BÀI TẬP VISUAL BASIC CĂN BẢN

Chương 1 THIẾT KẾ BIỂU MẪU DÙNG CÁC ĐIỀU KHIỂN

Mục tiêu:

Chương này gồm các bài tập nhằm rèn luyện cho sinh viên các thao tác cần thiết cho phép thiết kế các ứng dụng đơn giản trong môi trường lập trình Visual Basic cũng như một số kỹ năng lập trình cơ bản khi làm việc với Visual Basic.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

- Sử dụng các điều khiển để thiết kế giao diện trong Visual Basic.
- Vận dụng các cấu trúc lập trình trong Visual Basic để viết mã lệnh.
- Sử dụng một số cấu trúc dữ liệu trong Visual Basic.

Kiến thức có liên quan:

- Giáo trình “Visual Basic”; Chương 1, 2, 3, 4, 5.

Tài liệu tham khảo:

- **Visual Basic 6 Certification Exam Guide** - Chapter 1, Page 1; Chapter 2, Page 41; Chapter 4, Page 89 - **Dan Mezick & Scot Hillier** - **McGraw-Hill - 1998**.

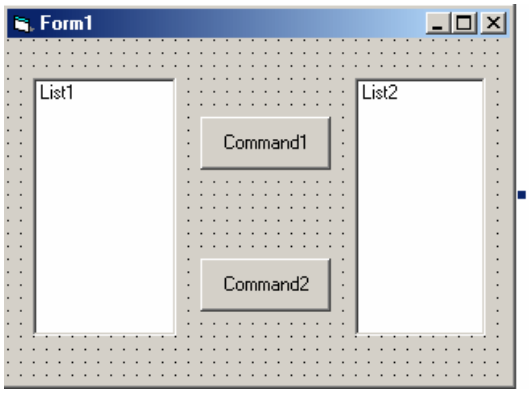
I. SỬ DỤNG MỘT SỐ ĐIỀU KHIỂN

I.1 Bài tập có hướng dẫn

Bài tập 1I-1
THAO TÁC TRÊN LISTBOX

Bước 1: Tạo thư mục Basic\Bt1-1. Tạo một dự án mới kiểu Standard EXE, lưu vào trong thư mục trên.
Bước 2: Thêm 2 List Box và một Button vào form (hình 1). Nhấn đúp lên form để mở ra cửa sổ Code, nhập các đoạn mã sau trong sự kiện Form_Load:

```
Form1.List1.AddItem "Thing 3"
Form1.List1.AddItem "Thing 2"
Form1.List1.AddItem "Thing 1"
```



Hình 1.1: Thao tác với List Box

Bước 3: Chạy ứng dụng bằng cách chọn Run/Start. List1 hiển thị 3 phần tử vừa thêm vào ở bước 2. Chấm dứt chương trình bằng cách chọn Run/End trên menu để trở về môi trường soạn thảo.
Bước 4: Nhấp đúp lên Button Command1 để hiển thị sự kiện Click của Command1.
Bước 5: Mục đích của Command1 là chuyển những phần tử được chọn từ List1 sang List2. Thêm đoạn mã sau vào thủ tục sự kiện Click của Command1:

```
' Kiem tra neu mot phan tu duoc chon
If Form1.List1.ListIndex = -1 Then Exit Sub
' Chep cac phan tu duoc chon tu List1 sang List2
Form1.List2.AddItem Form1.List1.List(Form1.List1.ListIndex)
```

Bước 6: Chạy ứng dụng. Nhấp phần tử thứ nhất của List1, sau đó nhấp Command1. Điều gì xảy ra? Phần tử được chọn của List1 phải được hiển thị bên List2. Chấm dứt ứng dụng và trở về môi trường soạn thảo.
Bước 7: Tìm trong phần trợ giúp các thuộc tính sau của ListBox:

- o ListCount
- o List

- ListIndex

Bước 8: Tìm trong phần trợ giúp các hàm sau của ListBox:
- AddItem
- RemoveItem
- Clear

Bước 9: Tìm trợ giúp cho lệnh VB:

Exit Sub

Bước 10: Đoạn mã trong thủ tục Command1_Click thực hiện thao tác chép phần tử từ một ListBox sang một ListBox khác. Bây giờ ta làm ngược lại: loại bỏ phần tử trong List1. Để làm điều này ta nhấp đúp lên Command1 và thêm dòng code sau vào cuối thủ tục:

```
' Xoa  phan tu duoc chon trong List1
Form1.List1.RemoveItem Form1.List1.ListIndex
```

Bước 11: Chạy chương trình và chọn phần tử thứ nhất trong List1. Điều gì xảy ra?

Bước 12: Nếu không chọn phần tử nào trong List1, nhấp Command1. Điều gì xảy ra? Tại sao?

Bước 13: Ta đã có một button dùng để chuyển các phần tử được lựa chọn từ trái sang phải (List1 sang List2), với button còn lại ta sẽ dùng để chuyển các phần tử được chọn từ phải sang trái (List2 sang List1).

Bước 14: Với Command2 ta sẽ copy đoạn mã từ Command1 với 1 vài thay đổi nhỏ.

Bước 15: Command2 thực hiện các thao tác giống với Command1, nhưng có nhiệm vụ di chuyển phần tử được lựa chọn từ List2 sang List1. Đoạn mã trong Command1 sẽ được sử dụng lại với một vài thay đổi nhỏ. Nhấp đúp lên Command1, chọn các mã lệnh đã thêm vào ở các bước trước. Chọn Edit/Copy trên menu.

Bước 16: Đóng cửa sổ Code và nhấp đúp lên Command2. Sự kiện Command2_Click sẽ hiển thị trong cửa sổ Code. Nhấp bất kỳ bên trong thủ tục sự kiện và chọn Edit/Paste trên menu. Như vậy ta đã chép đoạn mã từ Command1 sang Command2.

Bước 17: Sửa lại các mã lệnh vừa được chép. Thay đổi các chú thích cho thích hợp; đổi List1 thành List2 và ngược lại. Những sửa đổi này giúp Command2 có thể thực hiện thao tác chuyển các phần tử được chọn từ List2 sang List1.

Lưu các công việc đã thực hiện bằng cách chọn File/Save Project.

Bước 18: Chạy chương trình. Chọn phần tử thứ nhất trong List1 và chọn Command1 để chuyển nó sang List2. Bây giờ chọn phần tử thứ nhất trong List2, và nhấp Command2. Nếu Command2 không thực thi, trở lại môi trường soạn thảo. Kiểm tra lại đoạn mã lệnh trong thủ tục Command2_Click ta vừa chép ở bước trên.

Bước 19: Lưu ý rằng các phần tử ở cả 2 ListBox không được sắp thứ tự; nếu muốn sắp thứ tự, ta nhấp List1 và đổi thuộc tính Sorted thành True, tương tự đối với List2.

Bước 20: Lưu dự án lại và chạy chương trình. Tất cả các phần tử phải được hiển thị theo thứ tự trong cả 2 ListBox, bất chấp thứ tự chúng được thêm vào trong ListBox.

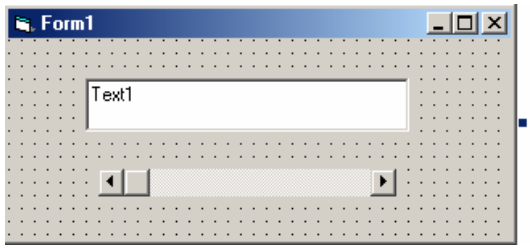
Bài tập 1I-2

THAO TÁC VỚI SPINCONTROL

Một SpinControl là sự kết hợp của TextBox và Slider. Slider tạo một miền giá trị số được hiển thị trong TextBox. Các giá trị này có thể được thay đổi bằng cách nhập trực tiếp vào trong TextBox.

Bước 1: Tạo thư mục Basic\Bt1-2. Tạo dự án mới trong thư mục trên.

Bước 2: Trong Form1, thêm một TextBox và Horizontal Scroll Bar như hình 2. Thiết lập các thuộc tính sau cho mỗi Control:



Hình I.2: Spin Control

Item1: TextBox
Name: Text1
Text: <blank>
Item2: Horizontal Scroll Bar
Name: Hscroll1
LargeChange: 10
Max: 100

Bước 3: Nhấp đúp lên scrollbar để nhập mã lệnh, đây là sự kiện Change của Scroll Bar gọi là hàm HScroll1_Change. Thêm đoạn mã sau để hiển thị giá trị hiện thời của scroll bar trong TextBox.

```
Text1.Text = HScroll1.Value
```

Bước 4: Chạy ứng dụng bằng cách chọn Run/Start trên menu. Bây giờ nhấp các mũi tên trái và phải của scroll bar. Giá trị trong TextBox phải thay đổi.

Bước 5: Bây giờ thêm mã để thay đổi giá trị bằng cách nhập trực tiếp giá trị trong TextBox. Nhấp đúp vào TextBox và thêm đoạn mã sau để thiết lập giá trị cho scroll bar khi TextBox thay đổi:

```
HScroll1.Value = Text1.Text
```

Bước 6: Chạy chương trình và nhập 50 vào TextBox. Vạch của scroll bar thay đổi theo. Thay đổi vạch của scroll bar, giá trị trong TextBox cũng thay đổi.

Bước 7: Trong khi chạy chương trình, nhập ký tự A vào TextBox. Điều gì xảy ra? Nguyên nhân vì scroll bar chỉ nhận các giá trị là số chứ không phải ký tự.

Bước 8: Để ngăn chặn những ký tự không mong muốn được nhập vào TextBox, ta sử dụng sự kiện KeyPress. Sự kiện này xảy ra khi có một phím trên bàn phím được nhấn, nhưng trước khi giá trị thực sự được hiển thị trên TextBox. Sự kiện này nhận một giá trị số nguyên của phím được nhấn, gọi là ASCII. Mỗi ký tự trên bàn phím được đại diện bằng một mã ASCII duy nhất. Do đó ta có thể kiểm tra phím nào được nhấn và bỏ qua nó nếu ta thấy không cần thiết.

Bước 9: Thêm đoạn mã sau vào sự kiện Text1_KeyPress để ngăn chặn các giá trị không phải là số.

```
' Loai bo ky tu khong can thiet
```

```
If KeyAscii = vbKeyBack Then Exit Sub
If KeyAscii < vbKey0 Or KeyAscii > vbKey9 Then
        KeyAscii = 0
End If
```

Bước 10: Lưu dự án lại và chạy chương trình.

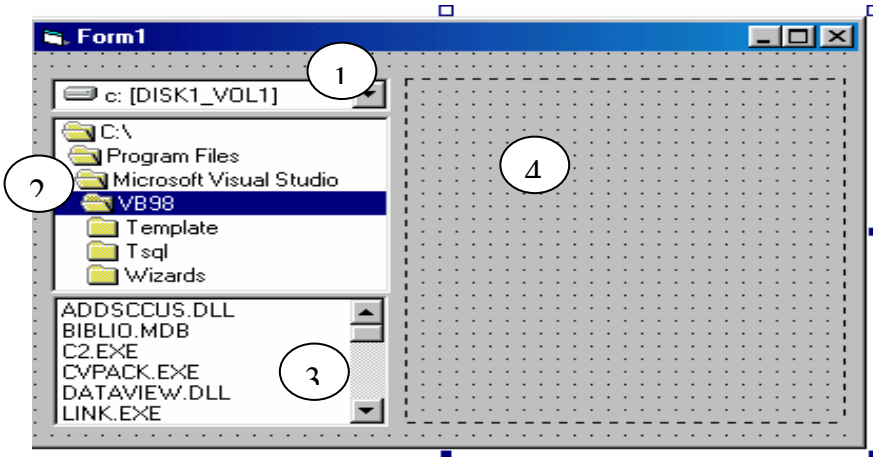
Bài tập 1I-3

THAO TÁC VỚI DRIVELISTBOX, DIRLISTBOX, FILELISTBOX

Trong ví dụ này ta phải tạo 5 đối tượng, trong đó có 4 điều khiển:

- Một Form.
- Một điều khiển DriveListBox
- Một điều khiển DirListBox
- Một điều khiển FileListBox
- Một điều khiển ImageBox

Bước 1: Tạo giao diện người dùng. Ta chỉ cần nhấp và vẽ đúng vị trí từng điều khiển trên Form.



Hình I.3: Giao diện lựa chọn tập tin hình ảnh để hiển thị

1: **DriveListBox**
Name: drvSource

2: **DirListBox**
Name: dirSource

3: **FileListBox**
Name: filSource
Pattern: *.bmp;*.wmf;*.ico;*.jpg

4: **ImageBox**
Name: ImgSource
Stretch: TRUE

Bước 2: Viết mã trao đổi thông tin giữa các đối tượng:

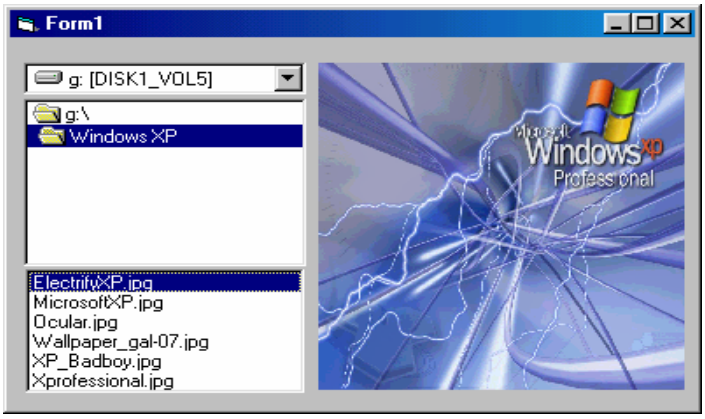
Trong cửa sổ thiết kế Form, nhấp đúp vào DriveListBox, cửa sổ Code hiện ra, xử lý sự kiện sau:

```
Private Sub drvSource_Change()
   dirSource.Path = drvSource.Drive
End Sub
```

Tương tự cho DirListBox & FileListBox

```
Private Sub dirSource_Change()
   filSource.Path = dirSource.Path
End Sub
Private Sub filSource_Click()
        imgSource.Picture = LoadPicture(filSource.Path & "\" & filSource.FileName)
End Sub
```

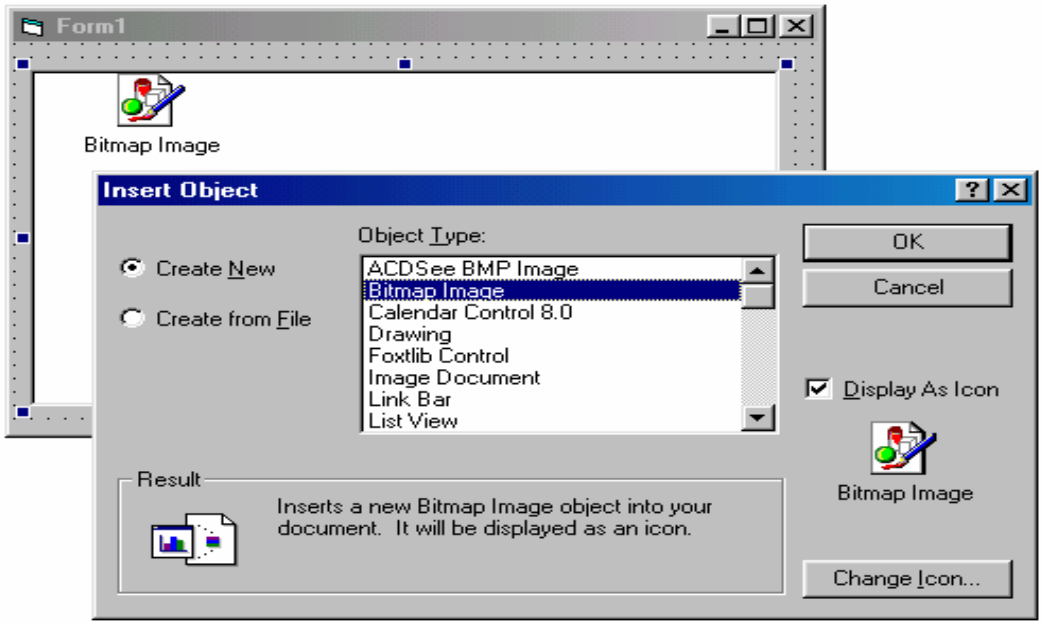
Bước 3: Lưu dự án lại vào thư mục Basic\Bt1-3. Chạy chương trình nhờ phím F5.



Hình I.4: Kết quả thực thi

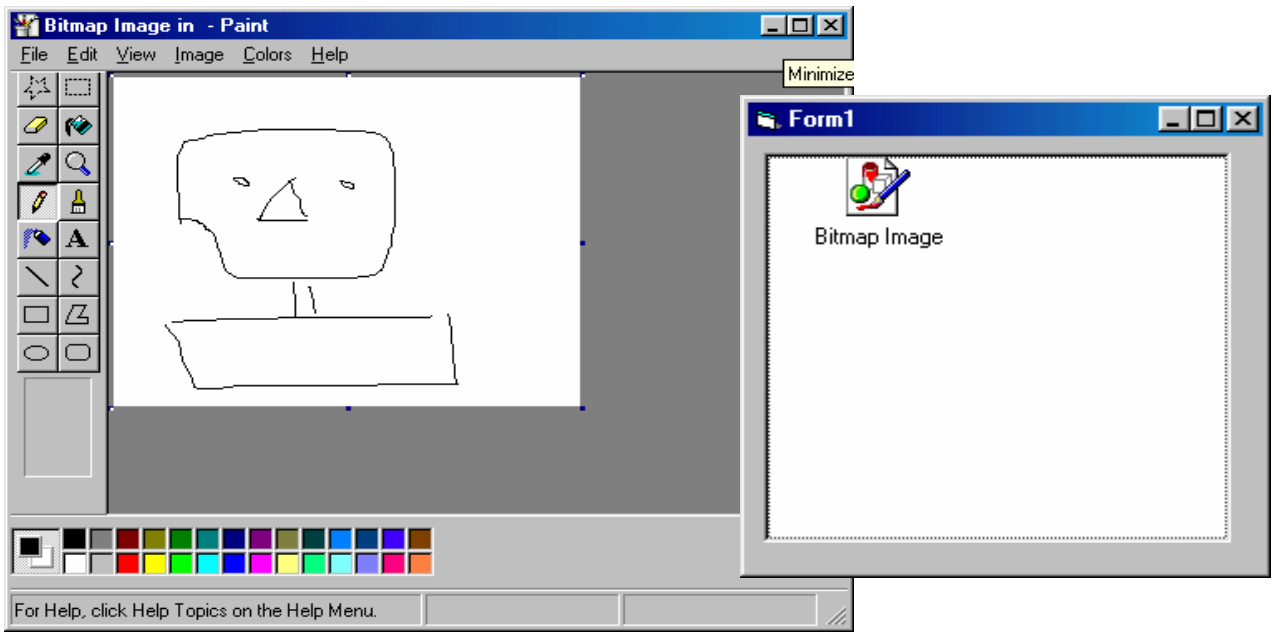
Bước 1: Tạo dự án mới, trong đó ta có sử dụng OLE.

Hộp thoại **Insert Object** hiện ra để ta lựa chọn, ở đây chọn kích hoạt **Create New**, **Object Type** là **Bitmap Image**; đánh dấu chọn **Display as Icon**.



Hình I.5: Sử dụng OLE Control

Bước 2: Nhấp OK, VB sẽ gọi trình ứng dụng Paint & ta vẽ hình trên cửa sổ Paint. Sau đó chọn **Exit & Return** trong cửa sổ Form, ta được:

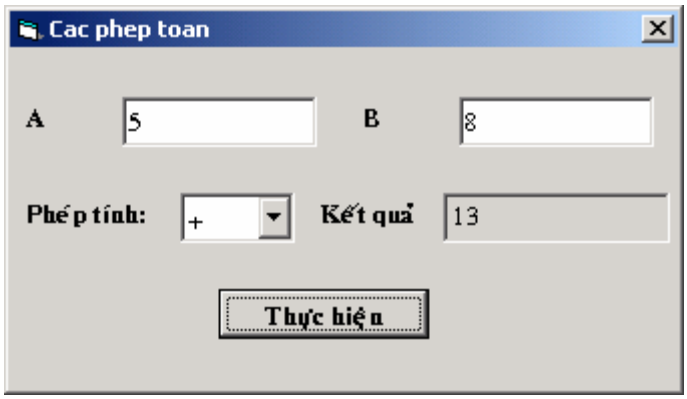


Hình I.6: Kết quả thực thi ứng dụng

Bước 3: Lưu dự án vào thư mục Basic\Bt1I-4 và chạy chương trình; nhấp đúp vào biểu tượng Bitmap Image,VB sẽ khởi động Paint để ta hiệu chỉnh hình vẽ đầu.

I.2 Bài tập tự làm

1) Thiết kế chương trình như sau:



Hình I.7 Các phép tính cơ bản

Nhập vào 2 giá trị A, B; sau đó chọn một phép toán (+,-,*,/). Nhấp chọn nút nhấn **Thực hiện**, kết quả sẽ hiển thị trong điều khiển nhãn **Kết quả**.

2) Thiết kế chương trình để nhập vào tọa độ của hai điểm (x1,y1); (x2,y2) và cho phép:

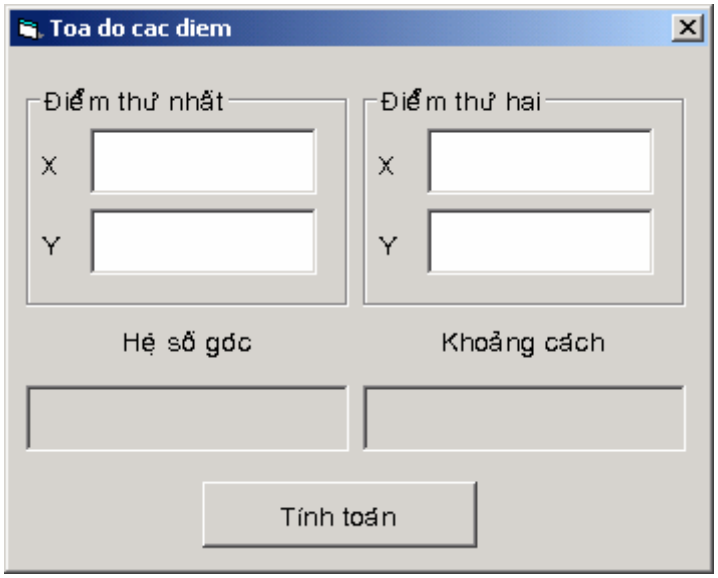
a) Tính hệ số góc của đường thẳng đi qua hai điểm đó theo công thức:

Hệ số góc = (y2 - y1) /(x2 - x1)

b) Tính khoảng cách giữa hai điểm theo công thức:

khoảng cách = $\sqrt{(x2-x1)^2+(y2-y1)^2}$

Giao diện chương trình có thể như sau:



Hình I.8: Tọa độ các điểm

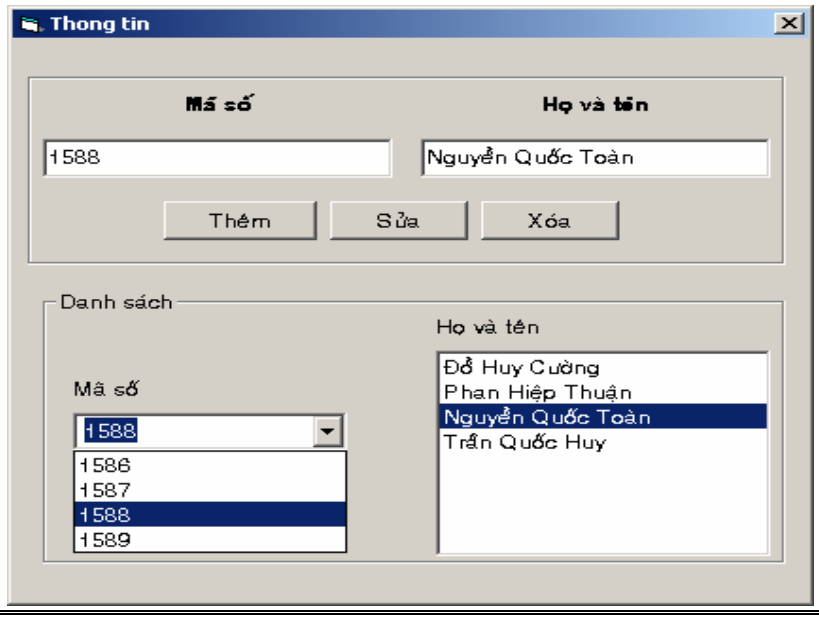
3) Viết chương trình cho phép nhập vào bán kính r của một hình tròn. Tính chu vi và diện tích của hình tròn theo công thức :

Chu vi CV = 2*Pi*r

Diện tích Dt = Pi*r*r

Hiển thị các kết quả lên màn hình.

4) Thiết kế chương trình có giao diện như hình dưới và thực hiện các chức năng sau:



Hình I.9: Lựa chọn tên

- Mỗi khi người sử dụng chương trình nhập thông tin vào 2 ô TextBox, sau đó nhấp chọn nút **Thêm**, giá trị của ô **Mã số** được đưa vào ComboBox, còn giá trị của ô **Họ và tên** được đưa vào ListBox.
- Mỗi khi họ chọn một **mã số** nào đó trong ComboBox, giá trị **họ và tên** tương ứng cũng sẽ được chọn trong ListBox; đồng thời chúng sẽ được hiển thị lên trên các điều khiển TextBox tương ứng (như hình). (*Xử lý sự kiện Combo1_Click & List1_Click*)
- Đối với **mã số** và **họ tên** của một người, ta có thể sửa đổi giá trị của chúng trong các ô nhập TextBox, sau đó chọn nút **Sửa**, giá trị của chúng trong ComboBox & ListBox cũng sửa đổi theo.
- Khi người dùng *chọn một mã số* (hay họ tên) trên ComboBox (hoặc ListBox), sau đó họ chọn **Xóa**, các thông tin này được xóa ra khỏi ComboBox & ListBox.

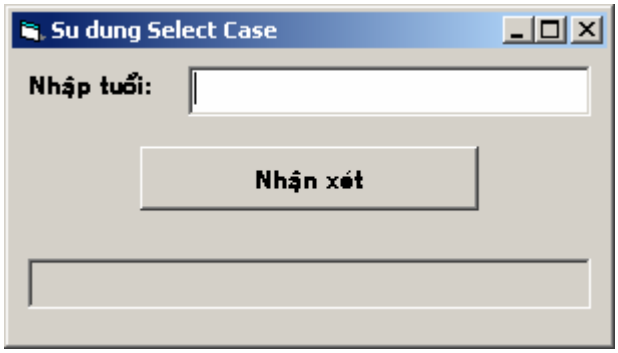
II. CÁC CẤU TRÚC LẬP TRÌNH TRONG VB

II.1 Bài tập có hướng dẫn

Bài tập 1II-1

SỬ DỤNG SELECT CASE

Tạo thư mục Basic\Bt1II-1. Thiết kế chương trình có giao diện & lưu trong thư mục trên:



Hình I.10: Select Case

Ở đây, người sử dụng chương trình nhập vào một tuổi nào đó trong ô nhập tuổi, sau đó họ nhấp nút **Nhận xét**, một nhận xét sẽ xuất hiện ứng với tuổi mà họ nhập từ bàn phím.

