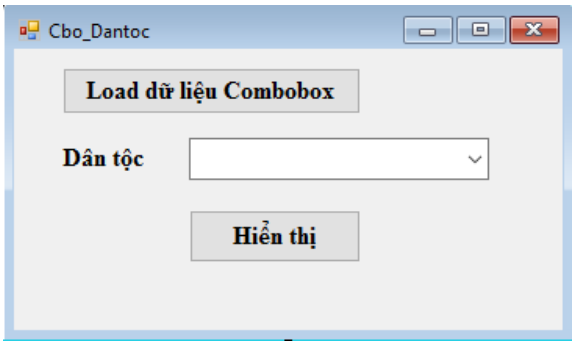
- Chuyển tất cả item từ listbox trái sang listbox phải

```
private void btnQuaPhaiAll_Click(object sender, EventArgs e)
{
```

```
        // Chuyển giá trị items bên trái sang phải
        lst_Phai.Items.AddRange(lst_Trai.Items);
        // Xóa giá trị bên trái
        lst_Trai.Items.Clear();
}
```

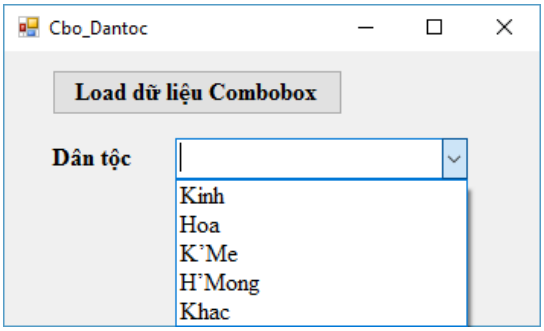
- Sinh viên tự viết code thực hiện cho các yêu cầu còn lại.

Bài 2: Thiết kế giao diện sau:



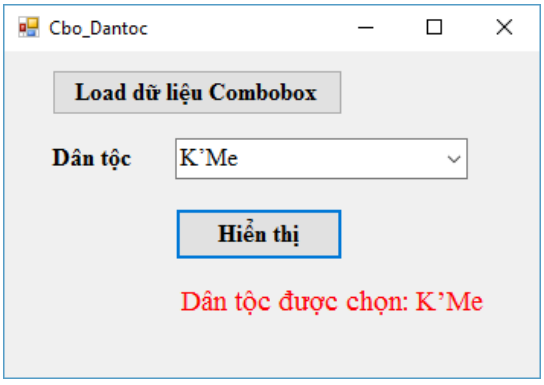
Yêu cầu:

– Click chọn button Load dữ liệu Combobox, 5 dân tộc: Kinh, Hoa, K‟Me, H‟Mong, Khác sẽ được đọa vào combobox Dân tộc.

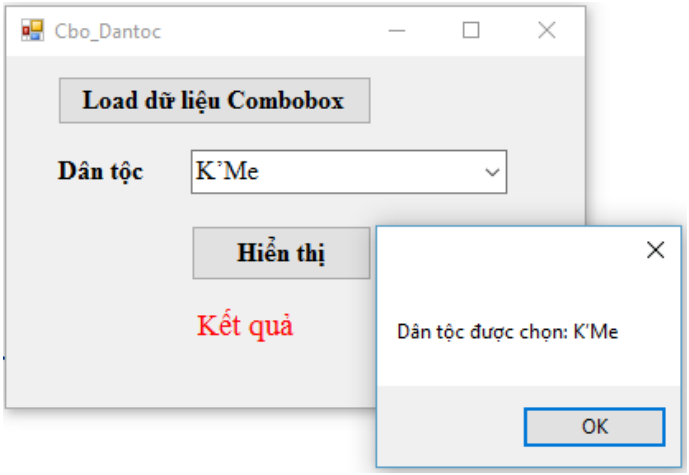


– Chọn một dân tộc trong combobox, hiển thị dân tộc được chọn theo 2 cách:

+ Cách 1: click button Hiển thị, dân tộc được chọn sẽ xuất hiện trong một Label hoặc Textbox.



+ Cách 2: Dân tộc được chọn sẽ xuất hiện trên MessageBox ngay khi chọn.