Lúc này ta sử dụng toán tử so sánh (=, <, <=, >, >=, <>) cùng với các từ khóa **Is** và **To** trong biểu thức.

Is: so sánh biến với biểu thức được liệt kê sau từ khóa **Is**.

To: định nghĩa phạm vi của giá trị.

Sự kiện Command1_Click():

```
Dim Age As Integer
Age = Val(Text1.Text)
```

```
     Select Case Age
          Case Is < 18
             Label2.Caption = "Ban con thieu nien, ban phai hoc
thoi!"
          Case 18 To 30
             Label2.Caption = "Ban da truong thanh, lap gia dinh
thoi!"
          Case 31 To 60
             Label2.Caption = "Lua tuoi trung nien roi!"
          Case Else
             Label2.Caption = "Ban co con chau day dan roi
nhe!"
     End Select
```

Bài tập 1II-2

BIẾN VÀ CẤU TRÚC

Bước 1: Tạo thư mục Basic\Bt1II-2. Tạo dự án mới (VB Standard EXE) trong thư mục trên; thêm một modul vào dự án, trong modul nay thêm vào đoạn mã sau:

```
Public Const tieude As String = "Quan ly hanh chinh"
Public Const sohieu As String = "1.0"
```

Thêm đoạn mã sau vào hàm xử lý sự kiện Form_Load của Form1:

```
Form1.Caption = tieude & " phien ban " & sohieu
```

Chạy ứng dụng, ta thấy tiêu đề của Form: "Quan ly hanh chinh phien ban 1.0".

Bây giờ, mở Modul1 và thay Public bằng Private. Chạy chương trình. Điều gì xảy ra?

Bước 2: Đổi các khai báo trên thành Public, thêm dòng sau đây vào đầu thủ tục Form_Load:

```
tieude = "Loi xuat hien"  & "Hang so khong the thay doi duoc."
```

Chạy chương trình, điều gì xảy ra?

Bước 3: Thêm dòng sau trong hàm xử lý sự kiện Form_Resize:

```
MsgBox "FORM RESIZE"
```

Bước 4: Chạy chương trình, khi Form bắt đầu được hiển thị (sự kiện Form_Load), sự kiện Resize của Form được thực hiện. Chỉ có hàm xử lý sự kiện Resize mới cho biết chắc rằng hàm Form_Load được thực thi. Để kiểm chứng ta tạo một biến trên form và trong hàm Form_Load ta thiết lập giá trị của nó. Sau đó, hàm Form_Resize có thể kiểm tra biến và xử lý trên biến này.

Bước 5: Khai báo một biến Private trong Form1 tên *sukienLoad*:

```
Private sukienLoad As Boolean
```

Trong hàm Form_Load, đặt giá trị True cho biến trên:

```
sukienLoad = True
```

Bây giờ ta kiểm tra giá trị của biến trong hàm Form_Resize. Thêm vào đoạn mã sau trong hàm Form_Resize:

```
If sukienLoad = True Then
      SukienLoad =False
      Exit Sub
End If
MsgBox "Form Resize"
```

Chạy ứng dụng, khi Form bắt đầu được hiển thị, ta không thấy xuất hiện câu thông báo, nhưng khi ta thay đổi kích thước của Form (nhấn các nút _, ☐ của form), câu thông báo lại xuất hiện. Ở đây ta đã sử dụng một biến làm trung gian cho sự giao tiếp giữa sự kiện Form_Load và sự kiện Form_Resize. Bởi vì cả 2 hàm này nằm trong Form1, nên ta có khai báo Private cho chúng, các ứng dụng khác không thể truy xuất đến các biến này.

CHƯƠNG TRÌNH CON

Bước 6: Ta viết một chương trình con để xử lý chuỗi. Đầu vào của chương trình con là một chuỗi, kết quả của chương trình con là chuỗi đó nhưng các từ đều được viết hoa ký tự đầu tiên. Bài tập này giúp ta khai báo (định nghĩa) một chương trình con và gọi thực thi chương trình con đó trong chương trình ứng dụng của mình.

Chọn Modul1 trong cửa sổ soạn thảo chương trình, sau đó nhấp chọn Tools\Add Procedure. Định nghĩa một hàm public tên Doihoa() như sau:

```
Public Function Doihoa(s As String) As String
        Dim s1 As String
        Dim s2 As String
        Do While InStr(s, " ") <> 0
        s1 = Left(s, InStr(s, " "))
        s = Right(s, Len(s) - InStr(s, " "))
        ' Doi chu hoa
        s1 = UCase(Left(s1, 1)) & Right(s1, Len(s1) - 1)
        s2 = s2 & s1
        Loop

        ' Tra ket qua
        Doihoa = s2 & " " & s
End Function
 ' Ham nay khong viet hoa tu cuoi cung.
```

Bước 7: Hàm Doihoa có nhiệm vụ nhận vào một chuỗi và đổi ký tự đầu tiên của các từ trong chuỗi thành chữ hoa. Bây giờ ta kiểm tra hàm này như sau: Thêm một TextBox và một nút nhấn (Button) lên Form1. Nhấp vào Button, ta thêm đoạn mã sau vào hàm xử lý sự kiện Command1_Click:

```
        Form1.Caption = Doihoa(Text1.Text)
```

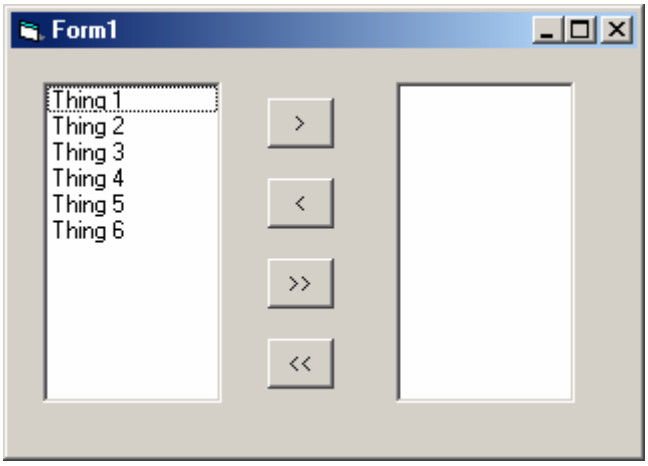
Chạy ứng dụng, nhập một chuỗi vào Text1, nhấp Command1. Chuỗi chữ hoa sẽ xuất hiện trên tiêu đề của Form1.

Bước 8: Sửa lại sao cho có thể viết hoa ký tự đầu tiên của tất các từ.

Bài tập 1II-3

LỰA CHỌN VỚI LISTBOX

Bước 1: Tạo thư mục Basic\Bt1II-3. Tạo dự án mới VB Standard EXE trong thư mục trên, sau đó tạo Form có dạng sau:



Hình I.11: Lựa chọn với ListBox

Ta có 2 ListBox và các nút nhấn (Button); trong đó:

Nút > chuyển một phần tử từ trái sang phải

Nút < chuyển một phần tử từ phải sang trái.

Nút >> chuyển tất cả các phần tử từ trái sang phải.

Nút << chuyển tất cả các phần tử từ phải sang trái.

Thêm 2 ListBox và 4 Button vào Form1. Trong hàm xử lý sự kiện Form_Load thêm vào đoạn mã:

```
List1.AddItem "Thing 1"
List1.AddItem "Thing 2"
List1.AddItem "Thing 3"
List1.AddItem "Thing 4"
List1.AddItem "Thing 5"
List1.AddItem "Thing 6"
```

Chạy chương trình.

Bước 2: Thêm hàm xử lý sự kiện Click cho nút nhấn 1 (>) Command1_Click:

```
' Kiem tra co chon hay khong?
If Form1.List1.ListIndex = -1 Then Exit Sub
 ' Chep tu trai sang phai
Form1.List2.AddItem Form1.List1.List(Form1.List1.ListIndex)
 ' Xoa ben trai
Form1.List1.RemoveItem Form1.List1.ListIndex
```

Bước 3: Chạy chương trình, chọn phần tử trong List1 và nhấp nút >, phần tử đó chuyển sang List2. Bây giờ ta làm ngược lại: chuyển phần tử được chọn từ List2 sang List1. Trở về cửa sổ soạn thảo; chọn đoạn mã vừa nhập trong List1, chọn Edit\Copy trong menu của VB. Nhấp lên Button <, chọn Edit\Paste. Bây giờ ta sửa lại đoạn mã sau trong hàm xử lý sự kiện Command2_Click:

```
' Kiem tra co chon hay khong?
If Form1.List2.ListIndex = -1 Then Exit Sub
```

```
' Chep tu phai sang trai
Form1.List1.AddItem Form1.List2.List(Form1.List2.ListIndex)
' Xoa ben phai
Form1.List2.RemoveItem Form1.List2.ListIndex
```

Bước 4: Lưu dự án và chạy chương trình.

Ta nhận thấy 2 đoạn mã lệnh trên (cho Button < và >) là như nhau (chỉ đổi chỗ List1 cho List2 và ngược lại). Do đó ta sẽ viết một chương trình con để chuyển dữ liệu từ ListBox này sang ListBox kia, và trong hàm xử lý sự kiện của 2 Button ta chỉ cần gọi chương trình con này để chuyển dữ liệu.

Thêm một Modul mới vào dự án tên Modul1, chọn Tool\Add Procedure để thêm một chương trình con vào tên Chuyendulieu()

Vào Modul1, sửa đổi lại thủ tục chuyển dữ liệu như sau:

```
Public Sub Chuyendulieu(L1 As ListBox, L2 As ListBox)
    ' Kiem tra co chon hay khong?
    If L1.ListIndex = -1 Then Exit Sub
    ' Chep
    L2.AddItem L1.List(L1.ListIndex)
    ' Xoa ben trai
    L1.RemoveItem L1.ListIndex
End Sub
```

Bây giờ hàm xử lý sự kiện của Command1 (Command1_Click) ta sửa lại như sau:

```
Call Chuyendulieu(Form1.List1, Form1.List2)
```

Hàm Command2_Click:

```
Call Chuyendulieu(Form1.List2, Form1.List1)
```

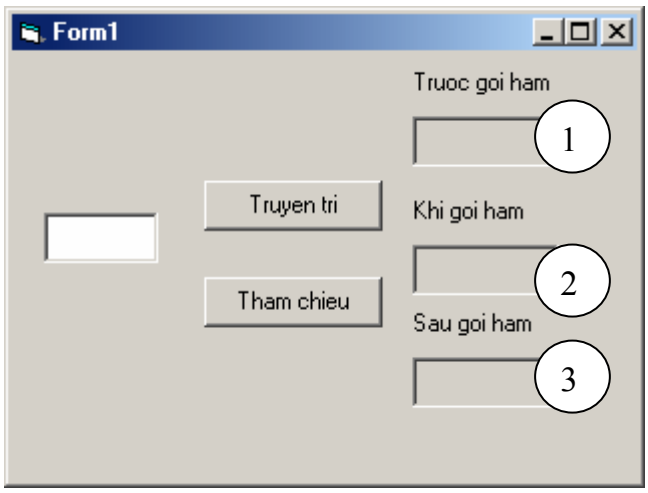
Lưu dự án và chạy chương trình. Kiểm tra kết quả.

Bài tập 1I -4

TRUYỀN THEO ĐỊA CHỈ VÀ TRUYỀN THEO GIÁ TRỊ

Bước 1: Tham số đưa vào chương trình con được truyền theo một trong 2 cách: theo địa chỉ và theo giá trị. Bây giờ ta tạo dự án mới trong thư mục Basic\Bt2-4 để kiểm tra chúng.

Bước 2: Tạo Form1 như sau:



Hình I.12: Truyền tham số

Button 1: Name: cmdTTri; Caption: Truyen tri
Button 2: Name: cmdTChieu; Caption: Tham chieu
TextBox: Name: Text1
Label 1: Name: lblTruoc
Label 2: Name: lblTrong
Label 3: Name: lblSau

Bước 3: Thêm 1 Modul vào dự án tên là Modul1, chọn Tools\Add Procedure thêm thủ tục Thamchieu như sau:

Name: Thamchieu
Type: Sub
Scope: Public

Bước 4: Thêm đoạn mã sau trong thủ tục Thamchieu

```
Public Sub Thamchieu(so As Integer)
        so = so + 2
        Form1.lblTrong.Caption = Str(so)
End Sub
```

Bước 5: Chọn Tool\Add Procedure để thêm thủ tục Truyentri như sau:

Name: Truyentri
Type: Sub
Scope: Public

Bước 6: Thêm đoạn mã sau trong thủ tục Truyentri

```
Public Sub Truyentri(ByVal so As Integer)
        so = so + 2
        Form1.lblTrong.Caption = Str(so)
End Sub
```

Bước 7: Sự khác nhau giữa 2 thủ tục trên là từ khóa **ByVal** trong thủ tục Truyentri. Bây giờ ta thêm thủ tục xử lý biến cố Click của Button cmdTTri. Thêm đoạn mã sau:

```
Private Sub cmdTTri_Click()
        Dim n As Integer
        n = Val(Text1.Text)
        lblTruoc.Caption = Str(n)
        Call Truyentri(n)
        lblSau.Caption = Str(n)
End Sub
```

Bước 9: Thêm hàm xử lý biến cố cmdTChieu_Click cho Button cmdTChieu:

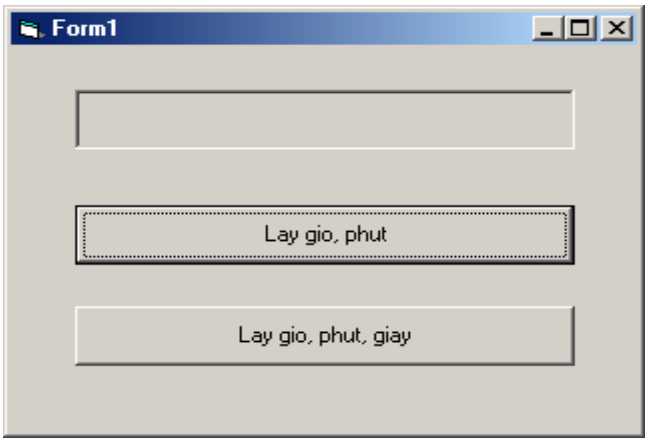
```
Private Sub cmdTChieu_Click()
        Dim n As Integer
        n = Val(Text1.Text)
        lblTruoc.Caption = Str(n)
        Call Thamchieu(n)
        lblSau.Caption = Str(n)
End Sub
```

Bước 10: Lưu dự án, chạy chương trình. Nhập số bất kỳ vào ô TextBox rồi nhấp nút Truyen tri, sau đó nhấp nút Tham chieu. Kiểm tra kết quả. Giải thích.

Bài tập 1II-5

THAM SỐ TÙY CHỌN

Bước 1: Tạo thư mục tên Basic\Bt1II-5. Tạo dự án mới trong thư mục này.
Bước 2: Tạo Form như sau:



Hình I.13: Lấy thời gian

Trong đó:
Label: Name: lblTg
Button 1: Name: cmdGioPhut
Button 2: Name: cmdGioPhutGiay

Bước 3: Thêm modul mới vào dự án tên Modul1. Chọn Tool\ Add Procedure tạo thủ tục:
Name: Laythoigian
Type: Sub
Scope: Public
Bước 3: Thêm đoạn mã sau vào thủ tục trên:

```
Public Sub Laythoigian(gio As String, phut As String, Optional giay As String)
        ' Tham so thu 3 co tu khoa Optional, nghia la  ta
        ' co the goi thu tuc co the co tham so nay hay khong co deu duoc
```

```
        ' Ham IsMissing kiem tra xem tham so nay co hay khong
        If IsMissing(giay) Then giay = ""
        Dim hientai
        hientai = Now
        gio = Format$(hientai, "hh")
        phut = Format$(hientai, "nn")
        giay = Format$(hientai, "ss")
End Sub
```

Bước 4: Thêm thủ tục xử lý sự kiện cho Button cmdGiophutgiay, trong thủ tục này chèn đoạn mã sau:

```
Private Sub cmdGiophutgiay_Click()
        Dim gioht As String
        Dim phutht As String
        Dim giayht As String
        Call Laythoigian(gioht, phutht, giayht)
        lblTg.Caption = gioht & ":" & phutht & ":" & giayht
End Sub
```

Bước 5: Thêm thủ tục xử lý sự kiên cho Button cmdGiophut, trong thủ tục này chèn đoạn mã sau:

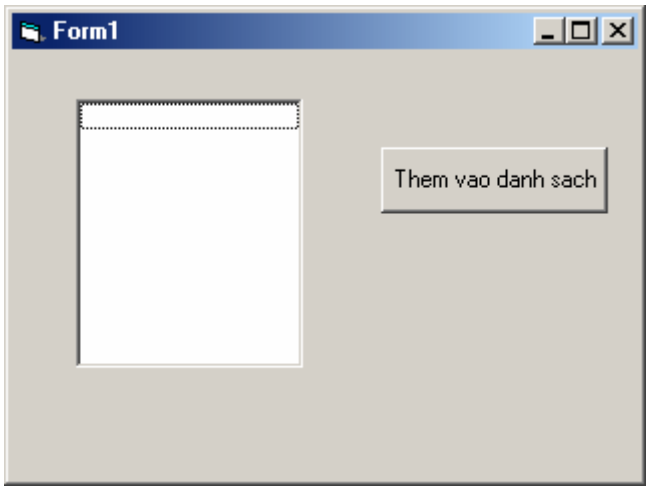
```
Private Sub cmdGiophut_Click()
        Dim gioht As String
        Dim phutht As String
        ' Khong su dung tham so thu ba
        Call Laythoigian(gioht, phutht)
        lblTg.Caption = gioht & ":" & phutht
End Sub
```

Bước 6: Lưu dự án lại và chạy chương trình. Kiểm tra kết quả.

Bài tập 1II-6

PARAM ARRAY

Bước 1: Tạo thư mục Basic\Bt1II-6. Tạo dự án mới trong thư mục này.
Bước 2: Tạo Form như hình sau:



Hình I.14: Param Array

Trong đó:

ListBox: Name: lstTen

Button: Name: cmdds; Caption: Them vao danh sach

Bước 3: Chèn modul mới vào dự án tên Modul1. Sau đó, chọn Tool\Add Procedure để chèn thủ tục sau:

Name: Diends

Type: Sub

Scope: Public

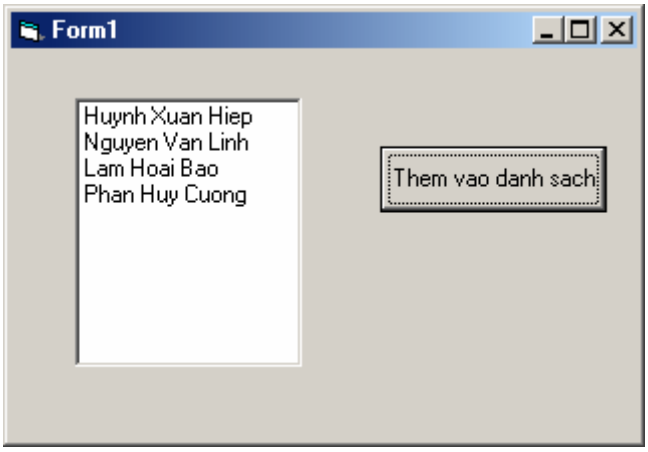
Bước 3: Chèn đoạn mã sau vào thủ tục Diends

```
Public Sub Diends(ParamArray Ten() As Variant)
    ' Su dung ParamArray thi mang phai kieu Variant va
    ' mang nay la tham so cuoi cung cua thu tuc
    Dim hten As Variant
    For Each hten In Ten()
    Form1.lstTen.AddItem hten
    Next
End Sub
```

Bước 4: ParamArray cho phép không cần xác định số lượng các đối số trong một chương trình con. Bây giờ, thêm hàm xử lý sự kiện cho nút cmdds: cmdds_Click:

```
Private Sub cmdds_Click()
  Call Diends("Huynh Xuan Hiep", "Nguyen Van Linh", "Lam Hoai Bao")
  Call Diends
  Call Diends("Phan Huy Cuong")
End Sub
```

Bước 5: Lưu dự án lại và chạy chương trình. Kiểm tra kết quả (hình bên dưới). Lưu ý đến lời gọi thủ tục trong sự kiện cmdds_Click (số lượng đối số khác nhau)

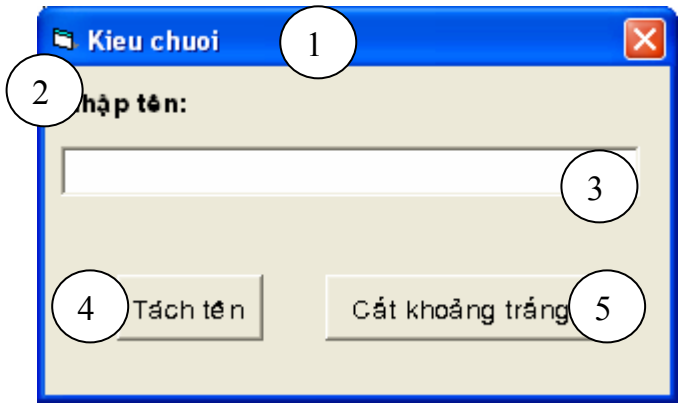


Hình I.15: Kết quả Param Array

Bài tập 1II-7

XỬ LÝ CHUỖI

Bước 1: Tạo dự án mới trong thư mục Basic\Bt2-7 với giao diện như sau:



Hình I.16: Xử lý chuỗi

1: Form: Name: frmMain; MinButton: False; MaxButton: False; Font: VNI-Times.
2: Label: Name: lblTen.
3: TextBox: Name: txtTen.
4: CommandButton: Name: cmdTen; Caption: Tách tên.
5: CommandButton: Name: cmdCKT; Caption: Cắt khoảng trắng.

Bước 2: Tạo một hàm cắt khoảng trắng như sau:

```
Private Function ATrim(ByVal Name As String) As String
   Name = LTrim(RTrim(Name))
   Do While InStr(Name, "␣␣") <> 0
      Name = Replace(Name, "␣␣", "␣")
   Loop
   ATrim = Name
End Function
```

Bước 3: Trong cửa sổ thiết kế Form; nhấp đúp vào *Tách tên*, ta xử lý đoạn mã cho sự kiện này:

```
Private Sub cmdTen_Click()
   Dim sName As String, Name As String
   sName = ATrim(StrConv(txtTen.Text, vbProperCase))
   Dim i As Long
   i = InStrRev(sName, "␣")
   Name = Right(sName, Len(sName) - i)
   MsgBox Name & ": " & Str(Len(Name))
End Sub
```

Bước 4: Sau đó, trở lại cửa sổ thiết kế, nhấp đúp vào *Cắt khoảng trắng*, ta xử lý:

```
Private Sub cmdCKT_Click()
   Dim sName As String
   sName = ATrim(StrConv(txtTen.Text, vbProperCase))
```

```
   MsgBox sName, , "Kieu du lieu chuoi"
End Sub
```

Bước 5: Lưu dự án và chạy chương trình.

Bài tập 1II-8

XỬ LÝ LỖI

Bước 1: Tạo một dự án mới. Dùng Tools\Add Procedure thêm một thủ tục mới tên GoiThuTuc vào Form1 với nội dung như sau:

```
Public Sub GoiThuTuc()
        Dim bien As Integer
  MsgBox "Truoc khi gan tri cho bien"
  bien = "Bien nguyen khong nhan gia tri la chuoi"
  MsgBox "Sau khi gan tri cho bien: " & "Bien = " & Format(bien)
End Sub
```

Bước 2: Thủ tục xử lý sự kiện Form_Load có nội dung như sau:

```
Private Sub Form_Load()
        MsgBox "Truoc khi goi thu tuc"
        Call GoiThuTuc
        MsgBox "Sau khi goi thu tuc"
End Sub
```

Lưu dự án vào thư mục Basic\Bt1II-8:
Form: tên là form1
Project: Debug

Bước 3: Chạy chương trình. VB đưa ra hộp thoại để bắt lỗi (debug) chương trình. Ta chọn End để trở về cửa sổ soạn thảo.

Tạo tập tin thực thi tên Debug.exe bằng cách chọn File\Make Debug.exe. Chạy tập tin Debug.exe từ Windows Explorer ta nhận được hộp thoại báo lỗi và chương trình tự động chấm dứt.

Nhận xét kết quả khi thực hiện chương trình.

Bước 4: Bây giờ ta thêm vào đoạn mã xử lý lỗi trong thủ tục của sự kiện Form_Load:

```
Private Sub Form_Load()
        On Error GoTo Xulyloi

        MsgBox "Truoc khi goi thu tuc"
        Call GoiThuTuc
        MsgBox "Sau khi goi thu tuc"

Thoat:
        Exit Sub
Xulyloi:
```

```
        MsgBox "Su kien Form_Load - Loi xay ra: " & Err.Description
        Resume Thoat
    End Sub
```

Bước 5: Lưu dự án và chạy chương trình. Nhận thấy, thay vì ta nhận được câu thông báo lỗi từ VB, một hộp thoại báo lỗi do ta đưa vào xuất hiện. Lưu ý, những lỗi được bắt trong thủ tục Form_Load (chứ không phải trong GoiThuTuc()). Nguyên nhân vì thủ tục GoiThuTuc() được gọi bởi thủ tục xử lý sự kiện Form_Load.

Bước 6: Biên dịch lại thành tập tin Debug.exe, chạy nó. Nhận xét kết quả.

Bước 7: Các kết quả trên cho ta biết được các lỗi trong sự kiện Form_Load được xử lý bởi các thao tác bắt lỗi trong thủ tục Form_Load. Nhưng nếu thủ tục GoiThuTuc() cũng có các thao tác bắt lỗi chương trình thì sao? Đơn giản giả sử một lỗi xuất hiện trong GoiThuTuc(). Bộ phận xử lý lỗi của GoiThuTuc (do ta thêm vào để bắt lỗi chương trình) sẽ thực thi thay vì đoạn lệnh bắt lỗi của sự kiện Form_Load được thực hiện. Khi GoiThuTuc chấm dứt, quyền xử lý lỗi mới trao lại cho sự kiện Form_Load.