Hướng dẫn:

- Sự kiện click của button Load dữ liệu Combobox

```
private void btn_loadDL_Click(object sender, EventArgs e)
{
     string[] dt = { "Kinh", "Hoa", "K'Me", "H'Mong", "Khac" };
     foreach (string s in dt)
     {
         cbo_dt.Items.Add(s);
     }
}
```

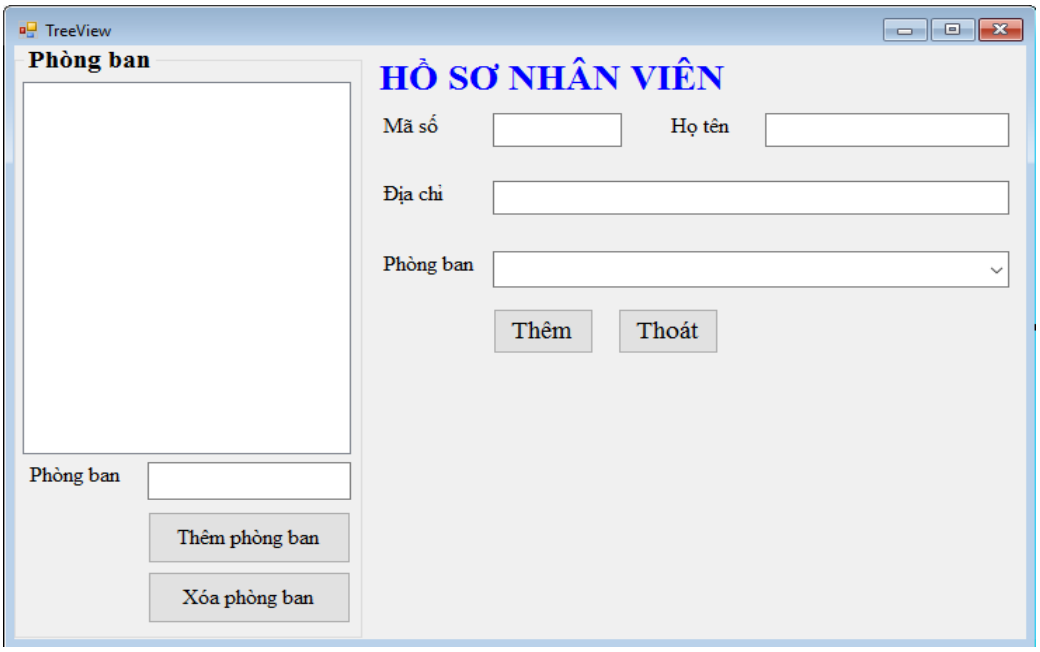
- Sự kiện click của button Hiển thị

```
private void btn_hienthi_Click(object sender, EventArgs e)
{
      string s = "Dân tộc được chọn: ";
      if (cbo_dt.SelectedIndex >= 0)
          lbl_kq.Text = s + cbo_dt.SelectedItem.ToString();
      else
           lbl_kq.Text = "Bạn chưa chọn dân tộc";
}
```

- Sự kiện SelectedIndexChanged của Combobox

```
private void cbo_dt_SelectedIndexChanged(object sender,EventArgs e)
{
  MessageBox.Show("Dân tộc được chọn:" +
                                    cbo_dt.SelectedItem.ToString());
}
```

Bài 3: Thiết kế giao diện sau:



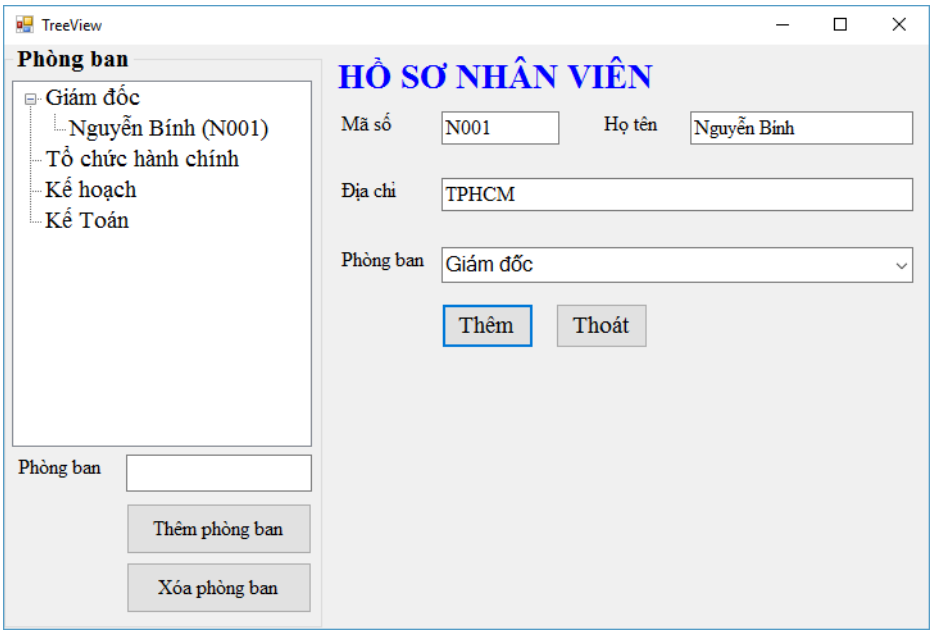
Yêu cầu:

– Khi Form hiển thị lên, TreeView phòng ban và ComboBox phòng ban có sẵn những phòng ban sau: Giám đốc, Tổ chức hành chính, Kế hoạch, Kế Toán.

– Thực hiện thêm mới phòng ban và xóa phòng ban được chọn (Lưu ý: kiểm tra trùng khi thêm và có xác nhận khi xóa).

– Khi một phòng ban mới được thêm thì phải thêm phòng đó vào ComboBox phòng ban.

– Thực hiện thêm mới nhân viên vào phòng ban được chọn lên TreeView.



Hướng dẫn:

– **Danh sách các thuộc tính của các Object:**

Object	Properties	Events
Form	Name: frmTreeView Text: “Phòng ban –Nhân viên” FontName: Times New Roman FontSize: 11	FormClosing
Label		
TextBox	Name: txt_phongban, txt_maso, txt_hoten, txt_diachi	
Combobox	Name: cbo_phongban	
Button	Name: btn_ThemPB, btn_XoaPB, btn_ThemNV, btn_Thoat	Click
TreeView	Name: trv_DS	

– **Xử lý sự kiện theo yêu cầu:**

- **Form Load**

```
private void TreeView_Load(object sender, EventArgs e)
{
     string[] pb = {"Giám đốc","Tổ chức hành chính","Kế hoạch",
                                                    "Kế Toán" };
     foreach (string s in pb)
     {
         trv_DS.Nodes.Add(s); //thêm node vào treeview
         cbo_phongban.Items.Add(s); //thêm item vào combobox
     }
     cbo_phongban.SelectedIndex = 0; //item đầu tiên trên được chọn
}
```

- **Thêm phòng ban mới vào treeview và combobox phòng ban**

```
private void btn_ThemPB_Click(object sender, EventArgs e)
{
      // Kiểm tra txt_Phongban có tồn tại trong TreeView chưa
      if (kiemtra(txt_Phongban.Text))
      {
```

```
            trv_DS.Nodes.Add(txt_Phongban.Text);
            cbo_phongban.Items.Add(txt_Phongban.Text);
        }
        else
            MessageBox.Show("Phòng ban đã tồn tại!");
        txt_Phongban.Text = "";
        txt_Phongban.Focus();
}
```

✍ **Lưu ý:** Sinh viên tự viết phương thức kiểm tra trùng phòng ban:

`bool kiemtra(string s)` và sử dụng phương thức so sánh chuỗi không phân biệt chữ hoa chữ thường:

```
string.Compare(string, string, bool)
```

- **Xóa phòng ban được chọn trong treeview**

```
private void btn_XoaPB_Click(object sender, EventArgs e)
{
      if(MessageBox.Show("Bạn có chắc muốn xóa?","Thông báo",
                  MessageBoxButtons.YesNo,MessageBoxIcon.Question,
                MessageBoxDefaultButton.Button1)==DialogResult.Yes)
      {
          // Một phòng ban trong treeView được chọn
          if (trv_DS.SelectedNode != null)
          {
              cbo_phongban.Items.Remove(trv_DS.SelectedNode.Text);
              trv_DS.Nodes.Remove(trv_DS.SelectedNode);
          }
       }
}
```