Sửa lại thủ tục GoiThuTuc như sau:

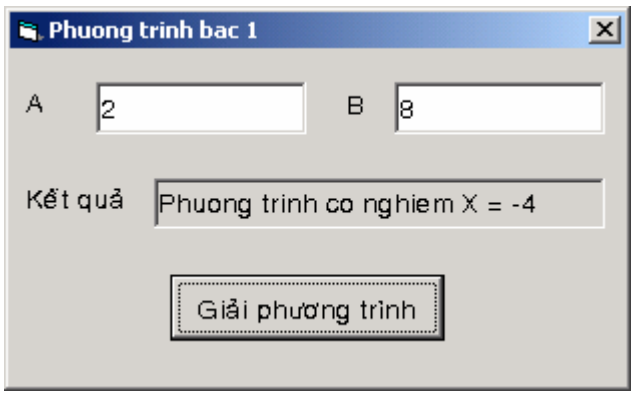
```
Public Sub GoiThuTuc()
        Dim bien As Integer
        On Error GoTo Xulyloicucbo
  MsgBox "Truoc khi gan tri cho bien"
  bien = "Bien nguyen khong nhan gia tri la chuoi"
  MsgBox "Sau khi gan tri cho bien: " & "Bien = " & Format(bien)
Thoatthutuc:
        Exit Sub
Xulyloicucbo:
        MsgBox "GoiThuTuc() - Loi xay ra: " & Err.Description
        Resume Thoatthutuc
End Sub
```

Bước 8: Lưu dự án và chạy chương trình. Thay đoạn mã Resume Thoatthutuc bằng Resume và chạy chương trình. Một vòng lặp vô tận xảy ra do chương trình sẽ quay lại đoạn mã bị lỗi và cố gắng thực thi nó; để thoát chương trình ta phải bấm tổ hợp phím Ctrl + Break.

Bây giờ thay Resume bằng Resume Next và chạy lại chương trình. Nhận xét kết quả. Giải thích.

II.2 Bài tập tự làm

1) Thiết kế chương trình cho phép nhập vào các hệ số a, b của phương trình bậc 1 dạng: ax+b=0; sau đó giải phương trình này. Giao diện chương trình có thể như sau:



Hình I.17: Phương trình bậc 1

2) Thiết kế chương trình cho phép nhập vào các hệ số a, b, c của phương trình bậc 2 dạng: $ax^2 + bx + c=0$; sau đó giải phương trình này.

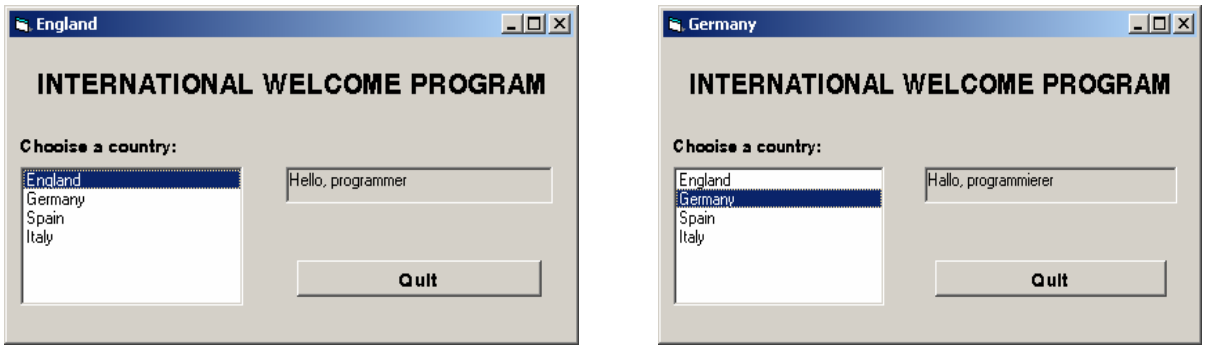
3) Thiết kế chương trình cho phép nhập vào một ký tự, sau đó kiểm tra xem ký tự đó thuộc tập hợp nào trong các tập ký tự sau:

Các ký tự chữ hoa: 'A' ...'Z'
Các ký tự chữ thường: 'a' ... 'z'
Các ký tự chữ số : '0' ... '9'
Các ký tự khác.

4) Giải phương trình bậc 1 bằng cách sử dụng cấu trúc Select Case

5) Tạo một chương trình hiển thị một danh sách chọn lựa cho người dùng trong một ListBox, sau đó xử lý với cấu trúc quyết định Select Case.

Mục đích của điều khiển sự kiện này là hiển thi một danh sách các quốc gia, sau đó hiển thị một thông điệp chào mừng bằng ngôn ngữ bản xứ khi người dùng chọn quốc gia của họ.



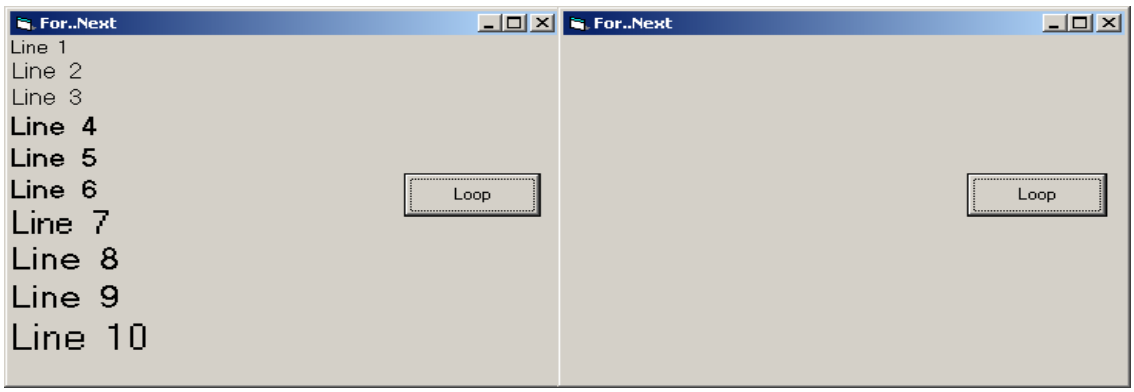
Hình I.18: Lời chào các nước

Chẳng hạn: Tiếng Anh: Hello, programmer
Tiếng Đức: Hallo, programmierer
Tiếng Tây Ban Nha: Hola, programador
Tiếng Ý: Ciao, programmatori

6) Sử dụng vòng lặp For.. Next

Sử dụng For.. Next để thay đổi độ lớn ký tự trên một Form bằng cách thay đổi thuộc tính FontSize của Form.

Thiết kế Form có giao diện:



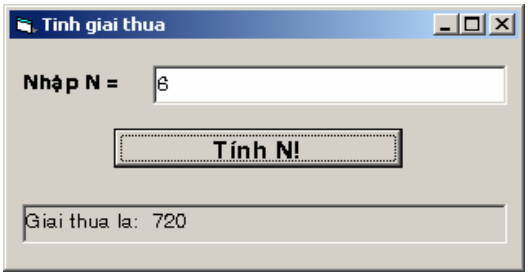
Hình I.19: For…Next

Sự kiện Command1_Click()

```
Dim i As Integer
For i = 1 To 10
     Form1.FontSize = 10 + i
     Print "Line "; i
Next
```

Chạy chương trình.

7) Thiết kế chương trình cho phép tính N! (N! = 1*2*3*…*N). Giao diện đề nghị:



Hình I.20: Tính N!

8) Thiết kế chương trình cho phép nhập vào một số nguyên N; sau đó tính các tổng sau:
   a. S=1 + 2 + ... + n
   b. S=1/2 + 2/3 + ... +n/(n+1)
   c. S= - 1 +2 - 3 +4 - ... + $(-1)^n$n

9) Thiết kế chương trình cho phép nhập vào số nguyên dương N; sau đó tìm số nguyên dương k nhỏ nhất sao cho $\frac{2}{1*3}+\frac{3}{2*4}+...+\frac{k}{(k-1)*(k+1)} \geq N$.

10) Thiết kế chương trình cho phép nhập vào 2 số nguyên A, B; sau đó tìm UCLN và BCNN của hai số a và b theo thuật toán sau (Ký hiệu UCLN của a, b là (a,b) còn BCNN là [a,b])
   - Nếu a chia hết cho b thì (a,b) = b
   - Nếu a = b*q + r thì (a,b) = (b,r)
   - [a,b] = a*b/(b,r)

11) Thiết kế chương trình cho phép nhập vào số nguyên N; sau đó viết 1 hàm tính N!; cuối cùng hiển thị kết quả giá trị N!.

12) Thiết kế chương trình cho phép nhập vào 2 số nguyên N, K; sử dụng hàm tính N! ở trên, viết một hàm tính giá trị tổ hợp chập K của N phần tử theo công thức $C_N^K = \frac{N!}{K!*(N-K)!}$.

13) Thiết kế chương trình cho phép nhập vào số thực X và số nguyên N; sau đó viết các hàm tính các tổng sau rồi hiển thị kết quả:

- $S= 1 + x + x^2 + x^3 + ... + x^n$
- $S= 1 - x + x^2 - x^3 + ... (-1)^n x^n$
- $S= 1 + x/1! + x^2/2! + x^3/3! + ... + x^n/n!$

14) Sử dụng vòng lặp **Do While … Loop** thiết kế chương trình cho phép nhập vào một số nguyên, sau đó thông báo kết quả xem số đó có phải là số nguyên tố hay không?

Đoạn chương trình kiểm tra số nguyên N có nguyên tố hay không:

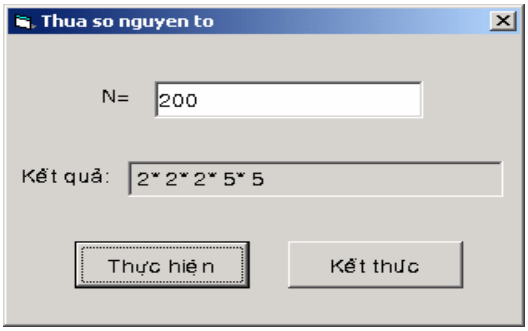
```
i = 2
Do While (i < N) And (N Mod i <> 0)
        i = i + 1
Loop
If  i = N Then N là số nguyên tố
Else N không là nguyên tố
```

15) Làm lại bài tập 11 (tính N!) nhưng sử dụng vòng lặp Do While … Loop.

16) Làm lại bài tập 15 (kiểm tra số nguyên tố) nhưng bằng cách sử dụng Do Until … Loop.

17) Làm lại bài tập 11 (tính N!) nhưng sử dụng vòng lặp Do Until … Loop.

18) Thiết kế chương trình cho phép nhập vào một số nguyên N; sau đó phân tích số nguyên này ra thừa số nguyên tố. Giao diện chương trình có thể như sau:



Hình I.21: Thừa số nguyên tố

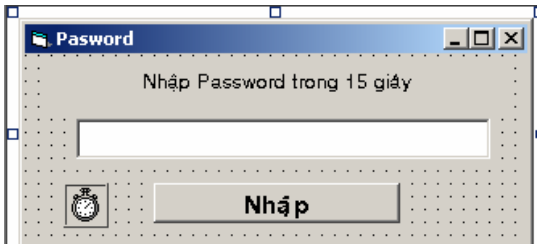
19) Sử dụng điều khiển định thời (Timer).

Tạo một chương trình cho phép người dùng 15 giây để nhập mật khẩu trong một TextBox.

Nếu người dùng không nhập mật khẩu đúng trong thời gian nói trên, chương trình hiển thị thông báo "**Time Expired**" (Hết thời gian) và đóng chương trình.

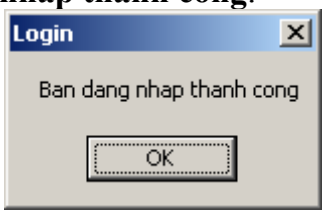
Thời gian làm bài tập: 30 phút.

Giao diện đề nghị:



Hình I.22: Giao

- Nhập vào mật khẩu cần thiết (giả sử mật khẩu là: **Secret**)
- Nếu nhập đúng mật khẩu, rồi nhấp nút **Nhập**, một hộp thông báo xuất hiện với nội dụng: **Ban dang nhap thanh cong**.



Hình I.23: Lỗi đăng nhập

- Nếu nhập mật khẩu sai, rồi nhấp nút **Nhập**, một thông báo xuất hiện với nội dung: **Xin loi, chung toi khong biet ban!**

Hình I.24: Lỗi đăng nhập

Sau đó nhấp nút OK trên hộp thông báo này thì chương trình cho bạn nhập lại mật khẩu.

- Nếu thời gian quá 15 giây mà người dùng chưa nhập đúng mật khẩu thì một thông báo sẽ hiện lên **Xin loi, thoi gian da het**; sau đó chương trình sẽ kết thúc.

Hình I.25: Báo hết giờ

20) Thiết kế chương trình tương tự như ứng dụng Canculator của Windows.

Chương 2 LẬP TRÌNH SỰ KIỆN NÂNG CAO & ĐỒ HỌA TRONG VISUAL BASIC

Mục tiêu:

Chương này gồm các bài tập nhằm mục đích rèn luyện sinh viên các kỹ năng lập trình sự kiện nâng cao như các thao tác xử lý chuột, bàn phím… cũng như giúp cho sinh viên có cái nhìn sơ lược về cách thức xử lý đồ họa trong Visual Basic.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

- Cách thức sử dụng menu trong thiết kế giao diện.
- Cách xử lý các sự kiện chuột và bàn phím.
- Các phương thức đồ họa cơ bản.

Kiến thức có liên quan:

Giáo trình Visual Basic, Chương 6.

Tài liệu tham khảo:

Visual Basic 6 Certification Exam Guide - Chapter 3, Page 69 - Dan Mezick & Scot Hillier - McGraw-Hill - 1998.

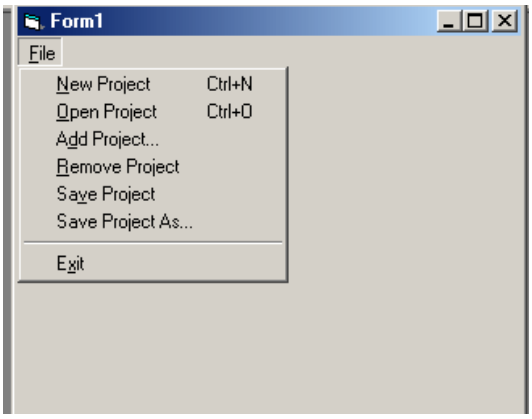
http://www.vovisoft.com/VisualBasic/VB6Chapter12C.htm

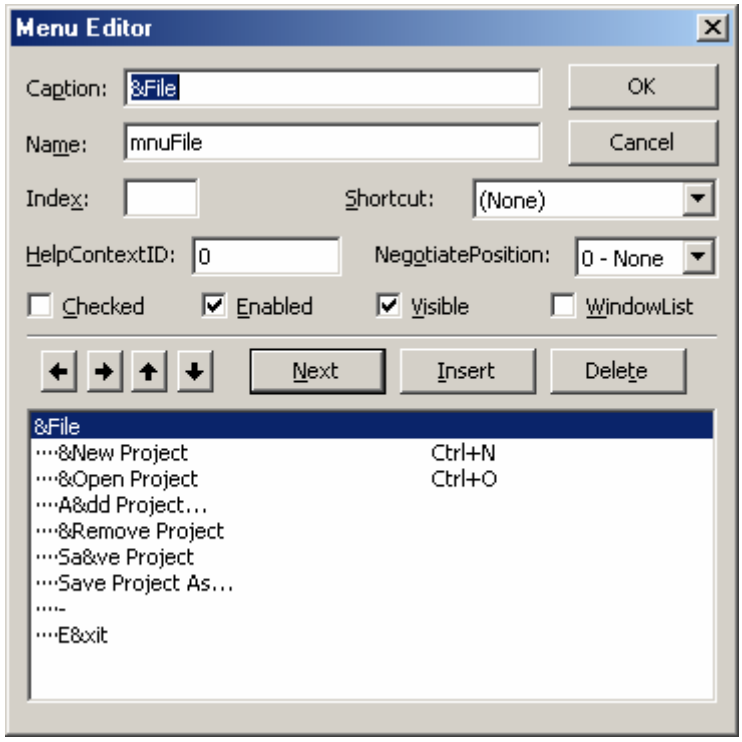
I. HƯỚNG DẪN

Bài tập 2-1

XÂY DỰNG MENU VÀ HỘP THOẠI CƠ BẢN

Bước 1: Tạo dự án mới tên Bt2-1 trong thư mục Basic\Bt2-1. Chú ý thường xuyên lưu dự án lại.
Bước 2: Bắt đầu với Menu Editor. Nhấp chuột phải lên Form1. Chọn Menu Editor. Lúc này cửa sổ Menu Editor sẽ hiện ra.





Hình II.1: Tạo menu

Bước 3: Tạo menu File bằng cách thiết lập các thuộc tính sau:

Caption: &File
Name: mnuFile

Bước 4: Định nghĩa các phần tử của menu File, các phần tử này sẽ xuất hiện khi ta nhấp vào File. Ta nhấp nút Next, vệt sáng sẽ di chuyển xuống 1 hàng, ta sẽ điền các thông tin vào.

Bước 5: Các phần tử của menu File phải được đặt trong cùng một cấp: Bằng cách nhấp chọn mũi tên phải, ta đã xác định các phần tử này thuộc menu File.

Bước 6: Định nghĩa các thuộc tính sau cho phần tử của menu sau:

Caption: &New Project
Name: mnuFileNew

Bước 7: Chọn nút OK của Menu Editor, sau đó thực thi dự án. Khi nhấp chuột vào menu File ta sẽ thấy xổ xuống phần tử New Project của Menu File. Bây giờ trở lại màn hình soạn thảo.

Bước 8: Trở lại cửa sổ Menu Editor và thêm các phần tử tiếp theo; nhớ kiểm tra thứ tự của cấp mà phần tử cần thêm vào (phải nằm trong menu File). Mỗi lần thêm một phần tử của menu (sau khi điền Caption và Name), cần chọn nút Next để định nghĩa một phần tử mới. Cần lưu ý các phần tử của menu File phải cùng một cấp.

Bước 9: Định nghĩa các phần tử sau:

Caption: &Open Project… Name: mnuFileOpen
Caption: A&dd Project… Name: mnuFileAdd
Caption: Sa&ve Project Name: mnuFileSave
Caption: Sav&e Project As… Name: mnuFileSaveAs

Bước 10: Phần tử kế tiếp của menu sẽ là đường phân cách, đường phân cách này cũng phải có một tên, ta không thể nhấp chuột trên nó để thực thi công việc. Đường phân cách có Caption là dấu "-". Bây giờ ta thêm đường phân cách và sau đó thêm mục Exit là hoàn tất.

Bước 11: Nhấp nút Next, thêm đường phân cách:

Caption: - Name: mnuSeparator1

Bước 12: Nhấp Next, thêm mục Exit

Caption: E&xit Name: mnuFileExit

Bước 13: Cấu trúc của hệ thống menu của ta như sau:

&File
…&New Project…
…&Open Project…
…A&dd Project…
…Sa&ve Project
…Sav&e Project As…
…-
…E&xit

Từ đây ta có thể chèn phần tử bất kỳ vào menu (ở các bước trên ta chỉ chèn sau).

Bước 14: Muốn chèn thêm một phần tử, nhấp vào phần tử dưới vị trí mà phần tử mới muốn đặt tại đó. Chẳng hạn, muốn chèn một phần tử trước mục Save Project, nhấp vào Save Project sau đó chọn nút Insert. Một phần tử trắng mới sẽ xuất hiện và ta điền thông tin vào.

Bước 15: Định nghĩa mục mới:

Caption: &Remove Project Name: mnuFileRemove

Nếu mục Name là khoảng trắng thì ta sẽ nhận được một thông báo lỗi: "Menu Control must have a name". Ta phải nhập Name vào.

Bước 16: Gán phím tắt. Phím tắt cho phép ta sử dụng bàn phím để truy xuất đến các mục của Menu. Chẳng hạn muốn cho mục Open Project có phím tắt là Ctrl + O, ta chọn mục Open Project trong hộp thoại Menu Editor.

Bước 17: Nhấp OK. Lưu dự án và thực thi chương trình. Nhấp chọn mục bất kỳ trong menu, ta thấy không tác dụng. Do đó ta phải cung cấp hàm xử lý sự kiện khi nhấp vào các mục của menu.

Bước 18: Đóng cửa sổ Menu Editor, nhấp File\Exit; cửa sổ Code xuất hiện. Thêm đoạn mã sau cho sự kiện Click của mnuFileExit:

```
MsgBox "Dong ung dung…"
End
```

Bước 19: Chạy ứng dụng, chọn File\Exit. Điều gì xảy ra?

Bước 20: Trở về cửa sổ soạn thảo; nhấp chuột vào File\Open Project để mở cửa sổ soạn thảo mã lệnh cho hàm xử lý sự kiện mnuFileOpen_Click. Thêm đoạn mã sau:

```
MsgBox "Ban da nhap vao muc File\Open Project"
```

Bước 21: Chạy ứng dụng. Nhấp vào File, rồi Open Project; một thông báo hiện ra. Đóng thông báo lại

Bước 22: Ta có thể dùng phím tắt để chọn Open Project; giữ phím Alt, bấm phím f rồi o.

Bước 23: Một cách khác để chọn File\Open Project là bấm phím Ctrl + O. Như vậy, ta thấy có 3 cách để chọn File\Open Project.

Bước 24: Trong nhiều ứng dụng có sử dụng menu, sau khi chọn 1 mục trên menu, ta thấy xuất hiện một hộp hội thoại gồm các nút OK và Cancel, trên đó có nhiều tùy chọn hay yêu cầu mà người sử dụng có thể chấp nhận hay hủy bỏ. Ở đây cũng vậy, ta sẽ mở một hộp thoại tương tự như trên.

Bước 25: Nhấp chuột vào Project\Components. Một danh sách các điều khiển mà ta có thể thêm vào dự án của mình. Chọn **Microsoft Common Dialog 6.0** bằng cách đánh dấu vào checkbox và chọn OK. Lúc này VB sẽ tự động thêm điều khiển mới này vào ToolBox.

Bước 26: Điều khiển Common Dialog sẽ xuất hiện trên ToolBox, nhấp đúp trên nó và đặt nó vào vị trí bất kỳ trên Form1.

Bước 27: Nhấp chuột vào mục File\Open Project, cửa sổ soạn thảo mã lệnh hiện ra, thêm vào đoạn mã sau trong hàm xử lý sự kiện mnuFileOpen_Click:

```
Form1.CommonDialog1.ShowOpen
MsgBox "Bạn da chon tap tin: " & Form1.CommonDialog1.FileName
```

Bước 28: Trong lệnh MsgBox ở trên ta có sử dụng phép toán nối 2 chuỗi lại với nhau: chuỗi "Ban da chon tap tin: " và chuỗi Form1.CommonDialog1.FileName. Lưu ý, ta phải sử dụng phép toán "&" để nối chuỗi lại.

Bước 29: Để mở hộp hội thoại (Common Dialog) ta phải có một lời gọi hàm: ShowOpen chẳng hạn. Lúc này hộp thoại mới hiện ra.

Bước 30: Lưu dự án và chạy chương trình. Chọn File\Open Project, hộp thoại hiện ra. Chọn tập tin nào đó, điều gì xảy ra tiếp theo?

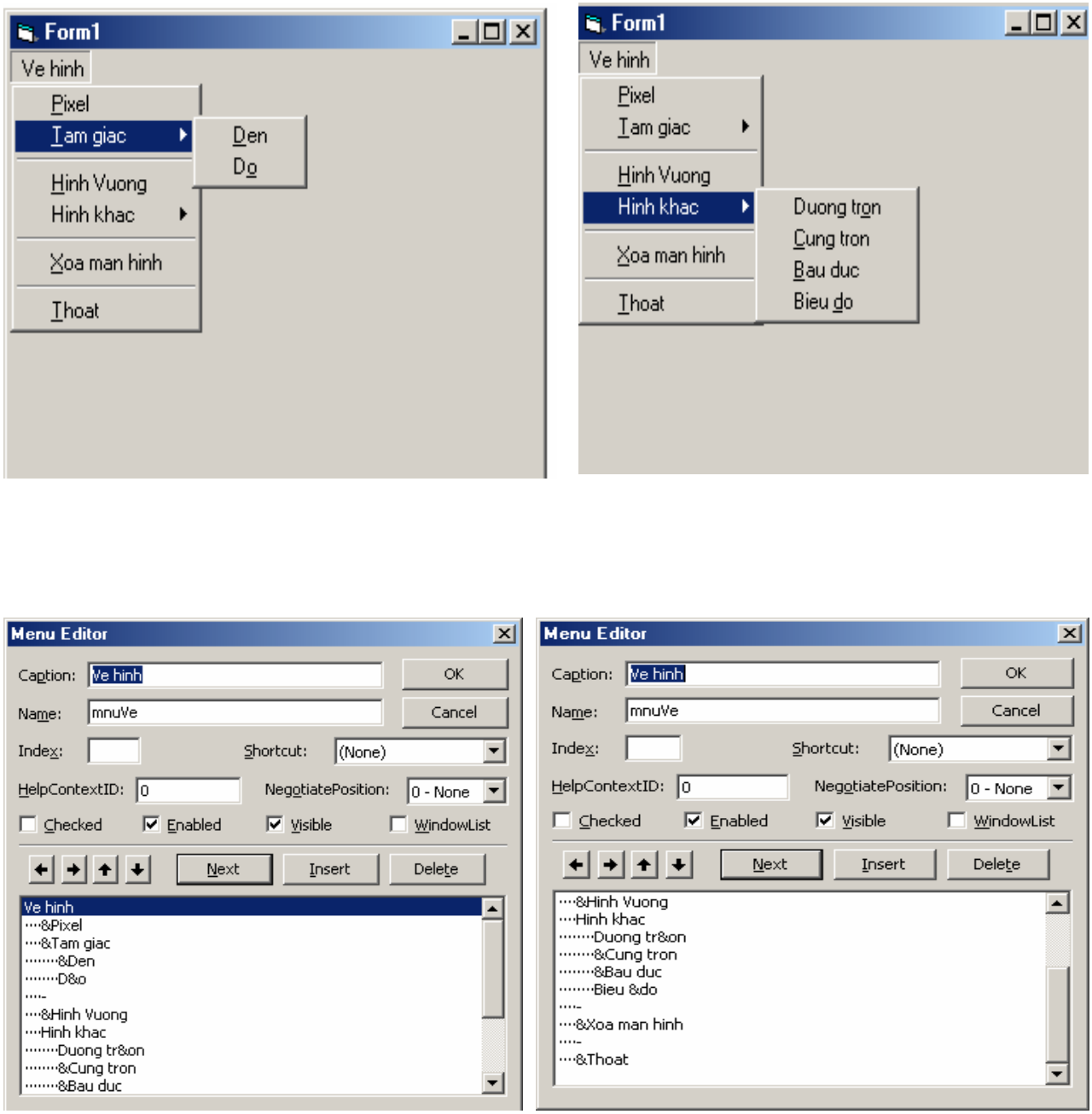
Bước 31: Tìm hiểu các lệnh ShowOpen, ShowSave, ShowPrinter, ShowColor. Ta có thể gọi chúng bằng cách thêm hàm xử lý sự kiện cho một mục của menu, chẳng hạn cho mnuFileSave_Click:

```
Form1.CommonDialog1.ShowSave
```

Bài tập 2-2

ĐỒ HỌA VỚI BASIC

Bước 1: Tạo một dự án mới trong thư mục Basic\Bt2-2.
Bước 2: Trong Form1 ta tạo Menu có dạng:



Hình II.2: Menu và giao diện

Với các thuộc tính như sau:

TT	Caption	Name	TT	Caption	Name
1	Ve hinh	MnuVe	9	Duong Tron	MnuTron
2	&Pixel	MnuPixel	10	&Cung tron	MnuCung
3	&Tam giac	MnuTg	11	&Bau duc	MnuBauduc
4	&Den	MnuTgDen	12	Bieu &do	MnuBieudo
5	D&o	mnuTgDo	13	-	MnuGach2
6	-	MnuGach1	14	&Xoa man hinh	MnuXoa
7	&Hinh vuong	mnuHV	15	-	MnuGach3
8	Hinh khac	MnuKhac	16	&Thoat	MnuThoat

HÀM PAINTPICTURE

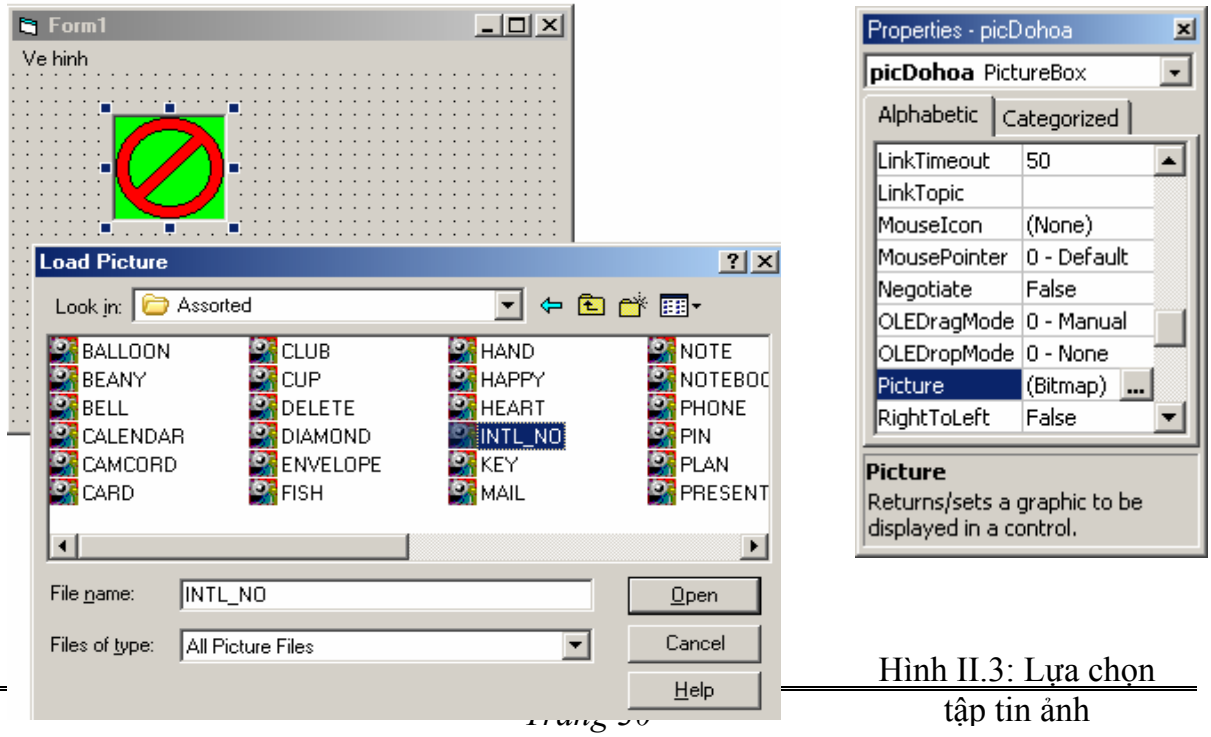
Bước 3: Hàm PaintPicture cho phép ta copy rất nhanh một khối dữ liệu đồ họa, nói nôm na là một khu vực trong một hình đồ họa trên form, PictureBox đến một nơi khác. Thí dụ ta copy một hình từ chỗ nầy đến chỗ khác trong form, hay từ form/PictureBox ra đối tượng Printer để sau đó ta in nó ra.