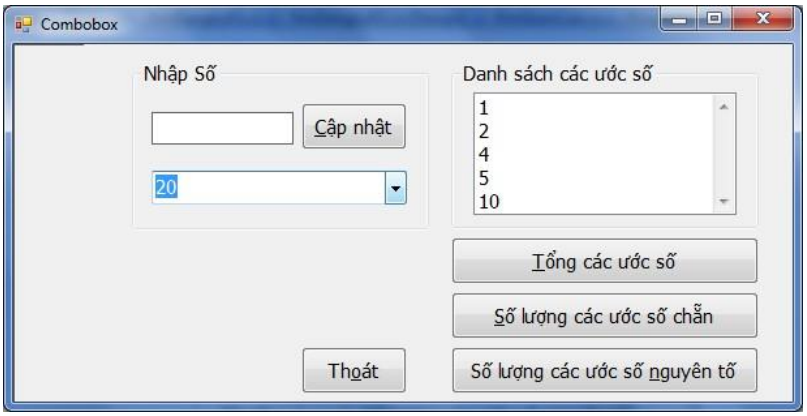
- **Thêm nhân viên của phòng ban vào treeview**

```
private void btn_ThemNV_Click(object sender, EventArgs e)
{
     //tìm index của node có nội dung là item được chọn trong
     //combobox phòng ban
     int index = -1;
     foreach (TreeNode node in trv_DS.Nodes)
     if (node.Text == cbo_phongban.Text)
     {
```

```
            index = node.Index;
            break;
        }
        trv_DS.Nodes[index].Nodes.Add(txt_hoten.Text+"
                                                ("+txt_maso.Text+")");
        trv_DS.ExpandAll(); //mở rộng treeview
}
```

2. Bài tập tại lớp

Bài 1. Thiết kế giao diện sau:



Yêu cầu:

– Nhập số vào textbox Nhập số (có kiểm tra dữ liệu nhập).

– btnCapNhat: thêm số vừa nhập vào combobox, đồng thời xóa dữ liệu trong textbox Nhập số, đặt con trỏ lại textbox (có kiểm tra số vừa nhập đã tồn tại trong combobox chưa trước khi thêm).

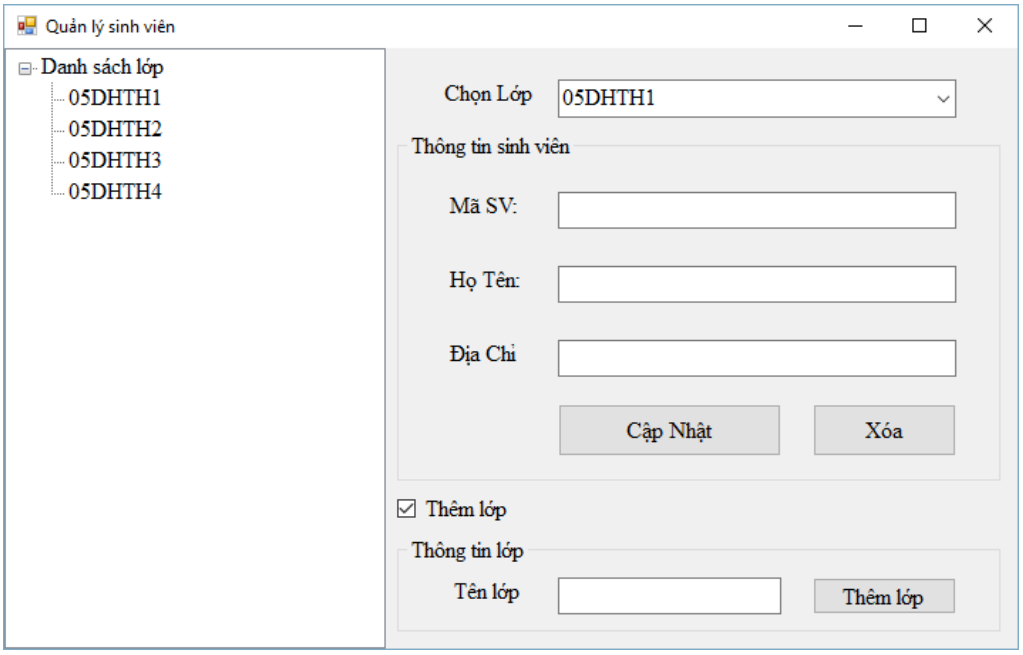
– Khi người dùng chọn một số trong combobox thì danh sách các ước số của số này sẽ hiển thị vào listbox bên phải tương ứng.

– Khi nhấn các button Tính: "Tổng các ước số", "Số lượng các ước số chẵn", "Số lượng các ước số nguyên tố" thì hiển thị thông tin tương ứng vào MessageBox dựa vào các ước số trên listbox.

– btnThoat: thoát khỏi ứng dụng có xác nhận từ người dùng.