Ta nhấp đúp lên PictureBox Icon trong ToolBox để đặt một PictureBox lên form với các thuộc tính sau:

Name: picDohoa.

Visible: False (để ta không thấy nó lúc chạy chương trình).

Bước 4: Bây giờ ta load một hình vào thuộc tính Picture của picDohoa bằng cách chọn một tập tin hình ảnh từ cửa sổ Properties. Ở đây ta chọn **INTL_NO.BMP** từ folder **\Program Files\Microsoft Visual Studio\Common\Graphics\Bitmaps\Assorted**



Hình II.3: Lựa chọn tập tin ảnh

Trong chương trình nầy ta muốn hễ khi đè nút trái của Mouse xuống và di chuyển chuột thì khi con trỏ chuột đi đến đâu, hình INTL_NO được vẽ đến đó.

Bước 5: Ta sẽ dùng một biến để đánh dấu nút-trái-của-Mouse-Down, đặt tên là **IsMousedown**. Khi nhận được sự kiện **MouseDown** ta đặt IsMouseDown thành **True**, và khi nhận được sự kiện **MouseUp** ta đặt lại IsMouseDown thành **False**. Mỗi lần nhận được sự kiện MouseMove thì nếu IsMouseDown là **True** ta sẽ vẽ hình INTL_NO.

Trong phần [General]\[Declaration], khai báo biến sau:

```
Dim IsMouseDown As Boolean
```

Bước 6: Đầu tiên biến này phải được khởi tạo là False trong sự kiện Form_Load:

```
Private Sub Form_Load()
    IsMouseDown = False
End Sub
```

Bước 7: Ta xử lý các sự kiện MouseUp, MouseDown, MouseMove của Form như sau:

```
Private Sub Form_MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
  IsMouseDown = True
End Sub

Private Sub Form_MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
  If IsMouseDown Then
    ' Vẽ hình tại vị trí X, Y
    PaintPicture picDohoa.Picture, X, Y, picDohoa.Width, picDohoa.Height
  End If
End Sub

Private Sub Form_MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
  IsMouseDown = False
End Sub
```

Bước 8: Chọn Form1 trong cửa sổ Project Explorer, ta sẽ xử lý sự kiện khi ta nhấp chọn mục Xoa man hinh trên menu như sau:

```
Private Sub mnuXoa_Click()
    Cls
End Sub
```

Bước 9: Khi ta nhấp chọn mục Thoat trên menu, chương trình tự động đóng lại $\Rightarrow$ sự kiện mnuThoat_Click được xử lý:

```
Private Sub mnuThoat_Click()
    End
End Sub
```

Bước 10: Lưu dự án lại, chạy chương trình, thử kéo chuột trên Form. Quan sát kết quả.

HÀM PSET

Bước 11: Ta dùng **hàm PSet** (**Point Set**) để vẽ một pixel lên form. Ta cần cho biết PSet ở đâu và với màu gì, tức là ta cho nó tọa độ X,Y của pixel và một màu được tính từ hàm **RGB**.

Dưới đây là đoạn mã để vẽ pixels đủ màu lên form một cách bất chừng (randomly) về vị trí và màu sắc khi người dùng chọn mục Pixel trên menu:

```
Private Sub mnuPixel_Click()
  Dim i As Integer
  ' Tọa độ vẽ (X, Y)
  Dim iXCoord As Integer
  Dim iYCoord As Integer
  ' Màu cơ bản
  Dim iRed As Integer
  Dim iGreen As Integer
  Dim iBlue As Integer
  ' Sinh các số ngẫu nhiên
  Randomize
  ' Vẽ 2000 điểm ngẫu nhiên
  For i = 1 To 2000
    ' Lấy tọa độ X (ngẫu nhiên)
    ' Note that Rnd(1) returns a real number between 0 and 1, eg: 0.384
    iXCoord = Int(Rnd(1) * ScaleWidth)
    ' Lấy tọa độ Y (ngẫu nhiên)
    iYCoord = Int(Rnd(1) * ScaleHeight)
    ' Lấy giá trị ngẫu nhiên từ 0 – 254 cho mỗi màu cơ bản
    iRed = Int(Rnd(1) * 255)
    iGreen = Int(Rnd(1) * 255)
    iBlue = Int(Rnd(1) * 255)
    ' Vẽ 1 pixel tại tọa độ iXCoord, iYCoord
    PSet (iXCoord, iYCoord), RGB(iRed, iGreen, iBlue)
  Next
  MsgBox ("Ve xong!")
End Sub
```

Trong thí dụ trên ta dùng hàm **Randomize** để sinh sẵn trong bộ nhớ các số thực bất chừng từ 0 đến 0.999. Sau đó mỗi lần ta gọi hàm **Rnd(1)** là nó sẽ trả về một số thực bất kỳ từ bộ số do hàm Randomize sinh ra. Do đó, **Rnd(1) * ScaleWidth** sẽ cho ta một số thực có trị số từ 0 đến ScaleWidth. Muốn đổi số thực đó ra số nguyên, ta dùng hàm **Int**.

Bước 12: Lưu dự án lại, chạy chương trình. Nhấp chọn Pixel trên menu.

HÀM LINE

Hàm **Line** vẽ một đường thẳng từ một tọa độ nầy đến một tọa độ khác trong màu do ta chỉ định. Với hai hàm PSet và Line ta có thể làm được rất nhiều chuyện. Thí

dụ muốn cho một vật di động, ta xóa vật ấy bằng cách vẽ lại nó với cùng màu của BackColor của form, rồi vẽ vật ấy ở vị trí mới. Muốn vẽ một đa giác như tam giác hay chữ nhật ta ráp nhiều đường thẳng lại với nhau, đầu của mỗi đường thẳng là cuối của đường thẳng vừa mới được vẽ trước. Muốn vẽ hình dạng bên trong một hình chữ nhật ta dùng PSet…

Có ba cách để chỉ định tọa độ của hai đầu của một đường thẳng ta muốn vẽ:

✓ Cho biết tọa độ của đầu và cuối đường thẳng: Ví dụ: **Line (50, 100)-(3000, 4000).** Khi đường nầy được vẽ xong thì vị trí của con trỏ đồ họa (Graphic Cursor) có tọa độ là vị trí của cuối đường, tức là CurrentX=3000 và CurrentY=4000 trong trường hợp này.

✓ Chỉ cho biết tọa độ cuối đường thẳng: Ví dụ: **Line -(3600, 4500), vbMagenta**. Trong trường hợp nầy vị trí của Graphic Cursor (CurrentX, CurrentY) được lấy làm tọa độ của đầu đường thẳng khi vẽ. Tức là nếu trước khi thực thi dòng mã này CurrentX=3000 và CurrentY=4000 thì dòng mã trên tương đương với:

Line (3000,4000)-(3600,4500), vbMagenta

✓ Dùng chữ **Step** để nói sự khác biệt từ CurrentX và CurrentY: Ví dụ: **Line Step(400, 600)-Step(800, -500), vbGreen**. Nếu trước khi thực thi dòng mã này CurrentX=3600 và CurrentY=4500 thì dòng mã trên tương đương với:

Line (4000,5100)-(4800,4600), vbGreen

Bước 13: Ta sẽ vẽ cùng một hình tam giác nhưng với 2 màu khác nhau: Đỏ và Đen. Ta sẽ xử lý sự kiện khi chọn mục Den trên menu như sau:

```
Private Sub mnuTgDen_Click()
   ' Vẽ tam giác với màu đen
  Line (700, 500)-(2800, 2400)
  Line (2800, 2400)-(1800, 900)
  Line (1800, 900)-(700, 500)
End Sub
```

Bước 14: Vẽ tam giác với màu đỏ cùng tọa độ trên. Sự kiện mnuTgDo_Click:

```
Private Sub mnuTgDo_Click()
   ' Vẽ tam giác màu đỏ
  Line (700, 500)-(2800, 2400), vbRed
  Line -(1800, 900), vbRed
  Line -(700, 500), vbRed
End Sub
```

Bước 15: Ta có thể vẽ một hình chữ nhật với 4 góc tròn như sau:

Chọn Tools\Add Procedure… để thêm một thủ tục vào:

Name: HcnTron
Type: Sub
Scope: Private

```
Private Sub HcnTron(ByVal X1 As Integer, ByVal Y1 As Integer, _
        ByVal X2 As  Integer, ByVal  Y2 As Integer)
  Const Delta = 50
  ' Vẽ hcn với 4 góc tròn
  Line (X1 + Delta, Y1)-(X2 - Delta, Y1)
```

```
  Line -Step(Delta, Delta)
  Line -(X2, Y2 - Delta)
  Line -Step(-Delta, Delta)
  Line -(X1 + Delta, Y2)
  Line -Step(-Delta, -Delta)
  Line -(X1, Y1 + Delta)
  Line -Step(Delta, -Delta)
End Sub
```

Bước 16: Ta cũng có thể tạo bóng bên trong hình chữ nhật bằng cách dùng hàm PSet để chấm các đốm cách nhau chừng 50 pixels như sau:

```
Private Sub TaoBong(ByVal X1 As Integer, ByVal Y1 As Integer, _
        ByVal X2 As Integer, ByVal Y2 As Integer)

  Const Delta = 50
  Dim i As Integer
  Dim j As Integer
  ' Kiem tra X1 < X2 ?
  ' Đổi giá trị X1, X2 nếu X1 > X2
  If X2 < X1 Then
    Temp = X1
    X1 = X2
    X2 = Temp
  End If
  ' Kiểm tra Y1 < Y2
  ' Đổi giá trị Y1, Y2 nếu Y1 > Y2
  If Y2 < Y1 Then
    Temp = Y1
    Y1 = Y2
    Y2 = Temp
  End If
  ' Vẽ các chấm trong hcn, mỗi chấm cách nhau 50 pixel
  For i = X1 + Delta To X2 - Delta Step 50
    For j = Y1 + Delta To Y2 - Delta Step 50
      PSet (i, j)
    Next
  Next
End Sub
```

Bước 17: Bây giờ phối hợp cách vẽ hình chữ nhật với thủ tục TaoBong nói trên và hàm **Print** ta có thể viết chữ bên trong một khung màu nhạt khi ta xử lý sự kiện mnuHcn_Click:

```
Private Sub mnuHV_Click()
  Dim X1 As Integer
  Dim Y1 As Integer
  Dim X2 As Integer
  Dim Y2 As Integer
```

```
  ' Khởi tạo tọa độ đầu
  X1 = 4200: Y1 = 1000
  X2 = 6200: Y2 = 2000
  ' Vẽ hcn
  HcnTron X1, Y1, X2, Y2
  ' Tạo bóng
  TaoBong X1, Y1, X2, Y2
  ' Vị trí để xuất chữ lên màn hình
  CurrentX = X1 + 50
  CurrentY = Y1 + 50
  ' Kích thước chữ
  Font.Size = 18
  ' Hiển thị ra màn hình
  Print "Xin chao!"
End Sub
```

HÀM CIRCLE

Bước 18: Dùng hàm Circle để vẽ hình tròn, hình bầu dục và cung tròn, với bên trong không màu hay được phủ bằng một màu ta chỉ định. Để vẽ, ta phải cho biết tọa độ của tâm của đường tròn và bán kính của nó.

Ta xử lý cho sự kiện mnuTron_Click như sau:

```
Private Sub mnuTron_Click()
   ' Vẽ đường tròn tâm 2000,1500 bán kính 800
  Circle (2000, 1500), 800
  ' Vẽ đường thẳng ngang từ tâm
  Line (2000, 1500)-Step(0, 800)
  ' Vẽ đường thẳng đứng từ tâm
  Line (2000, 1500)-Step(800, 0)
End Sub
```

Bước 19: Bây giờ, thay vì vẽ nguyên một đường tròn, ta sẽ chỉ vẽ một cung tròn với màu đỏ. Để chỉ định rằng ta sẽ vẽ từ vị trí nào trên đường tròn đến vị trí nào khác, thí dụ từ 45 độ đến 230 độ, ta cần phải đổi độ ra đơn vị Radian bằng cách dùng hàm **Rads** như sau:

Chọn Tools\Add Procedure… để thêm một hàm tên Rads với các giá trị sau:

Name: Rads
Type: Function
Scope: Private

```
Private Function Rads(ByVal Degree As Single) As Single
  ' Đổi độ sang Radian
  Const PI = 22 / 7
  Rads = Degree / 180 * PI
End Function
```

Bước 20: Cung tròn luôn luôn được vẽ ngược chiều kim đồng hồ. Dưới đây là đoạn mã của sự kiện mnuCung_Click để vẽ một cung tròn màu đỏ bán kính 800, tâm (4000, 2000), từ 45 độ đến 230 độ:

```
Private Sub mnuCung_Click()
   Circle (4000, 2000), 800, vbRed, Rads(45), Rads(230)
End Sub
```

Bước 21: Ta có thể cho tô màu bên trong các hình tròn, hay Pie Slice (một phần của hình tròn) bằng cách đặt thuộc tính **FillStyle** bằng 0 và chỉ định màu **FillColor**. Một Pie Slice là một vòng cung đóng kính bởi hai đường thẳng bán kính ở hai đầu. Muốn vẽ một Pie Slice ta đánh thêm dấu trừ ("-") trước hai trị số Radian, tức là dùng **-Rads(45), -Rads(230)** thay vì **Rads(45), Rads(230)**.

Dưới đây là mã lệnh vẽ hai Pie Slices, có tâm lệch nhau một chút, đồng thời thêm chú thích 87.5% và 12.5%. Hình vẽ này tương tự như các biểu đồ dân số, diện tích… Sự kiện mnuBieudo_Click:

```
Private Sub mnuBieudo_Click()
  FillStyle = 0  ' Cho phép tô màu
  FillColor = vbYellow
  ' Vẽ một Pie Slice từ 90 độ đến 45 độ màu vàng
  Circle (3000, 4000), 800, , -Rads(90), -Rads(45)
  ' Vị trí hiển thị văn bản
  CurrentX = 2800: CurrentY = 4400
  Print "87.5%"
  FillColor = vbBlue
  ' Vẽ một Pie Slice từ 45 độ đến 90 độ màu xanh
  Circle (3050, 3900), 800, , -Rads(45), -Rads(90)
  ' Vị trí hiển thị văn bản
  CurrentX = 3400: CurrentY = 3000
  Print "12.5%"
  FillStyle = 1  ' Không cho phép tô màu
End Sub
```

Bước 22: Hàm Circle còn được dùng để vẽ các hình bầu dục (Elip). Vẽ hình bầu dục giống như vẽ một hình tròn nhưng ta cần cho thêm một tham số gọi là **Aspect**. Aspect là sự liên hệ giữa bán kính ngang (chiều ngang) và bán kính dọc (chiều cao). Thí dụ nếu Aspect=2 thì chiều cao của hình bầu dục gấp đôi chiều ngang, ngược lại, nếu Aspect=0.5 thì chiều ngang sẽ gấp đôi chiều cao.

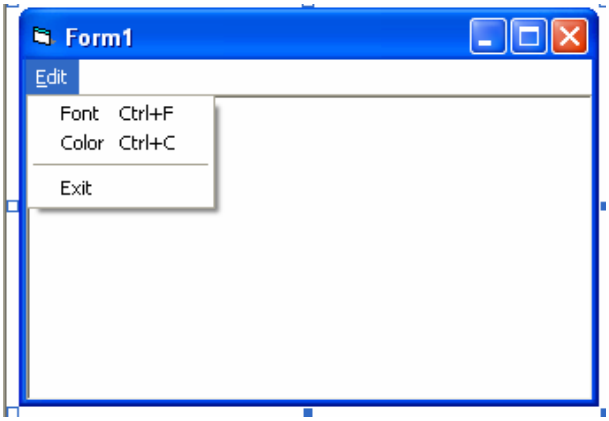
Dưới đây là đoạn mã ta dùng để vẽ hai hình bầu dục cùng kích thước, một nằm thẳng đứng và một nằm ngang $\Rightarrow$ Sự kiện mnuBauduc_Click được xử lý:

```
Private Sub mnuBauduc_Click()
  Circle (1400, 3000), 800, vbMagenta, , , 2
  Circle (1400, 3000), 800, vbBlue, , , 0.5
End Sub
```

Bước 23: Lưu dự án và chạy chương trình.

II. BÀI TẬP TỰ LÀM

Bài 1: Thiết kế chương trình có giao diện như sau:



Hình II.4: Sử dụng Common Dialog

- Mỗi khi người dùng chọn mục Font, một hộp thoại chọn Font mở ra cho phép chọn lựa các Font, sau khi họ chọn được Font, Font chữ tương ứng của TextBox cũng thay đổi theo. (*Hướng dẫn: Sử dụng thuộc tính Font của đối tượng TextBox*).
- Khi người dùng chọn Color, hộp thoại chọn màu hiển thị cho phép người dùng thay đổi màu chữ của TextBox theo màu đã chọn (*Hướng dẫn: Sử dụng thuộc tính ForeColor của đối tượng TextBox*).

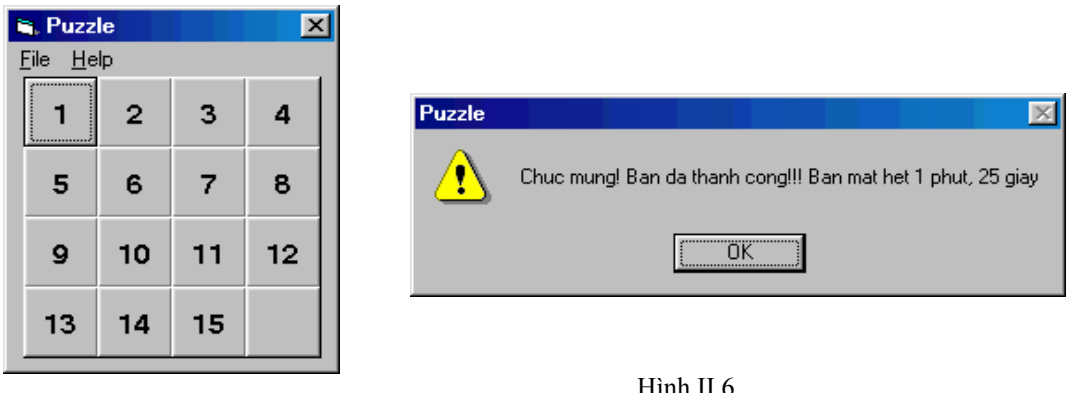
Bài 2: TRÒ CHƠI PUZZLE (SẮP SỐ)

MÔ TẢ

Không gian chơi gồm 16 ô số, được xếp trên 4 hàng, mỗi hàng gồm 4 cột. Trong đó có 15 ô có giá trị từ 1 đến 15 được phân bổ theo thứ tự ngẫu nhiên và 1 ô trống (Hình II.5)

Hình II.5

Người chơi phải tiến hành sắp lại các ô số này theo thứ tự để được kết quả như hình II.6 thì trò chơi kết thúc. Chương trình hiển thị câu chúc mừng: "Chuc mung! Ban da thanh cong!!!" & tổng thời gian chơi.



Hình II.6

Người chơi có thể thực hiện chơi lại bằng cách chọn File\New. Chương trình sẽ tự động xáo lại các ô chứa số & ô trống theo thứ tự ngẫu nhiên.

Việc sắp xếp lại các ô số được thực hiện bằng cách sử dụng ô trống. Người dùng có thể chuyển một số từ các ô lân cận đến ô trống bằng cách nhấp chuột lên ô số đó. Chẳng hạn trên hình 1 người dùng có thể nhấp ô chứa số 2 để chuyển nó đến ô trống bên dưới & khi đó ô chứa số 2 cũ sẽ thành ô trống mới.

Chú ý rằng người dùng chỉ có thể di chuyển các số thuộc những ô lân cận trống. Chẳng hạn các ô lân cận trống trong hình 1 là những ô 2, 4, 11, 15.

Hình II.7 là một ví dụ khác về các ô lân cận trống. Trong trường hợp này đó là các ô chứa số 9, 2 & 12. Để ý rằng hai ô chứa số 3 & 1 không được xem là ô lân cận.

Hình II.7

CÁC THUẬT TOÁN XỬ LÝ

Xác định một ô có phải là ô lân cận của ô trống hiện hành hay không?

Đánh số thứ tự các ô từ 0 đến 15 theo thứ tự từ phải qua trái & từ trên xuống dưới.

Xác định các lân cận cho từng ô.

Để ý rằng các ô có thứ tự 0 (hàng 1 cột 1), 3 (hàng 1 cột 4), 12 (hàng 4 cột 1) & 15 (hàng 4 cột 4) có số lân cận là 2.

8 ô có thứ tự 1 (hàng 1 cột 2), 2 (hàng 1 cột 3), 4 (hàng 2 cột 1), 7 (hàng 2 cột 4), 8 (hàng 3 cột 1), 11 (hàng 3 cột 4), 13 (hàng 4 cột 1), 14 (hàng 4 cột 3) có số ô lân cận là 3.

4 ô còn lại có thứ tự 5, 6, 9, 10 có số ô lân cận là 4.

Ví dụ: Trong hình 3:

Ô có thứ tự 6 (chứa số 2), có số ô lân cận là 4, đó là các ô có thứ tự 2 (chứa số 3), 5 (không chứa số), 7 (chứa số 15), 10 (chứa số 1).

Ô có thứ tự 16 (chứa số 7), có số ô lân cận là 2, đó là các ô có thứ tự 12 (chứa số 14), 15 (chứa số 13).

Ta có nhận xét rằng, có tất cả 16 ô mỗi ô có tối đa 4 ô lân cận. Như vậy ta có thể sử dụng một mảng 2 chiều để lưu trữ giá trị các ô lân cận.

```
Dim Neibors(0 To 15, 0 To 3) As Integer
```

Ví dụ: Ô có thứ tự 0:

```
Neibors(0,1) = 1 ' Lân cận thứ 1 của ô 0 là ô thứ 1
Neibors(0,2) = 4 ' Lân cận thứ 2 của ô 0 là ô thứ 4
Neibors(0,3) = -1 ' Không có lân cận 3
Neibors(0,4) = -1 ' Không có lân cận 4
```

Nếu một trong các ô lân cận của "ô được Click" có giá trị trống (ô trống) thì hoán đổi nội dung "ô được Click" với ô trống, ngược lại không làm gì cả.

Thuật toán xáo số

- Xem không gian chơi có 16 ô đều trống.
- Chọn ngẫu nhiên 1 trong 16 số (từ 0 đến 15) để đặt vào ô trống đầu tiên.
- Sau đó tiếp tục chọn các số còn lại (15 số chưa được chọn) để đặt vào ô thứ 2.
- Và cứ tiếp tục cho đến ô cuối cùng, sao cho đảm bảo nguyên tắc các số được chọn sẽ không được chọn lại. Điều này tránh được tình trạng có 2 hay nhiều ô có cùng giá trị số.
- Xóa trống nội dung ô chứa số 0 để tạo ô trống.

Sử dụng thủ tục Randomize & hàm Rnd, thử nghĩ xem cách thức để kiểm tra xem một số đã được sử dụng rồi hay chưa? Viết một chương trình con xáo số riêng.

Di chuyển số đến ô trống

Thực chất là hoán vị nội dung "ô được Click" & "ô trống".

- Gán nội dung "ô được Click" cho "ô trống".
- Xóa trống nội dung "ô được Click".

Kiểm tra trò chơi kết thúc

Trò chơi khi đạt đến trạng thái hình 2 là kết thúc. Viết hàm kiểm tra.