– Tạo HotKey cho các button như trên giao diện

Bài 2. Thiết kế giao diện như sau:

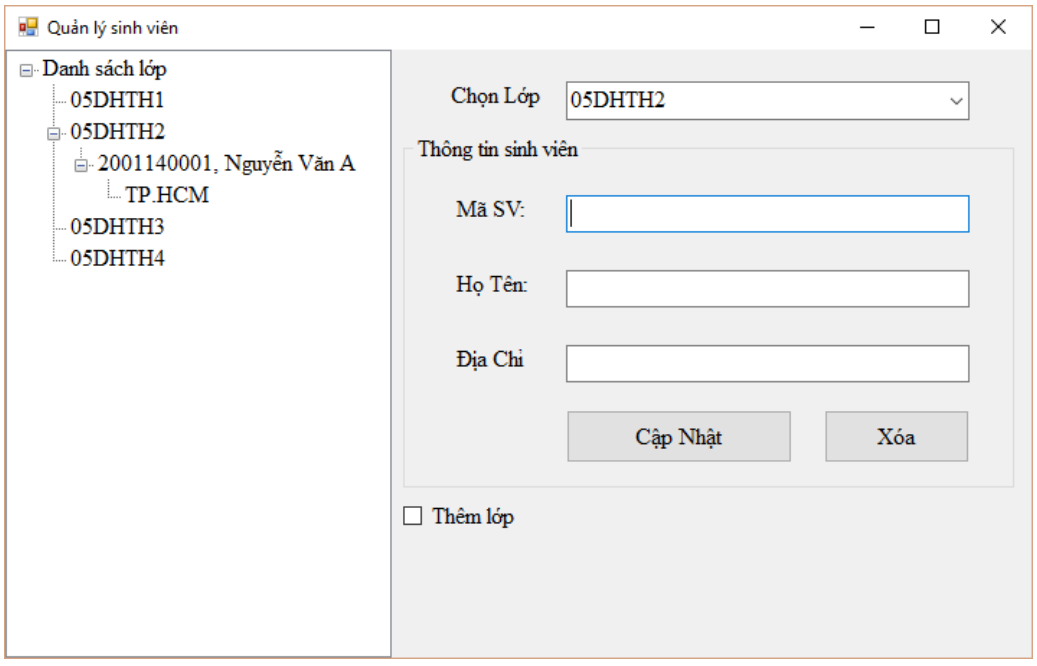


Yêu cầu:

– Khi Form hiện lên, danh sách lớp ở treeView đã tồn tại một số lớp; checkbox Thêm lớp là false (groupbox Thông tin lớp và các Control bên trong ẩn đi). Khi checkbox Thêm lớp là true groupBox Thông tin lớp và các control bên trong hiện lên.

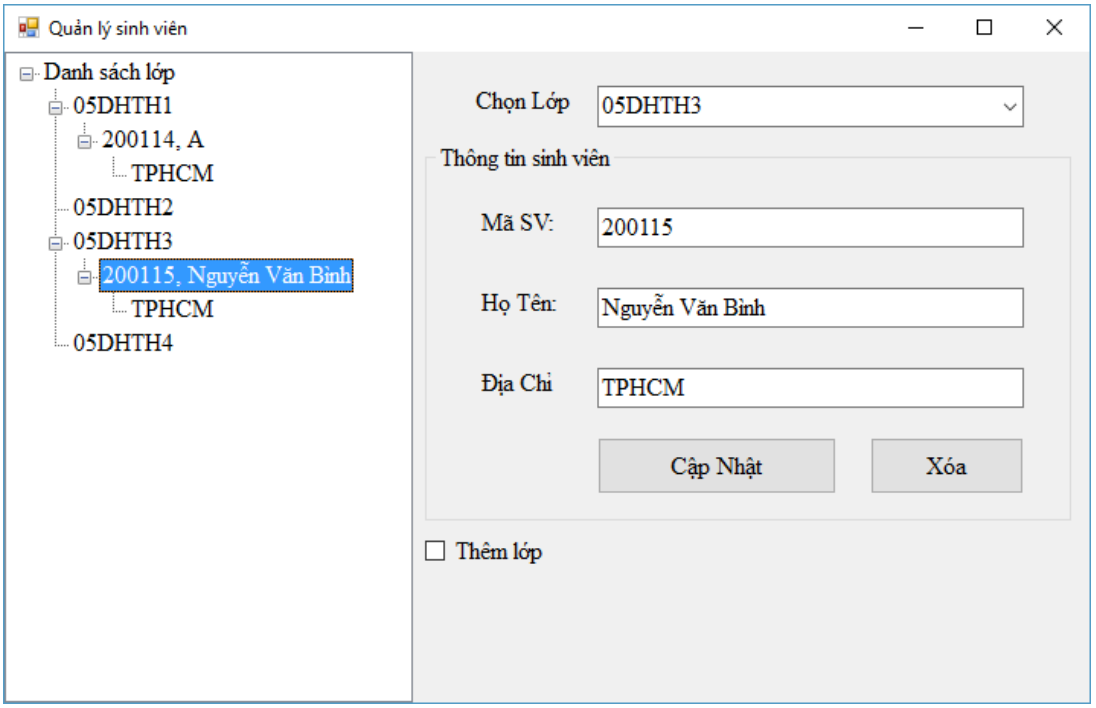
– Button Thêm lớp: Thực hiện thêm một lớp mới vào Danh sách lớp của TreeView (Kiểm tra tên lớp không được trùng trước khi thêm), đồng thời thêm lớp đó vào ComboBox chọn lớp.