Đếm thời gian chương trình thực thi

Sử dụng bộ định thời gian để đếm thời gian thực thi của chương trình.

Chương 3 TẬP TIN

Mục tiêu:

Chương này nhằm mục đích rèn luyện sinh viên các kỹ năng thao tác với hệ thống tập tin của Windows trong VB. Bên cạnh đó, việc hệ thống lại các kiến thức của các chương trước cũng là một mục tiêu quan trọng của chương.

Học xong chương này, sinh viên phải nắm được các vấn đề sau:

- Sử dụng mô hình hệ thống tập tin.
- Cách thức truy cập tập tin tuần tự.
- Cách thức truy cập tập tin truy xuất ngẫu nhiên.

Kiến thức có liên quan:

Giáo trình Visual Basic, Chương 7.

Tài liệu tham khảo:

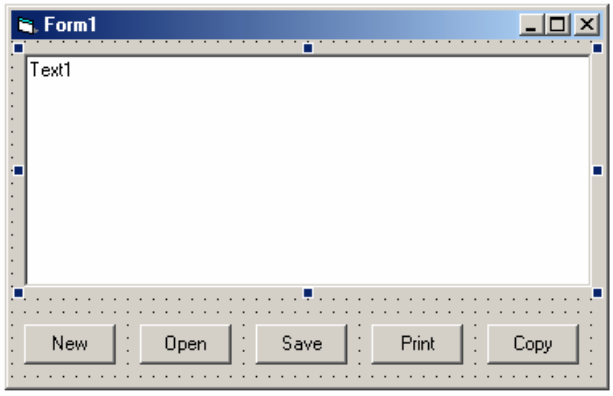
- **Visual Basic 6 Certification Exam Guide** - Chapter 7, Page 191; Chapter 13, Page 377 - **Dan Mezick & Scot Hillier** - **McGraw-Hill** - **1998**.

I. BÀI TẬP HƯỚNG DẪN

Bài tập 3-1:

XUẤT NHẬP TẬP TIN VĂN BẢN

Bước 1: Tạo Project mới tên Bt3-1 trong thư mục Basic\Bt3-1. Tạo giao diện có dạng sau:



Hình III.1: Tập tin văn bản

Item 1 – TextBox
- Name Text1
- Height 2220
- Width 6630
- Multiline True
- ScrollBars Both

Item 2 – CommandButton
- Name Command1
- Caption New

Item 3 – CommandButton
- Name Command2
- Caption Open

Item 4 – CommandButton
- Name Command3
- Caption Save

Item 5 – CommandButton
- Name Command4
- Caption Print

Item 6 – CommandButton
- Name Command5
- Caption Copy

Bước 2: Nút New có nhiệm vụ xóa văn bản trong TextBox để ta có thể khởi tạo một tài liệu mới. Do đó, trong hàm sự kiện Command1_Click, thêm vào đoạn mã:

```
Text1.Text = ""
```

GHI CHUỖI LÊN TẬP TIN

Bước 3: Ở đây ta nhập vào đoạn văn bản rồi ghi lên tập tin. Để đơn giản ta đọc và ghi từ một tập tin văn bản duy nhất tên là vidu.txt nằm trong thư mục của dự án của mình (ở đây là thư mục Bt5-1). Để ghi lên tập tin, trong hàm sự kiện Command3_Click, thêm đoạn mã sau:

```
' Ghi len tap tin
 Open App.Path & "\vidu.txt" For Output As #1
' Ghi du lieu
Print #1, Text1.Text
' Dong tap tin
Close #1
MsgBox "Van ban da duoc luu"
```

Bước 4: Việc thao tác trên tập tin được thực hiện nhờ thẻ tập tin. Thực chất đây là một số nguyên chỉ bởi VB một liên kết đến một tập tin xác định để xuất hay nhập vào tập tin đó. Ở đây là sử dụng #1. Câu lệnh Print sử dụng thẻ tập tin để ghi văn bản lên tập tin. Khi việc ghi hoàn tất, thẻ tập tin được đóng lại nhờ câu lệnh Close.
Bước 5: Chạy ứng dụng, nhấp nút Command3. Nếu chương trình thực thi tốt, ta có thể mở tập tin *vidu.txt* trong Notepad xem.

ĐỌC TỪ TẬP TIN VĂN BẢN

Bước 6: Đọc tập tin từ đĩa tương tự như ghi tập tin. Chèn đoạn mã sau trong hàm xử lý sự kiện Command2_Click

```
Text1.Text = ""
Close #1
' Mo tap tin
Open App.Path & "\vidu.txt" For Input As #1
Dim filetext As String ' Bien chuoi luu van ban

Do While Not EOF(1)
Input #1, filetext ' Doc tung dong
' Hien thi trong TextBox, chu y them vao ky tu xuong dong
Text1.Text = Text1.Text & filetext & vbCrLf
Loop
Close #1
```

Bước 7: Chạy ứng dụng. Nhấp nút Command2 để đọc từ tập tin *vidu.txt* vào TextBox. Ở đây ta có định nghĩa một biến trong lệnh

```
Dim filetext as String
```

Ở đây, mỗi lần ta đọc từng dòng trong tập tin *vidu.txt*; mỗi lần đọc như vậy ta lưu vào biến kiểu chuỗi filetext; sau đó ta nối chuỗi filetext vào sau chuỗi Text1.Text (hiển thị trong TextBox). Quá trình trên được thực hiện liên tục đến khi đọc hết nội dung tập tin nhờ vào vòng lặp:

```
Do While Not EOF(1)
```

EOF là một hàm được định nghĩa sẵn trong VB, hàm này có nhiệm vụ kiểm tra xem có đạt đến cuối tập tin hay không? Nếu nội dung tập tin vẫn chưa được đọc hết, quá trình đọc vẫn tiếp tục đến khi EOF là True.

Input #1

Đọc một chuỗi từ tập tin cho đến khi gặp ký tự xuống dòng. Ký tự xuống dòng này được bỏ qua trong lệnh Input; do đó nếu muốn hiển thị thành nhiều dòng trên TextBox, ta phải thêm vào ký tự xuống dòng cho mỗi dòng ta đọc được từ tập tin sau đó ta mới hiển thị trên TextBox. Hằng số vbCrLf là sự liên kết 2 ký tự xuống dòng và về đầu dòng.

IN VĂN BẢN RA MÁY IN

Bước 8: Nếu máy in được nối vào, máy in phải được kích hoạt. Ta có thể kiểm tra chúng bằng cách in thử vài dòng văn bản trong Word hay trong Notepad.

Bước 9: Đối tượng Printer sẽ chỉ đến máy in mặc định. Trong hàm xử lý sự kiện Command4_Click chèn thêm đoạn mã:

```
Printer.Print Text1.Text
```

Câu lệnh này dùng để in nội dung trong TextBox ra máy in. Tuy nhiên nội dung của TextBox chỉ được in khi chương trình chấm dứt. Để in ngay lập tức, ta cần phải thêm dòng sau:

```
Printer.EndDoc
```

CHÉP DỮ LIỆU VÀO CLIPBOARD

Bước 10: Trong nhiều ứng dụng, nhiều khi ta cần sử dụng dữ liệu qua lại với nhau. Chẳng hạn, người dùng có thể sử dụng dữ liệu được hiển thị trên form hiển thị của chương trình chúng ta sang chương trình xử lý văn bản Microsoft Word. Lúc này, một cách hiệu quả nhất là sử dụng đối tượng Clipboard, đối tượng này cho phép đọc và ghi lên Windows Clipboard từ chương trình ứng dụng:

Thêm đoạn mã sau vào hàm sự kiện Command5_Click:

```
Clipboard.Clear
Clipboard.SetText Text1.Text
```

Đóng cửa số mã lệnh lại và chạy chương trình ứng dụng. Nhập một đoạn văn bản, sau đó nhấp Command5. Từ Microsoft Word, sử dụng menu Edit\Paste để lấy dữ liệu từ Clipboard hiển thị.

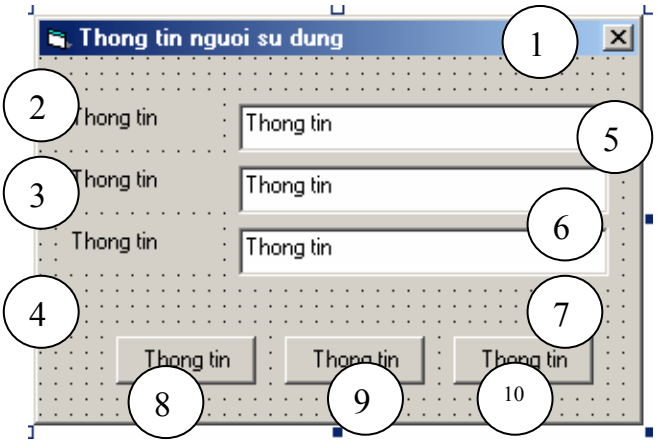
Bài tập 3-2

THAO TÁC VỚI RESOURCE FILE

Mục tiêu: Giúp làm quen với tập tin resource của VB, nhất là củng cố các thao tác trên tập tin.

Bước 1: Tạo thư mục Basic\Bt3-2. Tạo một dự án mới trong thư mục này.

Bước 2: Tạo giao diện như hình sau:



Hình III.2: Tập tin resource

Trong đó:

Item 1: Caption: Thong tin nguoi su dung
BorderStyle: 3-Fixed Dialog
StartUpPosition: 2-Center Screen

Item 2: Label
Name: lblHelp
Index: 0

Item 3: Label
Name: lblHelp
Index: 1

Item 4: Label
Name: lblHelp
Index: 2

Item 5: TextBox
Name: txtHelp
Index: 0

Item 6: TextBox
Name: txtHelp
Index: 1

Item 7: TextBox
Name: txtHelp
Index: 2

Item 8: CommandButton
Name: cmdHelp
Index: 0

Item 9: CommandButton
Name: cmdHelp
Index: 1

Item 10: CommandButton
Name: cmdHelp
Index: 2

Bước 3: Ta nhận thấy các điều khiển có cùng một tên hiển thị (Thong tin). Mục tiêu của ta là sử dụng tập tin resource (tài nguyên) để thay đổi tên hiển thị trên các điều khiển. Để tạo tập tin tài nguyên, ta vào mục **ADD-IN\ADD-IN MANAGER** trên menu của VB. Trong các mục của **ADD-IN MANAGER** nhấp đúp vào resource editor và đóng mục **ADD-IN MANAGER** lại.

Bước 4: Chọn Tools\Resource Editor trên menu. Mở String Table Editor bằng cách nhấp chuột lên biểu tượng **abc** của Resource Editor. Cửa sổ soạn thảo cho tập tin tài nguyên sẽ mở ra. Ta nhập các hàng như sau:

ID	RESOURCE STRING	ID	RESOURCE STRING
1	Ten	7	So dt
2	Ho	8	So CMND
3	Ma nv	9	T. trang hn
4	Huy bo	10	Huy bo
5	Vo hieu hoa	11	Ve truoc
6	Ke	12	Hoan tat

Bước 5: Lưu tập tin tài nguyên lại.

Bước 6: Mở cửa sổ soạn thảo mã lệnh. Tạo kiểu do người dùng định nghĩa để lưu dữ liệu cần nhập vào. Thêm đoạn mã sau:

```
Private Type yeucau
        ho As String
        ten As String
        manv As String
        sodt As String
        socmnd As String
        tinhtranghn As String
End Type
```

Bước 7: Ta thêm 2 biến nữa; một biến lưu thông tin về người sử dụng (theo kiểu ở trên), một biến lưu thứ tự các bước mà người sử dụng đã nhập thông tin của mình vào.

```
Private chisobuoc As Integer
Private cacyeucau As yeucau
```

Bước 8: Trong chương trình này, người sử dụng phải nhập thông tin của mình vào thông qua các bước nhập, trong đó các điều khiển được sử dụng như một mảng các điều khiển. Để tận dụng chúng ta cần khai báo các hằng số để biết hiện thời người dùng đang ở bước thứ mấy của quá trình nhập thông tin cũng như biết được mình đã nhấp vào nút nhấn nào trong quá trình trên. Do đó, ta thêm đoạn khai báo sau:

```
' cac hang so
Private Enum buoc
        buoc1 = 1
        buoc2 = 2
        buoc3 = 3
End Enum


Private Enum nhannut
        nuttrai
        nutgiua
        nutphai
```

```
End Enum
```

Bước 9: Chương trình này thể hiện trên một form duy nhất và sử dụng mảng các điều khiển để tạo các bước để người dùng nhập thông tin vào. Do đó ta sử dụng tập tin tài nguyên để hiển thị các tên của điều khiển nhằm hiển thị cho chính xác. Vì thế ta cần có một hàm (thủ tục) để cập nhật thông tin nhập vào dựa vào các bước của người dung khi nhập thông tin vào. Vào Tools\Add Procedure để thêm thủ tục sau:

```
Public Sub Hienthi()
        Dim i As Integer
        ' Kiem tra cac buoc
        Debug.Assert chisobuoc = 1 Or chisobuoc = 2

        For i = 0 To 2
  ‘ Nhan
  lblHelp(i).Caption = LoadResString((chisobuoc - 1) * 6 + (i + 1))
  ' Nut
  cmdHelp(i).Caption = LoadResString((chisobuoc - 1) * 6 + (i + 4))
  If UCase(cmdHelp(i).Caption) = "VO HIEU HOA" Then
        cmdHelp(i).Visible = False
  Else
        cmdHelp(i).Visible = True
  End If
   txtHelp(i).Text = ""
        Next
End Sub
```

Bước 10: Khi chương trình thực hiện, ta phải ở bước thứ nhất của quá trình nhập liệu
$\Rightarrow$ Thêm đoạn mã sau trong thủ tục xử lý sự kiện Form_Load:

```
        chisobuoc = 1
        Hienthi
```

Bước 11: Mỗi khi có một nút nhấn được nhấp, quá trình nhập liệu chuyển sang bước kế tiếp; người sử dụng có thể đi đến bước kế tiếp hay trở về bước trước đó trong quá trình này. Vì các nút nhấn (button) là một mảng điều khiển (control array) nên chúng có cùng một sự kiện Click tác động vào gọi là cmdHelp_Click. Hàm xử lý này có tham số là một chỉ số kiểu Integer để nhận biết nút nhấn nào được nhấp. Ở đây, ta thêm đoạn mã sau trong hàm xử lý sự kiện này.

```
Private Sub cmdHelp_Click(Index As Integer)
  Select Case chisobuoc
        Case buoc1
                cacyeucau.ten = txtHelp(0).Text
                cacyeucau.ho = txtHelp(1).Text
                cacyeucau.manv = txtHelp(2).Text
        Case buoc2
                cacyeucau.sodt = txtHelp(0).Text
                cacyeucau.socmnd = txtHelp(1).Text
                cacyeucau.tinhtranghn = txtHelp(2).Text
  End Select
```

```
' Cac nut nhan
  Select Case Index
        Case nuttrai
              ' Huy bo
         End
        Case nutgiua
              ' ve truoc
              chisobuoc = buoc1
              Hienthi
        Case nutphai
              ' di toi
              chisobuoc = chisobuoc + 1
              If chisobuoc = buoc2 Then
                    Hienthi
              Else
                   Guiyeucau
              End If
  End Select
End Sub
```

Bước 12: Khi quá trình nhập thông tin kết thúc, thông tin này được lưu vào trong một tập tin văn bản, nhờ thủ tục Guiyeucau. Thêm thủ tục Guiyeucau vào nhờ mục Tools\Add Procedure và nhập đoạn mã sau:

```
Public Sub Guiyeucau()
  On Error GoTo Guiloi
        ' Lay the tap tin
        Dim intFile As Integer
        intFile = FreeFile()

        ' Viet len tap tin
        Open App.Path & "\yeucau.txt" For Output As #intFile
        Print #1, "ho: " & cacyeucau.ho
        Print #1, "ten: " & cacyeucau.ten
        Print #1, "manv: " & cacyeucau.manv
        Print #1, "sodt: " & cacyeucau.sodt
        Print #1, "socmnd: " & cacyeucau.socmnd
        Print #1, "tinhtranghn: " & cacyeucau.tinhtranghn
        Close #intFile

        MsgBox "Yeu cau cua ban da duoc goi di", vbOKOnly + vbInformation,
_
              "Goi yeu cau"
  End
  Exit Sub

  Guiloi:
        MsgBox Err.Description, vbOKOnly + vbExclamation, "Goi yeu cau"
```

```
    Exit Sub
End Sub
```

Bước 13: Lưu và thực thi chương trình.

Bài tập 3-3
CHƯƠNG TRÌNH XỬ LÝ VĂN BẢN ĐƠN GIẢN

GIAO DIỆN ĐA TÀI LIỆU

Bước 1: Tạo một dự án lưu trong thư mục Basic\Bt3-3.

Giao diện đa tài liệu (MDI Form) gồm một cửa sổ cha chứa nhiều cửa sổ con (chẳng hạn như các chương trình Microsoft Word, Excel được tổ chức theo dạng này). Để thêm vào dự án, ta chọn mục Project\Add MDI Form từ menu của VB.

Bước 2: Ta cho Form1 trở thành một cửa sổ con của MDI Form bằng cách chọn thuộc tính MDIChild = True.

HÀM MAIN (SUB MAIN)

Bước 3: Trong chương trình ta cần điều khiển mọi thứ kể từ khi các cửa sổ con của MDI Form xuất hiện, do đó ta cần phải bắt đầu thực thi chương trình của ta từ hàm Main (Sub Main). Ta chọn mục Project\ Add Module để thêm một Modul vào dự án của mình, sau đó ta chọn Tools\Add Procedure để thêm hàm Main vào (Public Sub Main); hàm này ta dùng để bắt đầu gọi thực thi chương trình của mình. Để chọn thực thi chương trình từ hàm Main, chọn Project\Properties; chọn Start up Object là Sub Main.

Bước 4: Thêm dòng lệnh sau vào hàm Main:

```
MDIForm1.Show
```

Bước 5: Chương trình cần có một hệ thống menu để gọi thực thi. Do đó, chọn MDI Form, sau đó chọn Tools\Menu Editor để tạo menu sau:

Menu Name	Menu Caption
mnuFile	&File
mnuFileNew	&New
mnuFileOpen	&Open…
muFileSave	&Save
mnuFileBar	-
mnuFileExit	E&xit

Bước 6: Ta xử lý sự kiện mnuFileExit_Click nhờ đoạn mã sau:

```
Private Sub mnuFileExit_Click()
    Dim f As Form
    ' Thoat cac cua so con
     For Each f In Forms
        If TypeOf f Is Form1 Then
            Unload f
            Set f = Nothing
        End If
```

```
        Next

        ' Thoat cua so cha
        Unload Me
    End Sub
```

Bước 7: Để tạo ra một tài liệu trắng cho chương trình xử lý văn bản, ta cần phải có một TextBox trong Form1. Người sử dụng đánh nội dung vào TextBox, do đó ta thêm một TextBox vào Form1 với các thuộc tính sau:

MultiLine: True
ScrollBars: 2-Vertical

Ta xử lý sự kiện Form_Resize của Form1 như sau:

```
        Private Sub Form_Resize()
            Text1.Height = Me.ScaleHeight
            Text1.Width = Me.ScaleWidth
            Text1.Left = 0
            Text1.Top = 0
        End Sub
```

Bước 8: Mỗi lần chọn mục New trên cửa sổ chương trình ứng dụng, một khung cửa sổ trắng hiện ra để ta nhập văn bản vào. Do đó, thêm đoạn mã sau trong thủ tục xử lý sự kiện mnuFileNew_Click:

```
        Private Sub mnuFileNew_Click()
            Dim f As Form1
            Static n As Integer

            Set f = New Form1
            f.Text1.Text = ""
            n = n + 1
            f.Caption = "Document " & Format(n)
            f.Show
        End Sub
```

THAO TÁC TRÊN TẬP TIN

Bước 9: Ta cần phải có hộp thoại nhằm chọn tập tin để lưu (hay mở tập tin) trong chương trình xử lý văn bản. Do đó ta cần thêm một Dialog Control vào chương trình. Đánh dấu vào mục chọn Microsoft Common Dialog Control 6.0 (SP3). Sau đó ta thêm Dialog Control từ ToolBox vào MDIForm1. Ta xử lý sự kiện mnuFileSave_Click nhờ đoạn mã sau:

```
        Private Sub menuFileSave_Click()
            Dim tenfile As String
            CommonDialog1.ShowSave
            tenfile = CommonDialog1.FileName
            Open tenfile For Output As #1
                Print #1, MDIForm1.ActiveForm.Text1.Text
            Close #1
        End Sub
```

Bước 10: Khi mục Open của menu được chọn, hộp thoại Open File được mở ra⇒ sự kiện mnuFileOpen_Click được xử lý như sau:

```
Private Sub mnuFileOpen_Click()
        Dim tenfile As String, s As String
        CommonDialog1.ShowOpen
        tenfile = CommonDialog1.FileName
        If UCase(Right(tenfile, 3)) <> "TXT" Then Exit Sub

        Call mnuFileNew_Click
        Open tenfile For Input As #1
        Do Until EOF(1)
        Line Input #1, s
        Me.ActiveForm.Text1.Text = Me.ActiveForm.Text1.Text & s & vbCrLf
        Loop
        Close #1
End Sub
```

Bước 11: Lưu dự án và chạy chương trình. Tạo mới, lưu, mở một số tài liệu. Nhận xét kết quả.

Bài tập 3-4

THAO TÁC VỚI ĐỐI TƯỢNG WORD

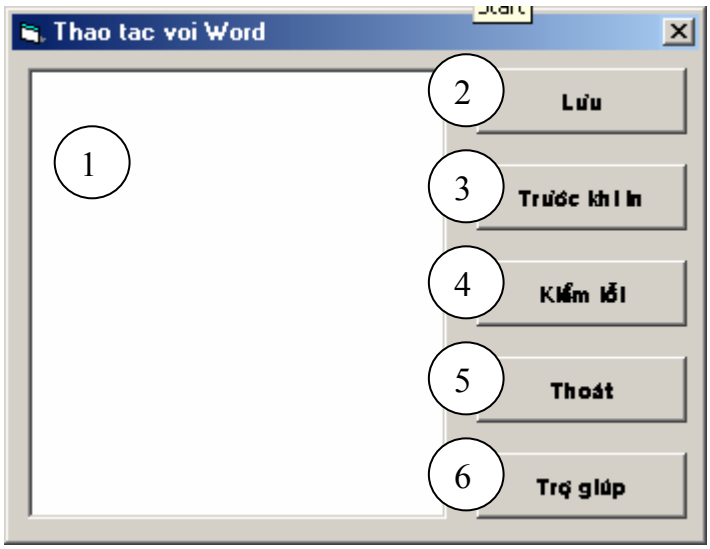
Mục đích: Windows có sẵn một số đối tượng khi ta cài đặt Windows hay khi cài một số phần mềm. Bài tập này giúp ta tìm hiểu cách thức truy xuất các đối tượng có sẵn này từ Visual Basic.

THAM CHIẾU ĐỐI TƯỢNG

Bước 1: Tạo thư mục Basic\Bt3-4. Khởi động một dự án mới trong thư mục này.
Bước 2: Trong bài tập này ta có tham chiếu đến đối tượng Word của Microsoft Word; do đó ta phải có thao tác tham chiếu đến đối tượng này trong màn hình soạn thảo VB bằng cách: Chọn **Project\References** trên menu. Trong cửa số References, thiết lập tham chiếu đến: **Microsoft Word 9.0 Object Library** và **Microsoft Office 9.0 Library**. Sau đó đóng cửa sổ References lại.
Bước 3: Ta có thể kiểm tra các đối tượng trên có được đưa vào hay chưa nhờ thao tác: Chọn **View\Object Browser**.

XÂY DỰNG ỨNG DỤNG

Bước 4: Tạo giao diện chương trình có dạng sau:



Hình III.3: Thao tác với đối tượng Word

Trong đó:
1: TextBox
Name: txtWord
Multiline: True
ScrollBar: 2-Vertical
2: CommandButton
Name: cmdLuu
Caption: Lưu
3: CommandButton
Name: cmdTruoc
Caption: Trước khi in
4: CommandButton
Name: cmdCTa
Caption: Kiểm lỗi.
5: CommandButton
Name: cmdThoat
Caption: Thoát
6: CommandButton
Name: cmdGiup
Caption: Trợ giúp

Bước 5: Để sử dụng được mô hình, ta phải khai báo một số biến đối tượng của Word. Trong phần [General]\ [Declarations], khai báo những biến sau:

```
Public ungdung As Word.Application
Public tailieu As Word.Document
Public trogiup As Office.Assistant
```

Bước 6: Khi chương trình thực hiện, điều ta muốn là một tài liệu mới của Word được tạo ra để ta có thể thao tác trên chúng một cách gián tiếp thông qua chương trình VB của mình. Tạo một tài liệu Word mới tương đương với việc tạo ra một thể hiện của đối tượng Document. Vì thế, chèn đoạn mã sau vào thủ tục Form_Load để tạo ra một tài liệu Word mới từ chương trình VB.

```
Set ungdung = CreateObject("Word.Application")
Set tailieu = ungdung.Documents.Add
Set trogiup = ungdung.Assistant
```

Bước 7: Nút Lưu có nhiệm vụ ghi tất cả những gì trên TextBox vào đối tượng Word mới tạo ra. Do đó, ta xử lý sự kiện cmdLuu_Click như sau:

```
' Ghi tai lieu moi
tailieu.Content.Text = txtWord.Text
MsgBox "Van ban duoc luu trong Word", vbOKOnly, "Word"
```

Bước 8: Nút Trước khi in có nhiệm vụ hiển thị tài liệu Word giống như khi chúng được in ra giấy; vì thế sự kiện cmdTruoc_Click được xử lý như sau:

```
tailieu.PrintPreview
ungdung.Visible = True
ungdung.Activate
```

Bước 9: Nút Kiểm lỗi thực hiện thao tác kiểm lỗi chính tả cho tài liệu Word, thao tác này được xử lý trong thủ tục cmdCTa_Click:

```
tailieu.CheckSpelling
txtWord.Text = tailieu.Content.Text
```

Bước 10: Nút Thoát sẽ đóng cửa sổ chứa tài liệu Word lại. Chèn đoạn mã sau trong thủ tục cmdThoat_Click để đóng Word lại:

```
ungdung.Quit SaveChanges = False
```

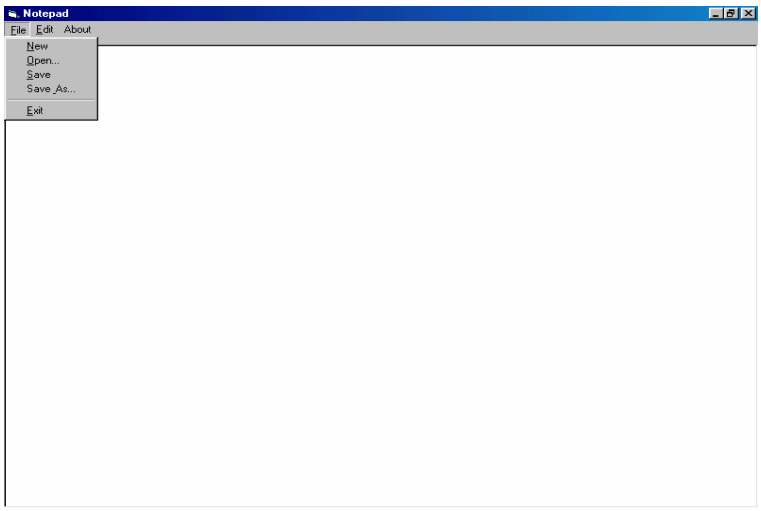
Bước 11: Khi ta nhấp vào nút Trợ giúp thì cửa sổ Help của Office hiện ra. Do đó, thêm đoạn mã sau trong thủ tục cmdGiup_Click để mở cửa sổ Help của Office:

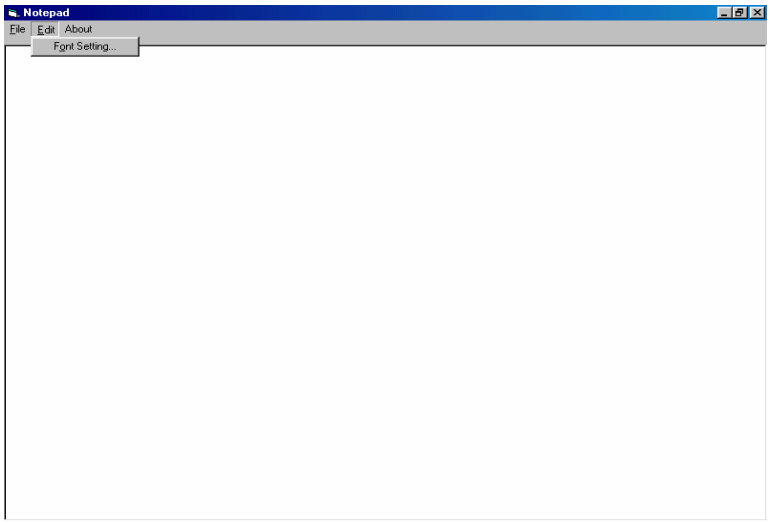
```
ungdung.Visible = True
ungdung.Activate
trogiup.Help
```

Bước 12: Lưu dự án và chạy chương trình. Nhập vài dòng văn bản vào TextBox và nhấp nút Lưu. Mỗi khi văn bản được lưu, ta nhấp nút Trước khi in để xem tài liệu trước khi. Khi văn bản được hiển thị, nhấp nút Kiểm lỗi. Sau đó thử nhấp nút Trợ giúp để xem phần giúp đỡ của Office. Cuối cùng nhấp nút Thoát để thoát khỏi Word.