– Button Cập Nhật: Thêm một sinh viên vào danh sách lớp đang chọn trên ComboBox với nội dung các node như hình. Trước khi thêm kiểm tra thông tin nhập bao gồm: các textbox nhập không được để trống, không trùng mã sinh viên.



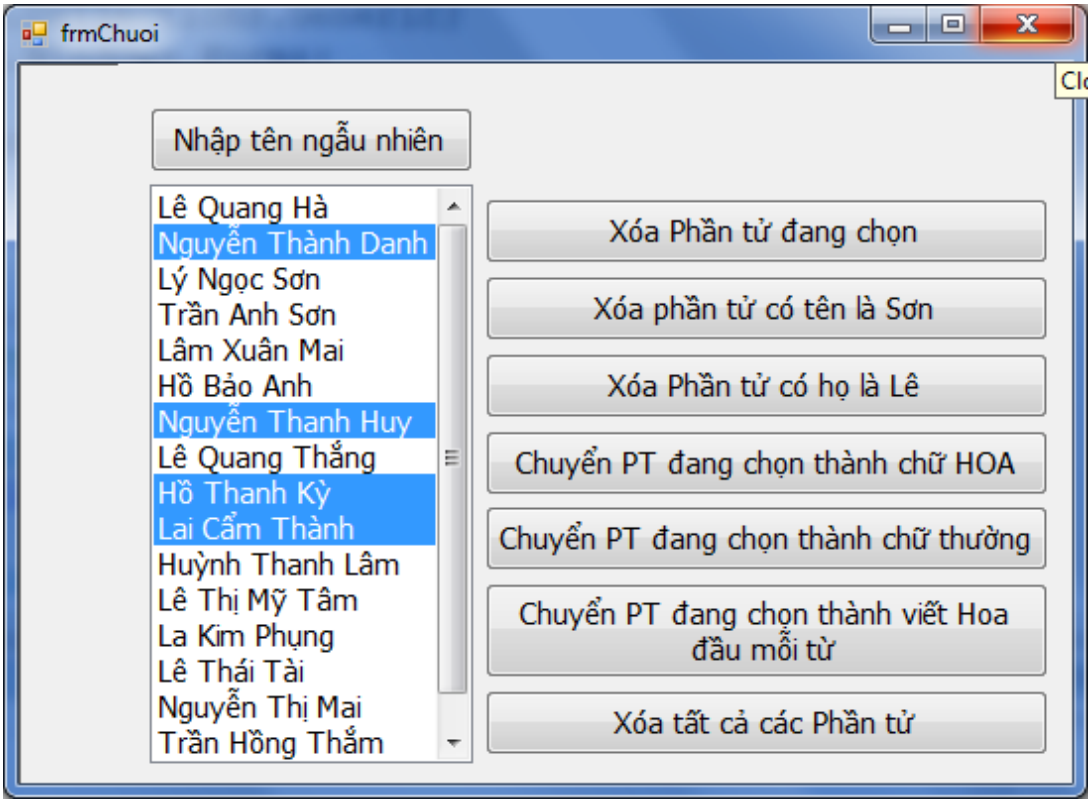
– Button Xóa: Chỉ cho phép xóa một node là sinh viên đang chọn trên TreeView (xác nhận trước khi xóa).

– Khi click chọn một node là mã sinh viên trong TreeView, hiển thị thông tin của sinh viên đó lên textbox tương ứng.