II. BÀI TẬP TỰ LÀM

1) Thiết kế chương trình như ứng dụng Notepad của Windows.
   - Thiết kế giao diện





Hình III.5: Giao diện Notepad

- Xử lý các sự kiện
  - ✓ Mỗi khi Form thay đổi kích thước, TextBox cũng thay đổi theo cho phù hợp với Form
  - ✓ New
  - ✓ Open, Save, Save As: mở hộp thoại Common Dialog cho phép chọn tập tin để mở hay lưu. Sử dụng đối tượng FileSystemObject để thao tác với tập tin văn bản.
  - ✓ Exit
  - ✓ Font Setting: Mở ra hộp thoại chọn Font, thiết lập Font của TextBox chính là Font được chọn trong hộp thoại.
- Xử lý mở rộng:
  - ✓ Khi người dùng đã lưu tập tin rồi, lần thứ hai bấm vào Save thì không mở hộp thoại Common Dialog nữa mà sẽ lưu với tên tập tin đã chọn trong lần Save đầu tiên.
  - ✓ Mỗi khi người dùng thay đổi nội dung của một tập tin, sau đó họ chọn Exit để đóng ứng dụng lại; một hộp thông điệp (Message Box) sẽ mở ra hỏi có lưu tập tin hay không?

2) Đại lý Minh Thành của công ty Unilever Việt Nam tại Cần Thơ cần quản lý thông tin về các mặt hàng mà đại lý nhận từ công ty. Các thông tin cần quản lý gồm: *Mã mặt hàng, tên mặt hàng, đơn vị tính, giá* của mặt hàng đó. Các thông tin này được mô tả như sau:

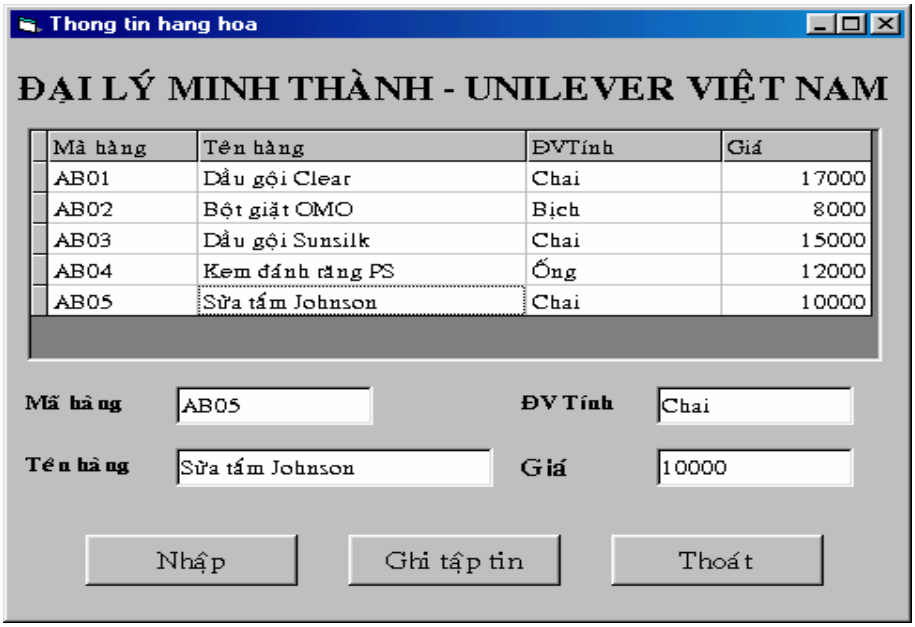
```
Type HangHoa
        MaHang As String*5
        TenHang As String*40
        DVTinh As String*15
        Gia As Double
End Type
```

Dựa vào thông tin mô tả trên, Anh (Chị) hãy:

a. Tạo dự án mới và viết các khai báo thích hợp.
b. Thiết kế Form chính như sau:

Hình III.6: Giao diện chính

c. Khi người dùng chọn mục **Thoát**, rồi nhấp chọn **Thực hiện**, chương trình chấm dứt. Viết mã lệnh để xử lý đối với trường hợp này.
d. Khi người dùng nhấp chọn **Nhập liệu**, rồi **Thực hiện**, một Form sẽ mở ra cho phép nhập thông tin hàng hóa vào. Hãy thiết kế Form này với các chức năng như hình dưới:

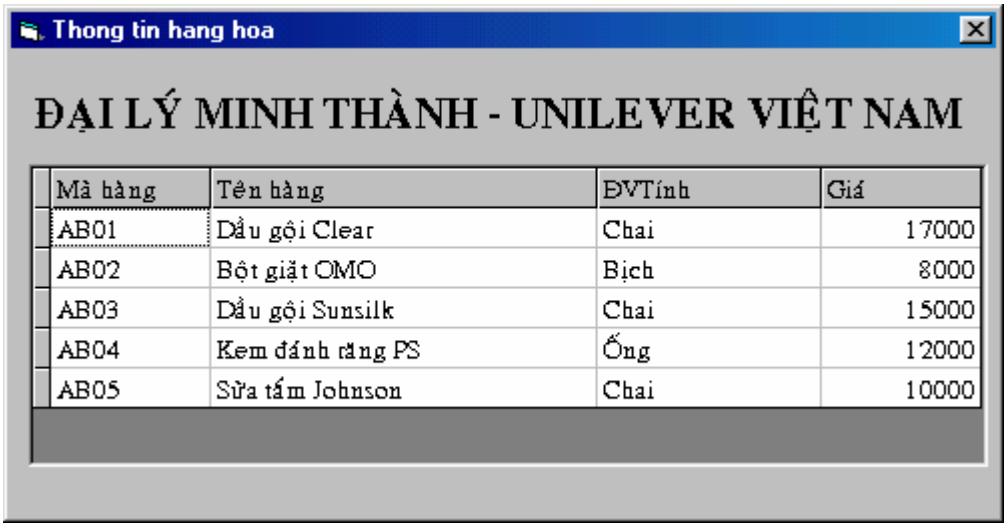


Hình III.7: Form nhập liệu

e. Mỗi khi người dùng nhập thông tin vào các ô TextBox, rồi chọn nút nhấn **Nhập**, những thông tin đó sẽ được lưu lên lưới hiển thị. Khi chọn **Ghi tập tin**, một hộp thoại (Common Dialog) *lưu tập tin* hiện ra cho phép chọn đường dẫn

và tên tập tin, sau đó ghi những thông tin trên lưới vào tập tin đã chọn (*với cấu trúc tập tin được mô tả ở phần đầu*). Nút nhấn **Thoát** sẽ đóng Form này lại, trở về Form chính ban đầu. Viết các đoạn xử lý thích hợp.

f. Ở Form chính ban đầu (hình III.6), khi người dùng chọn **Hiển thị thông tin hàng hóa**, một hộp thoại *mở tập tin* (CommonDialog) hiện ra cho phép chọn tên tập tin chứa dữ liệu về hàng hóa đã được tạo ra ở câu e. Sau đó đọc dữ liệu từ tập tin rồi hiển thi trên lưới:



Hình III.8: Đọc từ tập tin

Hãy thiết kế Form và viết mã lệnh xử lý các sự kiện thích hợp.

CƠ SỞ DỮ LIỆU SỬ DỤNG

Các chương kế tiếp là phần lập trình Visual Basic truy xuất cơ sở dữ liệu (CSDL). Trong các bài tập trên CSDL, ta có sử dụng CSDL HangHoa.MDB của Access. Cơ sở dữ liệu này đã có sẵn, sinh viên có thể liên hệ cán bộ giảng dạy để lấy về. Thông tin các bảng (Table) của CSDL này như sau:

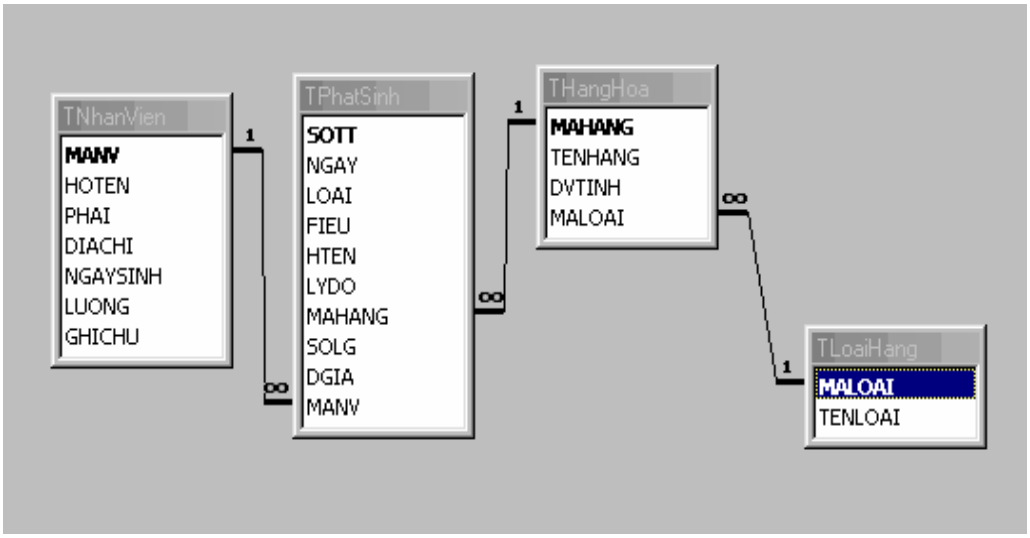
TLOAIHANG(**MaLoai**, TenLoai): Mỗi loại hàng hóa có mã loại và tên loại.

THANGHOA(**MaHang**, TenHang, DVTinh, *MaLoai*). Mỗi hàng hóa có mã hàng hóa, tên hàng hóa, đơn vị tính và chỉ thuộc 1 loại hàng hóa nào đó.

TNHANVIEN(**MaNV**, HoTen, Phai, Diachi, Ngaysinh, Luong, Ghichu): Mỗi nhân viên có mã nhân viên, họ tên, phái, địa chỉ nhân viên, ngày sinh, lương của nhân viên đó là bao nhiêu và có thể có một vài ghi chú về nhân viên đó.

TPHATSINH(**SOTT**, Ngay, Loai, Fieu, Hten, Lydo, *MaHang*, Solg, Dgia, *MaNV*): Mỗi một phát sinh được ghi nhận thành một chứng từ có SoTT, ngày phát sinh chứng từ, loại phát sinh là nhập (hay xuất)…, số phiếu, họ tên khách hàng, lý do phát sinh ứng với hàng hóa nào (mã hàng), số lượng và đơn giá là bao nhiêu, nhân viên phụ trách phát sinh là gì (MaNV).

Bảng quan hệ giữa các Table này như sau:



Các mối quan hệ của CSDL HangHoa.mdb

Chương 4 CÁC ĐỐI TƯỢNG TRUY CẬP DỮ LIỆU

Mục tiêu:

Chương này gồm các bài tập nhằm rèn luyện sinh viên cách thức lập trình cơ sở dữ liệu sử dụng thư viện đối tượng Data Access Objects (DAO). Đây là cách thức lập trình phổ biến đối với các ứng dụng chạy trên máy đơn.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

- Sử dụng điều khiển dữ liệu để truy xuất cơ sở dữ liệu.
- Sử dụng thư viện đối tượng DAO để lập trình cơ sở dữ liệu.

Kiến thức có liên quan:

- Giáo trình Visual Basic, chương 9.

Tài liệu tham khảo:

Visual Basic 6 Certification Exam Guide - Chapter 5, Page 139 - **Dan Mezick & Scot Hillier** - **McGraw-Hill** - **1998**.

http://www.vovisoft.com/VisualBasic/VB6Chapter14.htm
http://www.vovisoft.com/VisualBasic/VB6Chapter15.htm

I. HƯỚNG DẪN

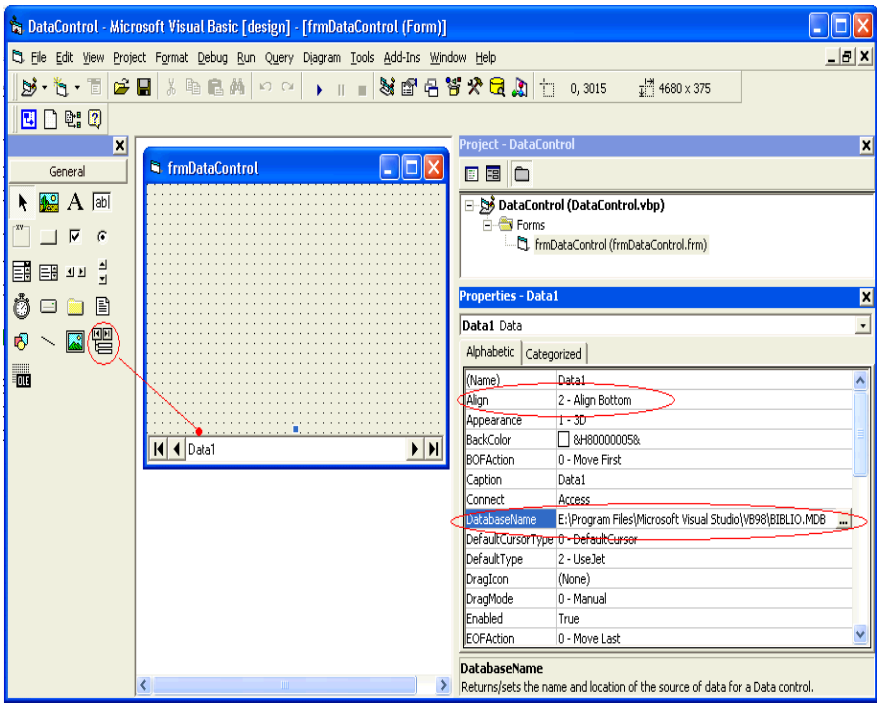
Bài tập 4-1

SỬ DỤNG DATA CONTROL

Bước 1: Tạo một dự án mới tên DataControl trong thư mục Basic\Bt4-1.

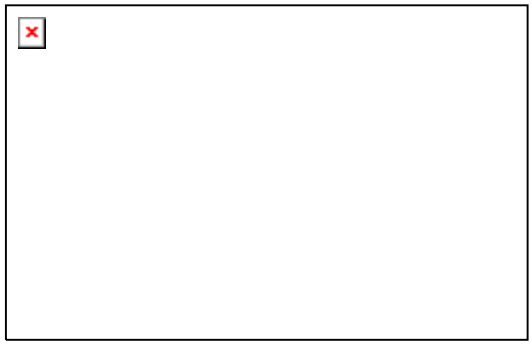
Bước 2: Nhấp đúp lên Icon của Control Data trong Toolbox. Một Control Data tên **Data1** sẽ hiện ra trên Form. Muốn cho nó nằm bên dưới Form, hãy đặt thuộc tính **Align** của nó trong Properties Window thành **2 - Align Bottom**.

Nhấp bên phải hàng **property DatabaseName**, kế đó click lên nút lựa chọn có ba chấm để chọn một file cơ sở dữ liệu Access từ hộp thoại cho Data1. Ở đây ta chọn **E:\Program Files\Microsoft Visual Studio\VB98\BIBLIO.MDB** (tùy máy tính có thể là ổ C hay ổ đĩa D)



Hình IV.1: Xác lập thuộc tính cho Data Control

Bước 3: Trong chương trình này ta làm việc với **table Titles** của cơ sở dữ liệu BIBLIO.MDB, để xem và sửa đổi các records. Để ý thuộc tính **DefaultType** của Data1 có trị số **2- UseJet**, tức là dùng kỹ thuật DAO, thay vì dùng kỹ thuật ODBC.

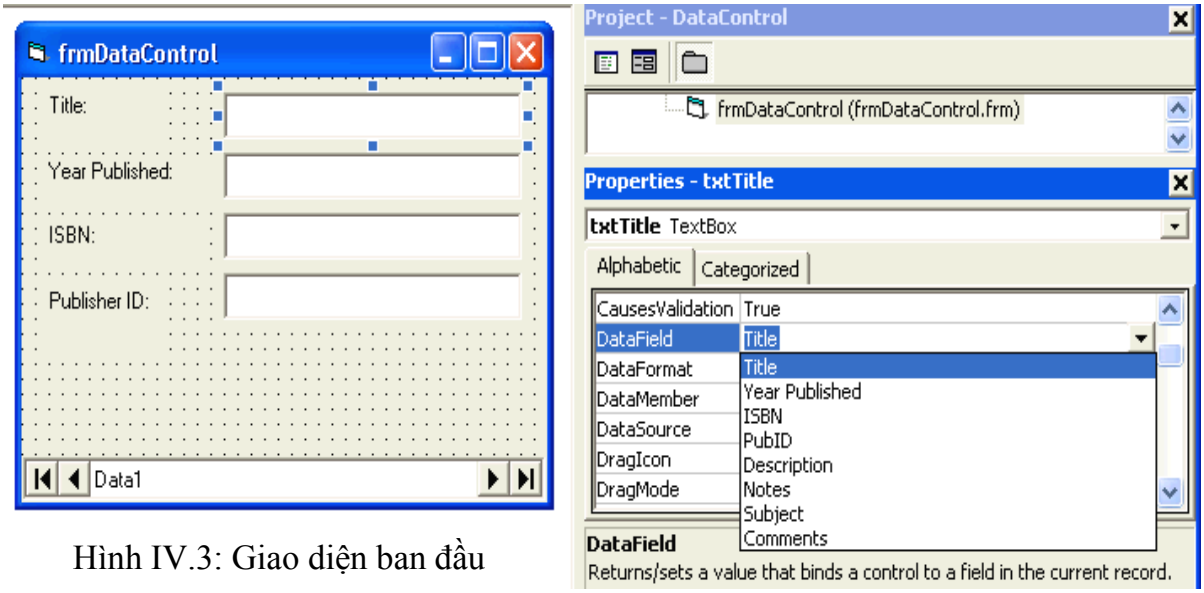


Hình IV.2: Recordset Type

Khi ta nhấp chuột lên thuộc tính **Recordsource** của Data1, rồi nhấp lên tam giác nhỏ bên phải, một ComboBox sẽ mở ra cho ta thấy danh sách các tables trong cơ sở dữ liệu, chọn **Titles**. Để ý thuộc tính **RecordsetType** của Data1 có trị số là **0 - Table**:

Bước 4: Một từ mới mà ta sẽ dùng thường xuyên khi truy cập dữ liệu trong VB6 là **Recordset** (bộ records). Recordset là một **Set of records**, nó có thể chứa một số records hay không có record nào cả. Một record trong Recordset có thể là một record lấy từ một Table. Trong trường hợp ấy có thể ta lấy về tất cả records trong table hay chỉ những records thỏa đúng một điều kiện, thí dụ như ta chỉ muốn lấy các records của những sách xuất bản trước năm 1990 (Year Published < 1990).

Tạo Form có dạng như sau:



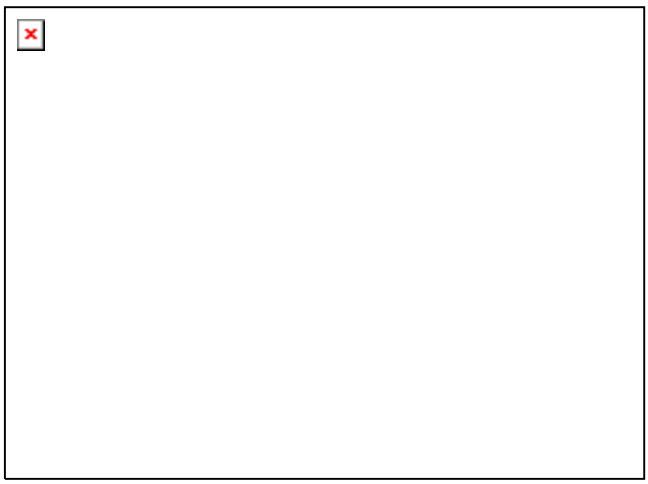
Hình IV.3: Giao diện ban đầu

4 labels với caption của chúng: **Title, Year Published, ISBN** và **Publisher ID**. Kế đó cho thêm 4 textboxes tương ứng và đặt tên chúng là **txtTitle, txtYearPublished, txtISBN** và **txtPublisherID**.

Bước 5: Chọn textbox txtTitle, rồi đặt thuộc tính **Datasource** của nó trong Properties Window thành **Data1**. Khi click lên **property Datafield** của txtTitle và mở ComboBox ra ta sẽ thấy liệt kê tên các trường trong table Titles. Đó là vì Data1 được coi như trung gian lấy table Titles từ cơ sở dữ liệu. Ở đây ta sẽ chọn cột Title.

Tương tự cho 3 textboxes còn lại, và chọn các cột Year Published (năm xuất bản), ISBN (số lý lịch trong thư viện quốc tế), và PubID (số lý lịch nhà xuất bản) làm Datafield cho chúng.

Bước 6: Lưu dự án và chạy chương trình. Ta sẽ thấy giao diện như sau:



Hình IV.4: Kết quả thực thi ứng dụng

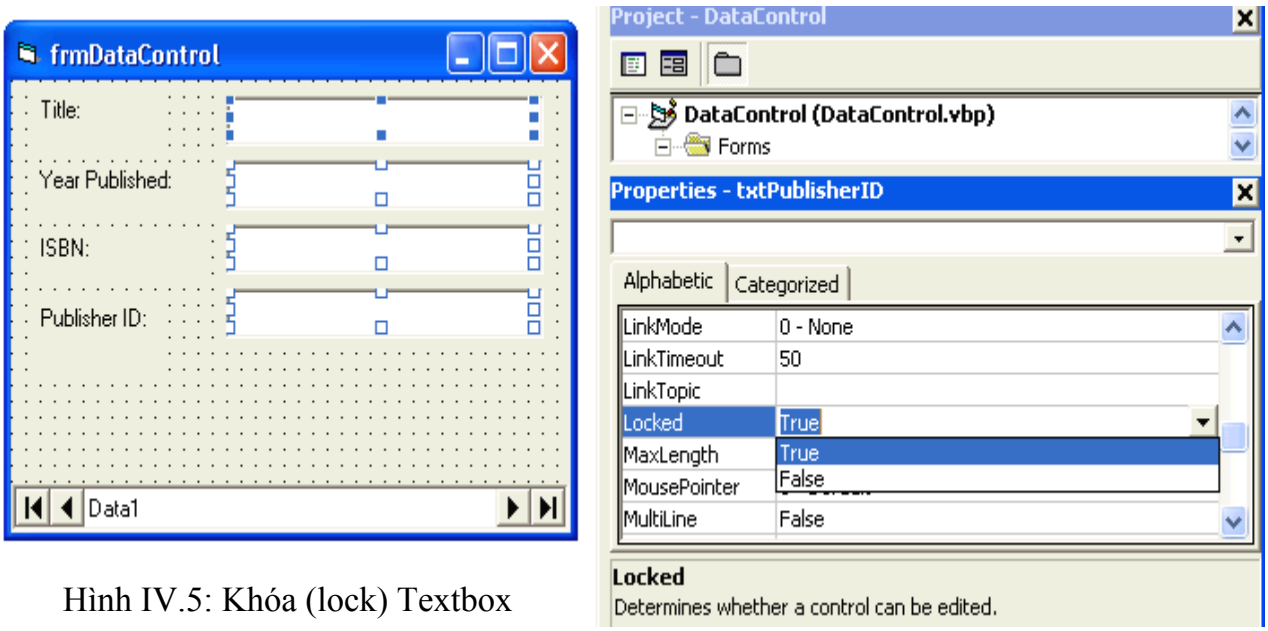
Nhận xét:

✓ Ta có thể bấm các nút di chuyển **Navigator Buttons** để đi đến các record **đầu (first), trước (previous), kế (next)** và **cuối (last)**. Mỗi lần ta di chuyển đến một record mới là chi tiết của record ấy sẽ hiển thị. Nếu không dùng các Navigator Buttons, ta cũng có thể viết đoạn mã để làm công tác tương đương bằng cách gọi các hàm trên Recordset là **MoveFirst, MovePrevious, MoveNext** và **MoveLast**.

✓ Khi record cuối của Recordset đang hiển thị, nếu ta gọi hàm MoveLast thì thuộc tính **EOF (End-Of-File)** của Recordset trở thành True. Tương tự như vậy, khi record thứ nhất của Recordset đang hiển thị, nếu ta gọi hàm MovePrevious thì thuộc tính **BOF (Begin-Of-File)** của Recordset trở thành True. Nếu một Recordset không có chứa một record nào cả thì cả hai thuộc tính EOF và BOF đều là True.

✓ Khi record đầu tiên đang hiển thị, nếu ta sửa **Year Published** để đổi từ 1985 thành **1983** rồi nhấp Navigator button Next để hiển thị record thứ nhì, kế đó click Navigator button Previous để hiển thị lại record đầu tiên thì ta sẽ thấy là trường Year Published của record đầu tiên đã thật sự được thay đổi (updated) thành 1983.

Điều nầy có nghĩa rằng khi di chuyển từ record nầy đến record khác thì nếu record nầy đã có sự thay đổi do người sử dụng, nó lưu trữ sự thay đổi đó trước khi di chuyển. Chưa chắc là ta muốn điều nầy, do đó, nếu ta không muốn người sử dụng tình cờ sửa đổi một record thì ta có thể đặt thuộc tính **Locked** của các textboxes ấy thành True để người sử dụng không thể sửa đổi các textboxes như trong hình dưới đây:



Hình IV.5: Khóa (lock) Textbox

CHỈ ĐỊNH VỊ TRÍ CƠ SỞ DỮ LIỆU LÚC CHẠY CHƯƠNG TRÌNH

Bước 7: Cách chỉ định tên cơ sở dữ liệu trong giai đoạn thiết kế (at design time) ta đã dùng trước đây tuy tiện lợi nhưng hơi nguy hiểm, vì khi ta cài chương trình nầy lên máy tính khác, chưa chắc tập tin cơ sở dữ liệu ấy nằm trong một thư mục có cùng tên. Ví dụ trên máy tính này thì cơ sở dữ liệu nằm trong thư mục E:\Program Files\Microsoft Visual Studio\VB98, nhưng trên máy tính khác thì cơ sở dữ liệu nằm trong thư mục D:\Basic\Bt4-1 chẳng hạn. Do đó, khi chương trình khởi động ta nên xác định lại vị trí của cơ sở dữ liệu. Chẳng hạn ta muốn để cơ sở dữ liệu trong cùng một thư mục với chương trình đang chạy, ta có thể dùng thuộc tính **Path** của Application Object **App**.

Khai báo một biến tên **duongdan** trong phần [General]\[Declaration] của Form1:

```
Dim duongdan As String
```

Bước 8: Ta xử lý sự kiện Form_Load như sau:

```
Private Sub Form_Load()
  duongdan = App.Path
  If Right(duongdan, 1) <> "\" Then duongdan = duongdan & "\"
  Data1.DatabaseName = duongdan & "BIBLIO.MDB"
End Sub
```

THÊM BỚT CÁC RECORDS

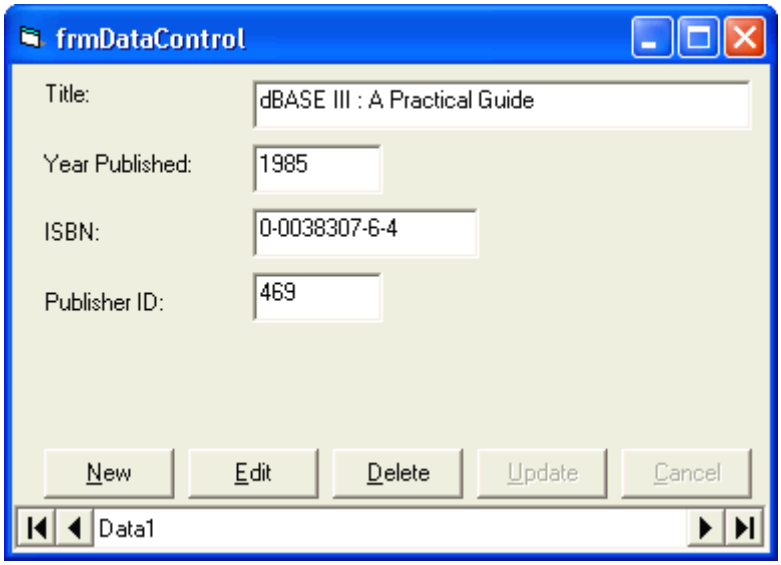
Bước 9: Chương trình đến đây tạm ổn, nhưng nó không cho ta công cụ để thêm (add), bớt (delete) các records. Bây giờ hãy đặt vào Form 5 buttons tên: **cmdEdit, cmdNew, cmdDelete, cmdUpdate** và **cmdCancel**.

Bước 10: Lúc chương trình mới khởi động, người sử dụng đang xem thông tin các records thì hai buttons **Update** và **Cancel** không cần phải làm việc. Do đó ta sẽ Lock (khóa) các textboxes và disable hai buttons nầy vì không cần dùng chúng.

Bước 11: Trong **Sub SetControls** dưới đây, ta dùng một tham số gọi là **Editing** với trị số False hay True tùy theo người dùng đang xem (browse) hay sửa đổi (Edit), ta gọi là **Browse mode** và **Edit mode**. Trong **Edit mode**, các Textboxes được unlocked (mở khóa) và các nút **cmdNew, cmdDelete** và **cmdEdit** trở nên vô hiệu lực:

```
Sub SetControls(ByVal Editing As Boolean)
  ' Lock/Unlock textboxes
  txtTitle.Locked = Not Editing
  txtYearPublished.Locked = Not Editing
  txtISBN.Locked = Not Editing
  txtPublisherID.Locked = Not Editing
  ' Enable/Disable buttons
  CmdUpdate.Enabled = Editing
  CmdCancel.Enabled = Editing
  CmdDelete.Enabled = Not Editing
  cmdNew.Enabled = Not Editing
  CmdEdit.Enabled = Not Editing
End Sub
```

Trong **Browse mode**, Form có dạng như sau:



Hình IV.7: Kết quả thực thi

Bước 12: Thủ tục SetControls được gọi trong **Sub Form_Load** khi chương trình khởi động và sự kiện **CmdEdit_Click** được xử lý như sau:

```
Private Sub Form_Load()
  duongdan = App.Path
  If Right(duongdan, 1) <> "\" Then duongdan = duongdan & "\"
  Data1.DatabaseName = duongdan & "BIBLIO.MDB"
  SetControls (False)
End Sub

Private Sub CmdEdit_Click()
```

```
    SetControls (True)
  End Sub
```

Bước 13: Khi ta xóa một record trong recordset, vị trí của record hiện tại (current record) vẫn không thay đổi. Do đó, sau khi xóa một record ta phải **MoveNext**. Tuy nhiên, nếu ta vừa xóa record cuối của Recordset thì sau khi MoveNext, thuộc tính **EOF** của Recordset sẽ thành True. Thành ra ta phải kiểm tra điều đó, nếu đúng vậy thì lại phải **MoveLast** để hiển thị record cuối của Recordset như trong đoạn mã của **Sub cmdDelete_Click** dưới đây:

```
Private Sub CmdDelete_Click()
  On Error GoTo DeleteErr
  With Data1.Recordset
    ' Xoa record
    .Delete
    ' Nhay den record ke
    .MoveNext
    If .EOF Then .MoveLast
    Exit Sub
  End With
DeleteErr:
  MsgBox Err.Description
  Exit Sub
End Sub
```

Bước 14: Ta có thể Update (cập nhật) một record trong Recordset bằng hàm **Update**. Nhưng ta chỉ có thể gọi hàm Update của một Recordset khi Recordset đang ở trong **Edit hay AddNew mode**. Ta đặt một Recordset vào Edit mode bằng cách gọi hàm **Edit** của Recordset, thí dụ như **Data1.Recordset.Edit**. Tương tự như vậy, ta đặt một Recordset vào AddNew mode bằng cách gọi hàm **AddNew** của Recordset, thí dụ như **Data1.Recordset.AddNew**.

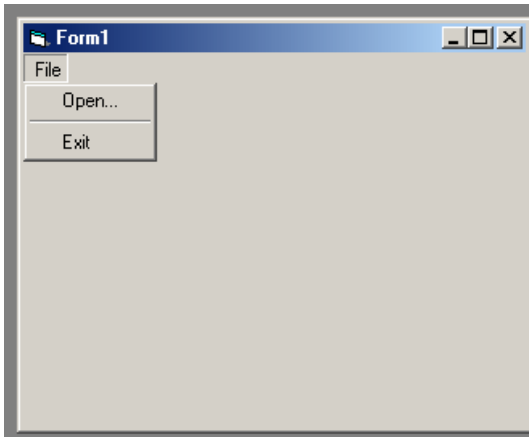
```
Private Sub cmdNew_Click()
  Data1.Recordset.AddNew
  SetControls (True)
End Sub

Private Sub cmdUpdate_Click()
   Data1.Recordset.Edit
   Data1.Recordset.Update
   SetControls (False)
End Sub
```

Bước 15: Lưu dự án và chạy chương trình.

Bài tập 4-2

CÁC ĐỐI TƯỢNG CƠ BẢN CỦA DAO

Bước 1: Tạo thư mục Basic\Bt4-2. Tạo giao diện cho chương trình như sau:



Hình IV.8: Giao diện ban đầu

Các tên của thành phần menu lần lượt là: mnuFile, mnuOpen, mnuExit.

Sau đó vào Project\References…, đánh dấu vào Microsoft DAO 3.51 Object Library; chọn OK.

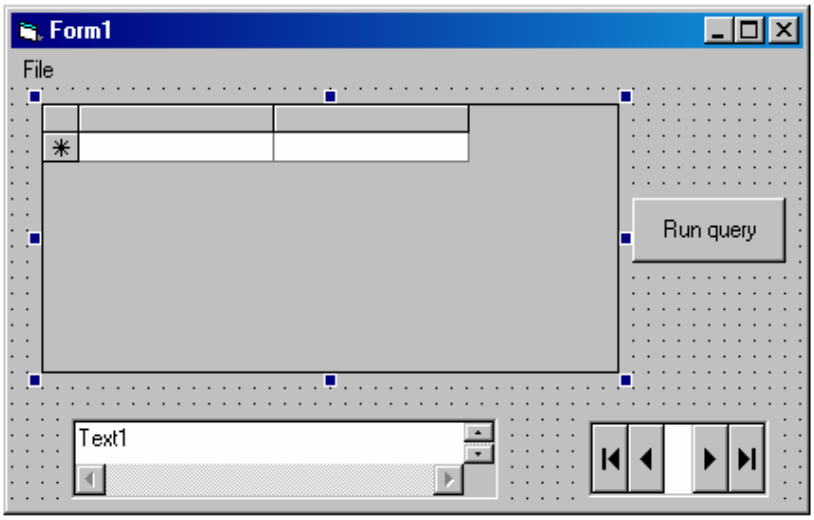
Bước 2: Thêm một Common Dialog vào Form1, tên là dlgDatabase.

Bước 3: Thêm một DBGrid vào form bằng cách chọn: Project\Components…, đánh dấu Microsoft Data Bound Grid Control 5.0 (SP3); rồi chọn DBGrid trên ToolBox. Sau đó thêm một TextBox và một Data Control vào form1. Ta có các tên của điều khiển là: DBGrid1, Text1, Data1 với các thuộc tính như sau:

Item 1: TextBox
- Name: Text1
- Multiline: True
- ScrollBars = 3

Item 2: DBGrid
- Name: DBGrid1
- DataSource = Data1

Ta được hình dạng của form1 như sau:



Hình IV.9: Giao diện đầy đủ

Sau đó, thêm đoạn mã sau trong thủ tục xử lý sự kiện mnuOpen_Click:

```
CommonDialog1.FileName = "*.mdb"
CommonDialog1.Filter = "Access DBs (*.mdb)|*.mdb"
CommonDialog1.ShowOpen
Data1.DatabaseName = CommonDialog1.FileName
```

Bước 4: Thêm một nút nhấn (Button) như hình trên, Caption là Run query. Nút này có mục đích là thực thi câu lệnh SQL mà người dùng nhập vào ô Text1. Để thực thi được lệnh SQL này, ta phải gán thuộc tính Recordsource của Data Control Data1 như trong thủ tục xử lý sự kiện Command1_Click:

```
Private Sub Command1_Click()
    Data1.RecordSource = Text1.Text
    Data1.Refresh
End Sub
```

Bước 5: Trong hàm xử lý sự kiện mnuExit_Click thêm dòng mã sau:

```
End
```

Chạy chương trình, trong mục File\Open của menu chọn tập tin C:\Program Files\Microsoft Visual Studio\VB98\Biblio.mdb. Sau đó ta gõ câu lệnh SQL sau vào Text Box:

```
Select * from Publishers
```

Nhấp chuột vào nút nhấn Run query. Quan sát kết quả hiển thị.

Ta đã tạo một chương trình cho phép người sử dụng để mở một CSDL và chạy câu SQL trên CSDL đó. Bây giờ, đối với CSDL được mở ở trên, tìm xem các bảng của nó là gì nhằm mục đích xây dựng các câu truy vấn cho phù hợp.

Bước 6: Thêm đoạn mã sau vào phần khai báo của Form1:

```
Private db As DAO.Database
Private td As DAO.TableDef
Private qd As DAO.QueryDef
Private fld As DAO.Field
```

Bước 7: Trong hàm xử lý sự kiện mnuOpen_Click ta cần kiểm tra xem tập tin được chọn có phải là tập tin CSDL của Access hay không (*.mdb)? Sau đó dùng các biến được khai báo ở trên để thao tác⇒ Sửa thủ tục mnuOpen_Click như dưới đây:

```
Private Sub mnuOpen_Click()
        CommonDialog1.FileName = "*.mdb"
        CommonDialog1.Filter = "Access DBs (*.mdb)|*.mdb"
        CommonDialog1.ShowOpen

        If UCase(Right(CommonDialog1.FileName, 3)) <> "MDB" Then
                MsgBox "Khong phai la tap tin cua Microsoft Access"
        Else
        On Error Resume Next
        db.Close
        On Error GoTo 0
        Screen.MousePointer = vbHourglass
        ' Mo CSDL

        Set db = _
                DBEngine.Workspaces(0).OpenDatabase(CommonDialog1.FileN
        ame)
        Form1.Caption = "Cau SQL: Chon " & CommonDialog1.FileName
        Screen.MousePointer = vbDefault
        Data1.DatabaseName = CommonDialog1.FileName
        End If
End Sub
```

Bước 8: Ta đã mở được CSDL, bây giờ ta dùng một List Box để hiển thị tất cả các bảng của CSDL được mở ỏ trên.

Thêm một ListBox vào Form tên List1, trong hàm xử lý sự kiện mnuOpen, thêm đoạn mã sau trước lệnh End If:

```
        ' Them vao ListBox
List1.Clear
For Each td In db.TableDefs
        List1.AddItem td.Name
Next
```

Chạy chương trình, ListBox sẽ hiển thị tất cả các bảng của CSDL trên.

Bước 9: Thêm một ListBox nữa vào Form, tên List2. Thêm đoạn mã sau trong hàm xử lý sự kiện List1_Click:

```
        Private Sub List1_Click()
                ' Tim bang duoc chon trong CSDL
                Set td = New TableDef
                For Each td In db.TableDefs
                If td.Name = Me.List1.List(Me.List1.ListIndex) Then
                                Exit For
                End If
                Next
```

```
        ' Hien thi cac truong cua bang duoc chon
        For Each fld In td.Fields
        List2.AddItem fld.Name
        Next
End Sub
```

Bước 10: Chạy chương trình, chọn File\Open để chọn tập tin CSDL, lúc đó List1 sẽ hiển thị các bảng của CSDL. Nhấp chọn một bảng trong List1, List2 sẽ hiển thị tên các trường của bảng đó. Bây giờ ta tiến thêm một bước nữa là hiển thị tất cả các câu truy vấn (SQL) được lưu trong CSDL trên bằng cách:

Thêm một ListBox nữa vào Form1 tên là List3, sau đó thêm vào đoạn mã sau trong hàm xử lý sự kiện mnuOpen trước lệnh End If:

```
List2.Clear
List3.Clear
Text1.Text = ""

For Each qd In db.QueryDefs
        List3.AddItem qd.Name
Next
```

Bước 11: Chạy chương trình, kiểm tra xem điều gì xảy ra trên List3.

Đóng chương trình lại, thêm đoạn mã sau trong hàm xử lý sự kiện List3_Click:

```
Private Sub List3_Click()
        For Each qd In db.QueryDefs
        If qd.Name = List3.List(List3.ListIndex) Then
                Text1.Text = qd.SQL
        End If
        Next
End Sub
```

Chạy chương trình, mở BIBLIO.MDB, nhấp vào List3. Quan sát kết quả.

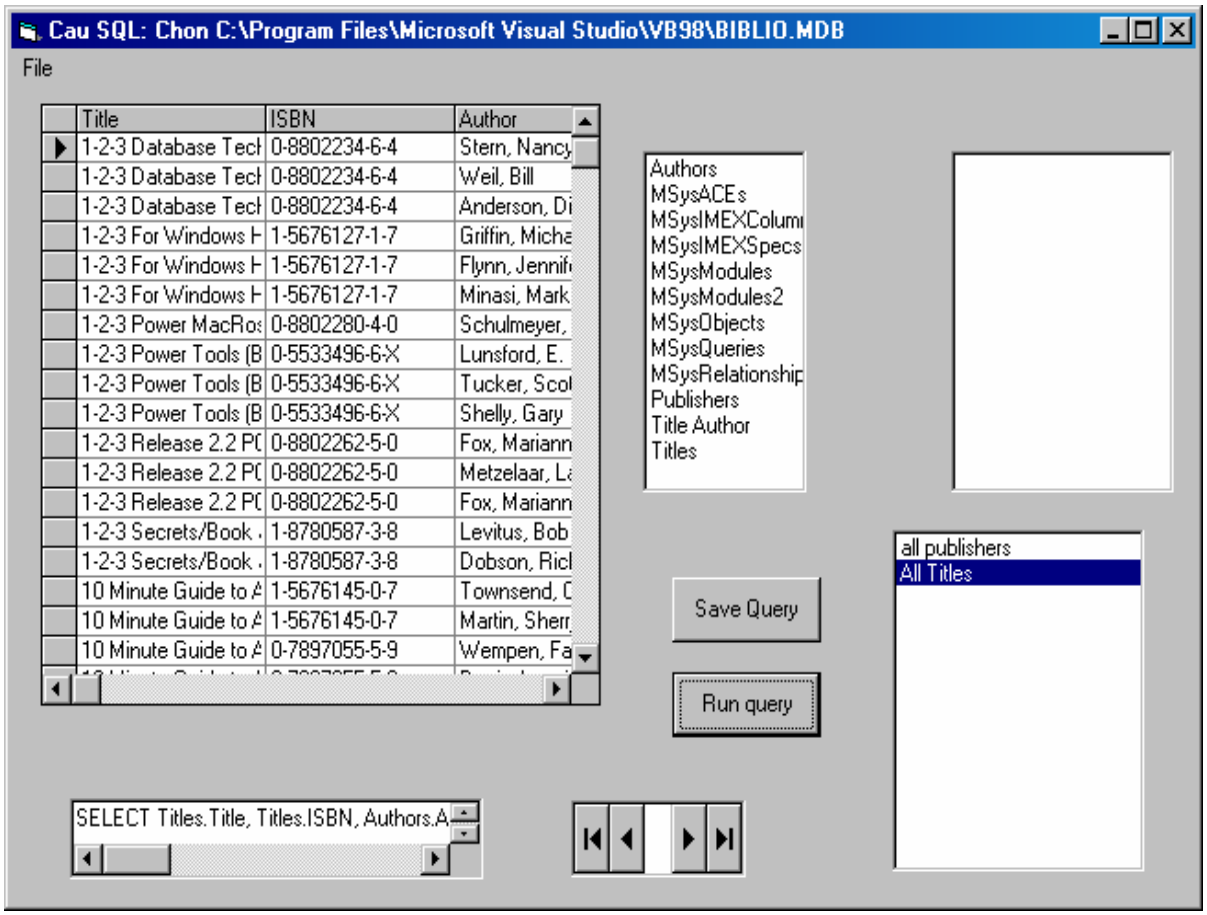
Bước 12: Chúng ta lưu câu SQL nhập từ bàn phím vào trong CSDL trên với một tên cho trước. Ý tưởng chính là ta kiểm tra câu SQL được nhập đó, nếu nó không có lỗi ta sẽ lưu vào CSDL.

Thêm một nút nhấn (Button) vào Form1 với Name: Command2, Caption: Save Query. Sau đó xử lý sự kiện Command2_Click như sau:

```
Private Sub Command2_Click()
        ' Luu cau SQL
        Set qd = New QueryDef
        qd.SQL = Trim$(Text1.Text)
        MsgBox "Cau SQL duoc luu la: " & qd.SQL
        ' Nhap ten cua cau SQL
        qd.Name = InputBox("Nhap ten cau SQL: ")
        db.QueryDefs.Append qd
End Sub
```

Bước 13: Chạy chương trình, mở BIBLIO.MDB, chọn câu một query, chạy nó (Run query); sau đó nhấp vào nút Save Query để lưu lại với tên ta phải nhập vào từ bàn phím. Để kiểm tra, hãy mở lại tập tin trên (File\Open): câu query trên được hiển thị trong List3.

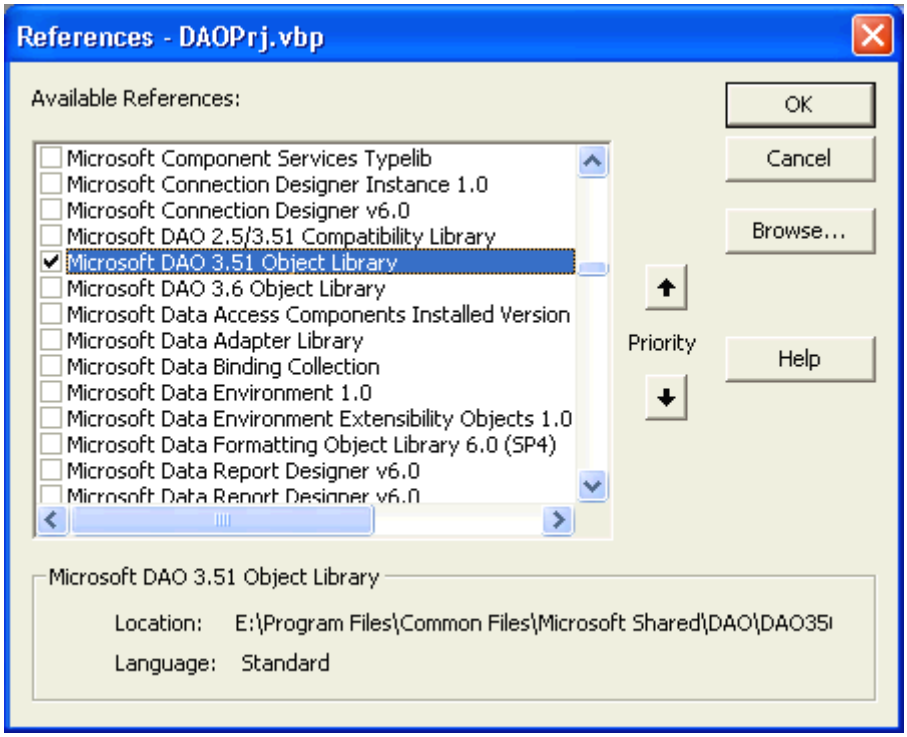
Hình bên dưới hiển thị kết quả khi thực thi chương trình.



Hình IV.10: Kết quả thực thi ứng dụng

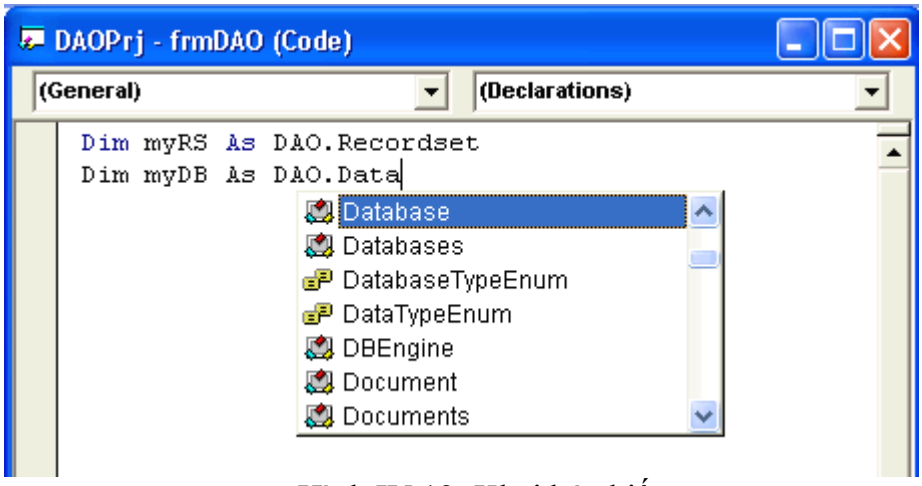
Bài tập 4-3
MÔ HÌNH DAO

Bước 1: Trong bài này ta sẽ tìm hiểu những cách lập trình căn bản với cơ sở dữ liệu MS Access qua kỹ thuật DAO mà không cần dùng đến **Control *Data*** như bài tập 4-1. Ta sẽ cần đến các đối tượng (Object) trong thư viện DAO, do đó nếu bạn mở một dự án VB mới thì hãy dùng Menu Command **Project | References...** để chọn **Microsoft DAO 3.51 Object Library** bằng cách click checkbox bên trái như trong hình dưới đây.



Hình IV.11: Tham chiếu đến thư viện DAO

Bước 2: Sau đó trong cửa sổ soạn thảo mã lệnh của Form chính ta sẽ khai báo biến **myDatabase** kiểu **DAO database** và biến **myRS** cho một **DAO recordset**. Ở đây ta nói rõ Database và Recordset là thuộc loại **DAO** để phân biệt với Database và Recordset thuộc loại **ADO (ActiveX Data Object)** sau này.



Hình IV.12: Khai báo biến

Bước 3: Bây giờ hãy đặt lên Form chính, tên **frmDAO**, 4 labels với captions: **Title, Year Published, ISBN** và **Publisher ID**. Kế đó cho thêm 4 textboxes tương ứng và đặt tên chúng là **txtTitle, txtYearPublished, txtISBN** và **txtPublisherID**.

Điều ta muốn làm là khi Form mới được thực thi, nó sẽ lấy về từ cơ sở dữ liệu một Recordset chứa tất cả records trong **table Titles** theo thứ tự abc của **field** (trường) **Title** và hiển thị record đầu tiên.

DÙNG TỪ KHÓA SET

Bước 4: Trước hết là mở một cơ sở dữ liệu dựa vào tên tập tin của Access database:

```
Set myDB = OpenDatabase(AppFolder & "BIBLIO.MDB")
```

Để ý từ khóa **Set** trong đoạn mã trên. Đó là vì myDB là một **Pointer** (con trỏ) chỉ đến một Object (đối tượng). Mặc dù từ đây về sau ta sẽ dùng myDB như một Database (cơ sở dữ liệu) theo cách giống như bất cứ một biến thuộc kiểu dữ liệu nào khác, nhưng khi chỉ định lần đầu là nó từ đâu đến thì ta dùng chữ Set, để nói rằng thật ra myDB không phải là Object Database, nhưng là Pointer đến Object Database.