3. Bài tập nâng cao

Bài 1. Thực hiện form xử lý chuỗi có giao diện như sau:



Yêu cầu:

– btnNgauNhien: thực hiện thêm vào listbox 50 tên dựa vào mảng chuỗi HO,TENLOT, TEN được kết hợp ngẫu nhiên

HO = {“Lê”, “Nguyễn”, “Lý”, “Trần”, “Lâm”, “Hồ”, “Lai”, “Huỳnh”, “La”}

TENLOT = {“Quang”, “Thành”, “Ngọc”, “Anh”, Xuân”, “Bảo”, “Cẩm”, “Thị”,“Kim”, “Thái”, “Hồng”}

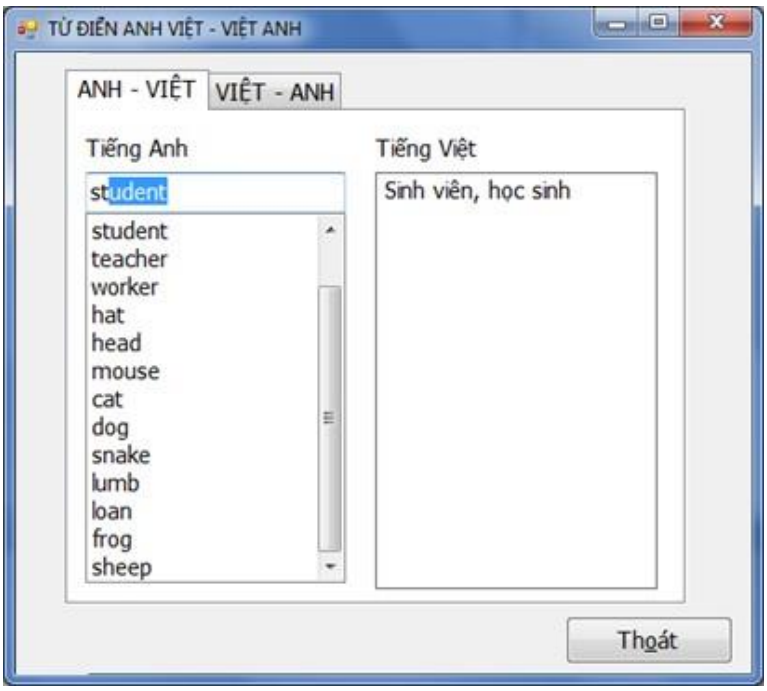
TEN = {“Hà”, “Danh”, “Sơn”, “Mai”, “Thắng”, “Kỳ”, “Thành”, “Lâm”, “Tâm”, “Phụng”, “Thắm”}

– Thực hiện các button tương ứng trên giao diện.

– Khi ngưới dùng double click vào tên đang chọn thì sẽ mở Inputbox, cho phépngười dùng thay đổi tên mới vào vị trí tên này.

4. Bài tập về nhà

Bài 1. Viết chương trình từ điển Anh – Việt và Việt – Anh



Mục đích:

Sử dụng

Arraylist,

Yêu cầu:

Xây dựng từ điển đơn giản như trên

– Khi nhập vào combobox từ cần tra thì chương trình sẽ dò tìm đến chữ nào khớpvới ký tự gần nhất.

– Khi nhấn button Enter hoặc double click vào từ cần tra thì nghĩa tương ứng của từ sẽ hiển thị vào textbox bên phải tương ứng.

– Danh sách các từ Lưu sẵn vào List

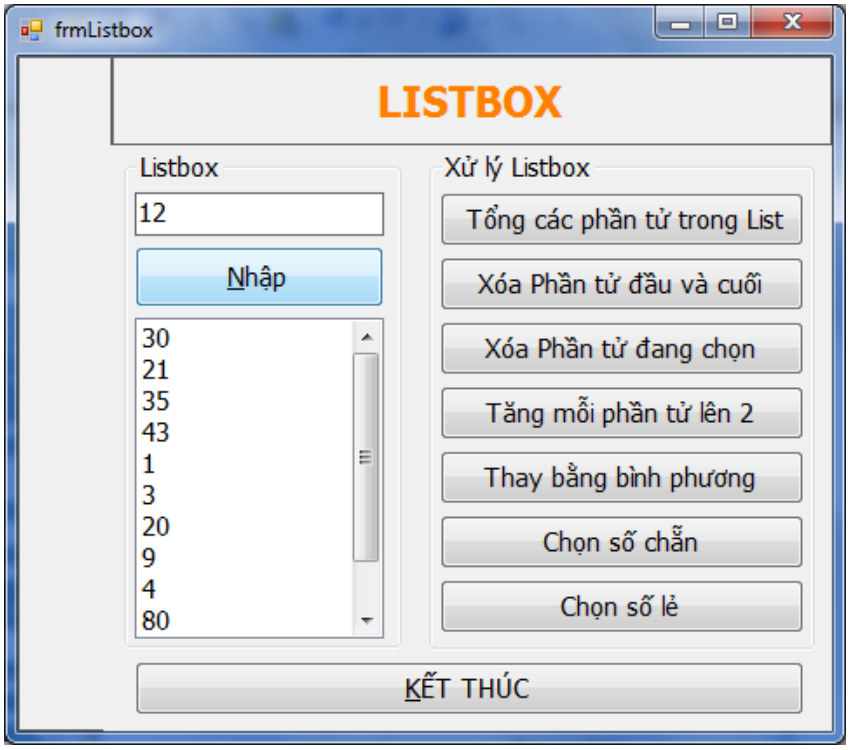
– Danh sách các từ lưu sẵn vào object (word) **Arraylist**

Hướng Dẫn:

– Bên trái là Combobox thể hiện dưới dạng Simple, chứa danh sách các từ cần tra cứu.

– Bên phải là TextBox thể hiện dưới dạng MultiLine, ghi nghĩa của các từ được chọn bên Combobox.

Bài 2. Viết chương trình thêm các phần tử vào listbox (listbox được chọn nhiều phầntử) các số tự nhiên N được nhập từ textbox.



Yêu cầu: các button lệnh thực hiện các công việc sau:

– Tính tổng các phần tử trong Listbox, hiển thị lên Messagebox.

– Xóa phần tử đầu và cuối của listbox.

– Xóa các phần tử đang chọn trong listbox.

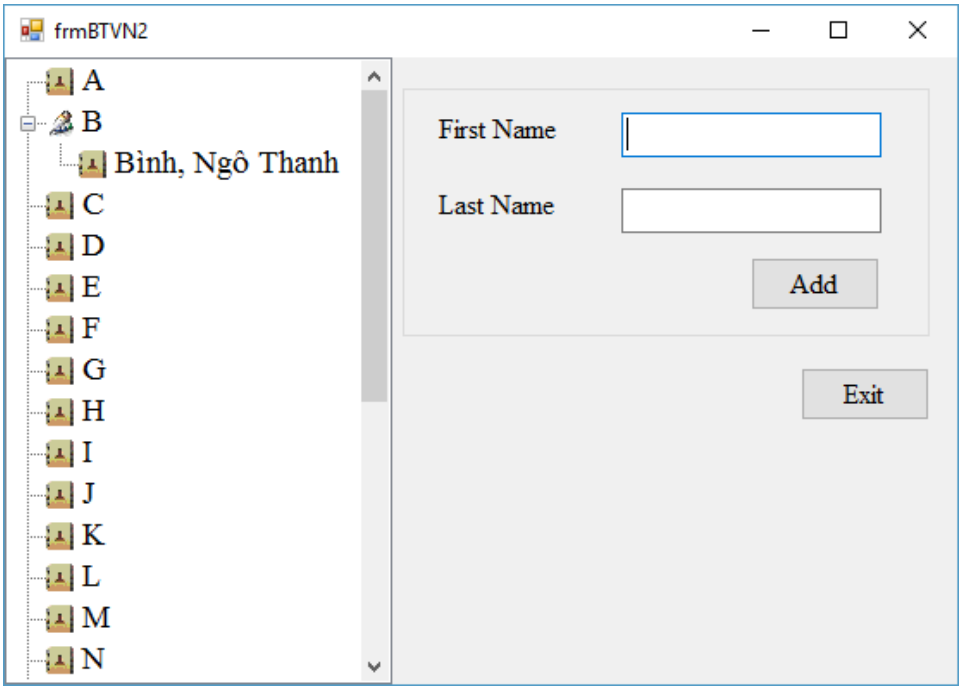
– Tăng giá trị mỗi phần tử lên 2.

– Thay mỗi giá trị của mỗi phần tử bằng bình phương của chính nó

– Thực hiện chọn các phần tử trong listbox là số chẵn.

– Thực hiện chọn các phần tử trong listbox là số lẻ.

Bài 3. Thiết kế giao diện nhập danh bạ với yêu cầu như sau:



Yêu cầu:

– Khi form được load lên treeView chứa tất cả các ký tự từ A→ Z.

– Nhằm mục đích tiện lợi cho người sử dụng, khi muốn thêm một người mới vào danh bạ, chương trình sẽ đưa người này vào treeView ở vị trí node tương ứng với chữ cái đầu của tên (xem hình).

----------Hết----------