Nguyên nhân là VB dành ra một phần trong bộ nhớ (memory) để chứa đối tượng Database khi ta nhận được nó khi hàm **OpenDatabase** thực thi. Dù vị trí chỗ chứa đối tượng Database trong bộ nhớ không nhất định, nhưng vì ta nắm cán chỉ đến vị trí ấy nên ta vẫn có thể làm việc với nó một cách bình thường. Cái cán ấy là trị số của biến myDB. Vì trị số này không phải là Object (đối tượng), nhưng nó chứa **memory address** (địa chỉ trong bộ nhớ) chỉ đến (**point to**) đối tượng Database, nên ta gọi nó là Pointer (con trỏ).

Tương tự như vậy, vì Recordset là một Pointer chỉ đến một đối tượng, ta cũng dùng **Set** khi chỉ định một DAO Recordset lấy về từ hàm **OpenRecordset** của database myDB.

```
Set myRS = myDB.OpenRecordset("Select * from Titles ORDER BY Title")
```

Tham số kiểu String ta dùng cho hàm OpenRecordset là một **câu lệnh SQL**. Nó chỉ định cho cơ sở dữ liệu lấy tất cả mọi trường của mỗi mẩu tin từ Table Titles làm một Recordset và sắp xếp các mẩu tin trong Recordset ấy theo thứ tự abc của trường Title (**ORDER BY Title**).

Để ý là Recordset nầy cũng giống như thuộc tính **Recordset** của một Data Control mà ta dùng trong bài 7-1. Bây giờ có Recordset rồi, ta có thể hiển thị chi tiết của record đầu tiên nếu Recordset ấy có ít nhất một record. Ta kiểm tra điều ấy dựa vào thuộc tính **RecordCount** của Recordset như trong đoạn mã dưới đây của sự kiện Form_Load:

```
Private Sub Form_Load()
  AppFolder = App.Path
  If Right(AppFolder, 1) <> "\" Then AppFolder = AppFolder & "\"
  Set myDB = OpenDatabase(AppFolder & "BIBLIO.MDB")
  Set myRS = myDB.OpenRecordset("Select * from Titles ORDER BY Title")
  If myRS.RecordCount > 0 Then
    myRS.MoveFirst
    Displayrecord
  End If
End Sub
```

Bước 5: Sau khi dùng hàm **MoveFirst** của Recordset để định vị mẩu tin hiện thời là mẩu tin đầu tiên, ta hiển thị trị số các trường của mẩu tin bằng cách gán chúng vào các textboxes của Form như sau:

```
Private Sub Displayrecord()
  With myRS
    txtTitle.Text = .Fields("Title")
    txtYearPublished.Text = .Fields("[Year Published]")
    txtISBN.Text = .Fields("ISBN")
    txtPublisherID.Text = .Fields("PubID")
  End With
End Sub
```

Để ý vì trường **Year Published** gồm có hai từ nên ta phải đặt tên của trường ấy giữa hai dấu ngoặc vuông (**[]**). Để tránh bị phiền phức như trong trường hợp nầy, khi đặt tên các trường tcủa table trong lúc thiết kế cơ sở dữ liệu hãy dán dính các chữ lại với nhau, đừng để rời ra. Thí dụ như dùng **YearPublished** thay vì **Year Published**.

CÁC NÚT DI CHUYỂN

Bước 6: Muốn có các nút Navigators giống như của một Control Data, ta hãy đặt lên Form 4 buttons mang tên **CmdFirst, CmdPrevious, CmNext** và **CmdLast** với captions: <<, <, >, >>.

Bước 7: Mã lệnh cho các nút nầy cũng đơn giản, nhưng ta phải coi chừng khi người dùng muốn di chuyển quá mẩu tin cuối cùng hay mẩu tin đầu tiên. Ta phải kiểm tra xem **EOF** có trở thành True khi người dùng nhấp CmdNext, hay **BOF** có trở thành True khi người dùng nhấp CmdPrevious.

Các sự kiện này được xử lý như sau:

```
Private Sub CmdNext_Click()
  myRS.MoveNext
  If Not myRS.EOF Then
    Displayrecord
  Else
    myRS.MoveLast
  End If
End Sub

Private Sub CmdPrevious_Click()
  myRS.MovePrevious
  If Not myRS.BOF Then
    Displayrecord
  Else
    myRS.MoveFirst
  End If
End Sub

Private Sub CmdFirst_Click()
  myRS.MoveFirst
  Displayrecord
End Sub
```

```
Private Sub CmdLast_Click()
  myRS.MoveLast
  Displayrecord
End Sub
```

Bước 7: Chạy chương trình. Khi chạy chương trình ta sẽ thấy nó hiển thị chi tiết của mẩu tin đầu tiên khác với các bài trước đây vì các mẩu tin đã được sắp xếp.

Ta hãy thử dùng các nút di chuyển <, <<, >, >> xem chúng làm việc có đúng không.

Tới đây, ta nhận thấy rằng dù người dùng có vô tình sửa đổi một chi tiết nào trong các textboxes, không có mẩu tin nào bị cập nhật hóa trong cơ sở dữ liệu khi người dùng di chuyển từ mẩu tin nầy đến mẩu tin khác. Lý do là các Texboxes không có ràng buộc dữ liệu (Data Bound) với các trường của Recordset.

THÊM BỚT CÁC RECORDS

Bước 8: Giống như chương trình trong bài rồi, ta sẽ thêm công cụ để thêm (add), bớt (delete) các mẩu tin. Hãy thêm vào Form 5 buttons tên: **cmdEdit, cmdNew, cmdDelete, cmdUpdate** và **cmdCancel**.

Bước 9: Chỗ nào trong chương trình 4-1 ta dùng **Data1.Recordset** thì bây giờ ta dùng **myRS**.

Ta sẽ dùng lại **Sub SetControls** với tham số **Editing** có trị số False hay True tùy theo người dùng đang xem (Browse) hay sửa đổi (Edit). Trong **Browse mode**, các Textboxes bị Locked (khóa) và các nút **cmdUpdate** và **cmdCancel** bị vô hiệu lực. Trong **Edit mode**, các Textboxes được unlocked (mở khóa) và các nút **cmdNew, cmdDelete** và **cmdEdit** bị vô hiệu lực.

Do đó ta chỉ cần nhớ là khi người dùng đang sửa đổi một mẩu tin hiện hành hay thêm một mẩu tin mới. Ta chứa trị số Boolean ấy trong biến **AddNewRecord**. Nếu user sắp thêm một record mới thì AddNewRecord = True, nếu User sắp Edit một record hiện hữu thì AddNewRecord = False.

Ngoài ra, khi người dùng sắp thêm một mẩu tin mới bằng cách nhấp nút New thì ta phải tự xóa hết các textboxes bằng cách gán chuỗi rỗng cho các TextBox đó.

Ta có các đoạn mã sau:

```
Dim AddNewRecord As Boolean

Private Sub ClearAllFields()
  txtTitle.Text = ""
  txtYearPublished.Text = ""
  txtISBN.Text = ""
  txtPublisherID.Text = ""
End Sub

Private Sub cmdNew_Click()
```

```
    AddNewRecord = True
    ClearAllFields
    SetControls (True)
End Sub

Private Sub CmdEdit_Click()
    SetControls (True)
    AddNewRecord = False
End Sub
```

Bước 10: Khi người dùng nhấp Cancel trong khi đang sửa đổi các textboxes, ta không cần gọi hàm vì Recordset chưa bị đặt vào AddNew hay Edit mode. Ở đây ta chỉ cần hiển thị lại chi tiết của mẩu tin hiện hành, tức là hủy bỏ những gì người dùng đang đánh vào:

```
Private Sub CmdCancel_Click()
    SetControls (False)
    Displayrecord
End Sub
```

Bước 11: Lúc người dùng nhấp Update, ta sẽ kiểm tra dữ liệu xem có trường nào bị bỏ trống (nhất là khóa chính **ISBN** bắt buộc phải có trị số) hay có gì không hợp lệ bằng cách gọi hàm **GoodData**. Nếu GoodData trả lại một trị số False thì ta không xúc tiến với việc Update. Nếu GoodData trả về trị số True thì ta đặt Recordset vào AddNew hay Edit mode tùy theo trị số của biến AddNewRecord là True hay False.

Giống như khi hiển thị chi tiết của một Record ta phải gán từng trường vào textbox, thì bây giờ khi Update ta phải làm ngược lại, tức là gán nội dung của từng textbox vào các trường tương ứng. Sau cùng ta gọi hàm **Update** của recordset và cho các điều khiển trở lại Browse mode:

```
Private Function GoodData() As Boolean
    GoodData = True
End Function

Private Sub CmdUpdate_Click()
    If Not GoodData Then Exit Sub
    With myRS
        If AddNewRecord Then
            .AddNew
        Else
            .Edit
        End If
        .Fields("Title") = txtTitle.Text
        .Fields("[Year Published]") = txtYearPublished.Text
        .Fields("ISBN") = txtISBN.Text
        .Fields("PubID") = txtPublisherID.Text
        .Update
    End With
    SetControls (False)
```

```
End Sub
```

TÌM MỘT RECORD

Bước 11: Tiếp theo đây, ta muốn liệt kê các sách có tiêu đề chứa một chữ hay câu nào đó, thí dụ như chữ "**Guide**". Kế đó người dùng có thể chọn một sách bằng cách chọn tiêu đề sách ấy và nhấp nút **Go**. Chương trình sẽ locate (tìm ra) record của sách ấy và hiển thị chi tiết của nó.

Bây giờ bạn hãy cho vào Form một textbox tên **txtSearch** và một Image tên **ImgSearch**. Kế đó đặt một frame tên **fraSearch** vào Form. Để lên frame nầy một listbox tên **List1** để tựa các sách.

Ta sẽ cho ImgSearch hiển thị hình một ống nhòm nên bạn hãy click vào bên phải **property Picture** trong Properties Window để chọn Icon BINOCULR.ICO từ folder E:\Program Files\Microsoft Visual Studio\Common\Graphics\Icons\Misc.

Khi người dùng nhấp vào ImgSearch, chương trình sẽ tự động tìm kiếm các sách có tựa được người dùng đánh vào trong TextBox. Sự kiện ImgSearch được xử lý như sau:

```
Private Sub ImgSearch_Click()
  fraSearch.Visible = True
  Dim SrchRS As DAO.Recordset
  Dim SQLCommand As String
  SQLCommand = "Select * from Titles where Title LIKE '" & "*" & txtSearch & "*"
& "' ORDER BY Title"
  Set SrchRS = myDB.OpenRecordset(SQLCommand)
  If SrchRS.RecordCount > 0 Then
    List1.Clear
    With SrchRS
      Do While Not SrchRS.EOF
        List1.AddItem .Fields("Title")
        .MoveNext
      Loop
    End With
  End If
End Sub
```

Trong câu SELECT trên ta dùng toán tử **LIKE**, nội dung của TextBox, có dấu * ở hai bên. Dấu * là chỗ có (hay không có) chữ gì cũng được.

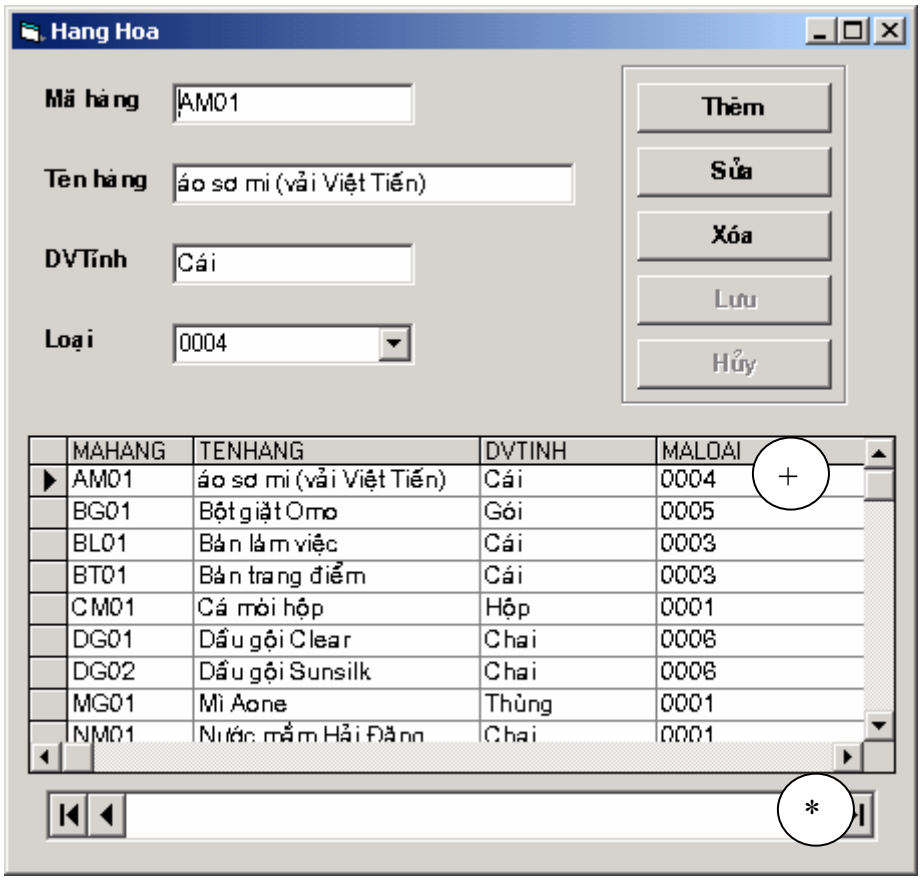
Bước 12: Lưu dự án và chạy chương trình. Kiểm tra kết quả.

Bài tập 4-4

THÍ DỤ VỀ SỬ DỤNG DAO

Bài tập này nhằm mục đích giới thiệu về cách thức sử dụng điều khiển dữ liệu và thư viện DAO trong việc thiết kế một Form nhập liệu hoàn chỉnh cho bảng THangHoa trong CSDL HangHoa.mdb. Giao diện cùng với mã lệnh chỉ mang tính chất gợi ý; sinh viên có thể thực hiện theo ý riêng của mình.

Bước 1: Thiết kế form như sau:



Hình IV.13: Sử dụng Data Control

Với *: DataControl: Điều khiển dữ liệu

DatabaseName: HangHoa.MDB
RecordSource: THangHoa
Name: datHH

Bước 2: Thiết lập các thuộc tính cho các TextBox & ComboBox.

DataSource: DatHH
DataField:

Bước 3: Đặt điều khiển lưới lên Form (+) sau khi đã tham chiếu đến nó (Project\Components\Microsoft Data Bound Grid Control 5.0 SP3).

DataSource: datHH

Chạy chương trình, ta được kết quả như trên.

Bước 4: Sự kiện cmdThem_Click

```
Private Sub cmdThem_Click()
   datHH.Recordset.AddNew
   DoiTThai False
End Sub
```

Trong đó:

```
Private Sub DoiTThai(ByVal TThai As Boolean)
   cmdThem.Enabled = TThai
   cmdSua.Enabled = TThai
   cmdXoa.Enabled = TThai
   cmdHuy.Enabled = Not TThai
   cmdLuu.Enabled = Not TThai
End Sub
```

Bước 4: Sự kiện cmdSua_Click & sự kiện cmdXoa_Click:

```
Private Sub cmdSua_Click()
   datHH.Recordset.Edit
   DoiTThai False
End Sub

Private Sub cmdXoa_Click()
   On Error GoTo Xuly
   datHH.Recordset.Delete
   Exit Sub

Xuly:
   MsgBox Err.Description, vbCritical + vbSystemModal, "Loi"
End Sub
```

Bước 5: Sự kiện cmdLuu_Click & cmdHuy_Click

```
Private Sub cmdHuy_Click()
   'dathh.Database =
   datHH.Recordset.CancelUpdate
   DoiTThai True
End Sub

Private Sub cmdLuu_Click()
   datHH.Recordset.Update
   DoiTThai True
End Sub
```

Bước 6: Sự kiện Form_Load:

```
Private Sub Form_Load()
   DoiTThai True
End Sub
```

- Có nhận xét gì nếu khi chạy chương trình, ta thêm mới một mẩu tin có khóa là Mahang trùng với một MaHang đã có. Để giải quyết ta làm thế nào?

II. BÀI TẬP TỰ LÀM

Sử dụng CSDL HangHoa.mdb, anh (chị) hãy:

1) Cải tiến Form nhập ở bài 4-4, sao cho trường *MaLoai* phải được lấy từ các *MaLoai* của bảng TLoaiHang. Hơn nữa thay vì hiển thị *MaLoai*, ta hiển thị *TenLoai* cho dễ theo dõi; nhưng khi thêm vào bảng THangHoa, ta lại thêm vào *MaLoai* của *TenLoai* đó.

2) Bằng cách sử dụng DAO (tương tự 4-3), anh (chị) hãy:
   a. Thiết kế Form nhập liệu cho bảng THANGHOA.
   b. Thiết kế Form nhập liệu cho bảng TNHANVIEN.

Chương 5 ĐỐI TƯỢNG DỮ LIỆU TỪ XA

Mục tiêu:

Chương này nhằm giới thiệu cho sinh viên cách thức sử dụng thư viện đối tượng RDO để truy cập dữ liệu, cách thức truy cập dữ liệu trước đây đối với cơ sở dữ liệu ở máy từ xa.

Học xong chương này, sinh viên phải nắm được các vấn đề sau:

- Thiết lập tên nguồn dữ liệu (Data Source Name) trên Windows.
- Sử dụng Remote Data Control để truy cập dữ liệu.
- Đối tượng UserConnection cũng như thư viện đối tượng RDO.

Kiến thức có liên quan:

Giáo trình Visual Basic, Chương 10.

Tài liệu tham khảo:

- **Visual Basic 6 Certification Exam Guide** - Chapter 11, Page 309 - **Dan Mezick & Scot Hillier** - **McGraw-Hill** - **1998**.
- **Microsoft Visual Basic 6.0 & Lập trình cơ sở dữ liệu – Chương 23, trang 735** - **Nguyễn Thị Ngọc Mai** (chủ biên) - **Nhà xuất bản Giáo dục** - **2000**.
- **Tự học Lập trình cơ sở dữ liệu với Visual Basic 6 trong 21 ngày** (T2) - Chương 17, trang 227 - **Nguyễn Đình Tê** (chủ biên), **Nhà xuất bản Giáo dục** - **2001**.

I. HƯỚNG DẪN

Bài tập 5-1

ĐỐI TƯỢNG USER CONNECTION

THIẾT LẬP NGUỒN DỮ LIỆU ODBC

Bước 1: Trong bài tập này, ta sẽ thao tác với CSDL Access BIBLIO.MDB thông qua VB.

Trước tiên ta phải thiết lập nguồn dữ liệu nhờ tiện ích ODBC Administrator của Windows. Chọn ODBC Administrator trong Control Panel (icon tên là ODBC). Nhấp đúp vào icon này để khởi động ODBC Administrator.

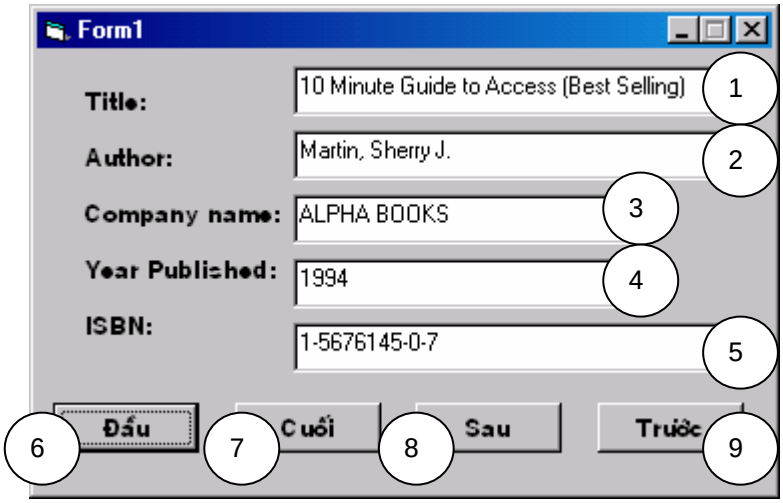
Bước 2: Một hộp thoại hiện ra hiển thị tất cả các nguồn dữ liệu ODBC hiện thời trên máy tính. Để tạo một nguồn dữ liệu mới, ta chọn mục **System DSN**. Nhấp vào nút **Add**. Hộp thoại cho phép tạo nguồn dữ liệu mới với tất cả các điều khiển (drivers) ODBC có sẵn trên máy. Ở đây, ta chọn **Microsoft Access ODBC driver** trong danh sách và chọn **Finish**.

Bước 3: Hộp thoại cài đặt ODBC xuất hiện. Trong hộp thoại này, định nghĩa một nguồn dữ liệu mới. Nhưng trước tiên ta phải xác định tên của nguồn dữ liệu. Ta có thể đặt với bất cứ một tên nào, chẳng hạn BIBLIO.

Sau đó, nhấp nút **Select** để tìm đến CSDL BIBLIO.MDB trong máy tính, chọn tập tin này; nhấp OK. Lúc này nguồn dữ liệu đã được định nghĩa xong.

TẠO GIAO DIỆN CHƯƠNG TRÌNH

Bước 4: Tạo một dự án mới trong thư mục Basic\Bt5-1. Tạo giao diện chương trình như hình dưới:



Hình V.1: Giao diện

1: TextBox
Name: txtTitle
2: TextBox
Name: txtAuthor
3: TextBox

Name: txtName
4: TextBox
Name: txtYear
5: TextBox
Name: txtISBN
6: CommandButton
Name: cmdDau
Caption: Đầu
7: CommandButton
Name: cmdCuoi
Caption: Cuối
8: CommandButton
Name: cmdSau
Caption: Sau
9: CommandButton
Name: cmdTruoc
Caption: Trước

ĐỐI TƯỢNG USER CONNECTION

Bước 5: Ở đây, ta sử dụng đối tượng **User Connection** để truy xuất RDO. Chọn **Project\Components…** trên menu; sau đó chọn mục **Designers**. Trong mục này, đánh dấu check vào **Microsoft UserConnection**. Sau đó ta thêm UserConnection vào dự án bằng cách chọn **Project\More ActiveX Designers\Microsoft UserConnection** trên menu.

Bước 6: Đối tượng UserConnection cung cấp cho ta giao diện đồ họa để truy xuất đến nối kết kiểu RDO và làm tiện lợi hơn khi xác định nối kết ODBC, nhất là truy xuất đến các câu truy vấn hay các thủ tục lưu trữ sẵn trong CSDL mà không cần đến một đoạn mã nào.

Bước 7: Trong cửa sổ thuộc tính của UserConnection, liên kết với nguồn dữ liệu BIBLIO ODBC vừa định nghĩa ở trên. Khi nối kết được thiết lập, đóng cửa sổ Properties lại.

Bước 8: Nhấp chuột phải vào icon UserConnection, một menu hiện ra, chọn **Insert Query** trên menu để thêm một câu truy vấn vào UserConnection. Hộp thoại tạo truy vấn mới xuất hiện cho phép định nghĩa câu truy vấn mới. Đặt tên cho câu truy vấn này là AllTiles.

Chọn tùy chọn **Base on User-Defined SQL** và thêm câu SQL sau vào cửa sổ soạn thảo:

```
SELECT Titles.Title, Titles.ISBN, Authors.Author, Titles.[Year Published],
    Publishers.[Company Name]
FROM Publishers, Titles, Authors, [title author]
WHERE ((((Authors.Au_ID = [title author].Au_ID)
      AND ([title author].ISBN = Titles.ISBN))
      AND (Titles.PubID = Publishers.PubID)))
```

```
ORDER BY Titles.Title
```

Bước 9: Đóng cửa sổ thiết kế câu truy vấn để lưu vào UserConnection.

VIẾT MÃ LỆNH CHO ỨNG DỤNG

Bước 10: Mở cửa sổ Code của Form1, trong phần [General]\[Declarations], định nghĩa các biến sau cho đối tượng nối kết RDO và ResultSet.

```
Private m_noiket As UserConnection1
Private m_ketqua As RDO.rdoResultset
```

Bước 11: Ta sẽ tạo nối kết trong sự kiện Form_Load, sau khi nối kết được tạo, ta chạy câu SQL vừa định nghĩa trong UserConnection. Thêm đoạn mã sau vào sự kiện Form_Load để truy xuất đến nguồn dữ liệu:

```
Set m_noiket = New UserConnection1
m_noiket.EstablishConnection
' Lay ket qua
Set m_ketqua = _
        m_noiket.rdoQueries("AllTitles").OpenResultSet(rdOpenDynamic)
' Dien vao Form
Call Hienthi
```

Thủ tục Hienthi dùng để hiển thị thông tin của các trường tương ứng của mỗi mẩu tin lên TextBox, thủ tục này được viết như sau:

```
Public Sub Hienthi()
   txtTitle.Text = m_ketqua!Title
   txtAuthor.Text = m_ketqua!Author
   txtName.Text = m_ketqua![Company Name]
   txtYear.Text = m_ketqua![Year Published]
   txtISBN.Text = m_ketqua!ISBN
End Sub
```

Bước 12: Phần còn lại của chương trình của ta là thêm phần xử lý sự kiện Click cho các nút nhấn. Mã lệnh cho các sự kiện này như sau:

```
Private Sub cmdCuoi_Click()
   m_ketqua.MoveLast
   Call Hienthi
End Sub
```

```
Private Sub cmdDau_Click()
   m_ketqua.MoveFirst
   Call Hienthi
End Sub

Private Sub cmdSau_Click()
   m_ketqua.MoveNext
   If m_ketqua.EOF Then
      Beep
      m_ketqua.MoveLast
   Else
      Call Hienthi
   End If
End Sub

Private Sub cmdTruoc_Click()
   m_ketqua.MovePrevious
   If m_ketqua.BOF Then
      Beep
      m_ketqua.MoveFirst
   Else
      Call Hienthi
   End If
End Sub
```

Bước 13: Lưu dự án và chạy chương trình.

Bài 5-2

SỬ DỤNG REMOTE DATA CONTROL

Bước 1: Tạo dự án mới trong thư mục Basic\Bt5-2. Đăng ký một DSN với tên là DBHH.

Bước 2: Trong bài tập này ta sử dụng Remote Data Control và lưới hiển thị dữ liệu, do đó ta tham chiếu đến các thành phần này bằng cách chọn Project\Components…, thiết lập tham chiếu đến Microsoft Remote Data Control và Microsoft Data Bound Grid Control. Nhấp OK.

Bước 3: Thiết kế Form có dạng sau:

Hình V.2: Sử dụng RDC

1: RemoteDataControl. Name: rdcHangHoa

2: DBGrid. Name: dbgHangHoa.

Bước 4: Đặt thuộc tính DataSourceName của điều khiển rdcHangHoa là DBHH (DSN đã tạo trước đây).

Bước 5: Định thuộc tính SQL của điều khiển rdcHangHoa là:

```
Select * From THANGHOA
```

Bước 6: Chỉ định thuộc tính DataSource của điều khiển dbgHangHoa là rdcHangHoa.

Bước 7: Thực thi chương trình.

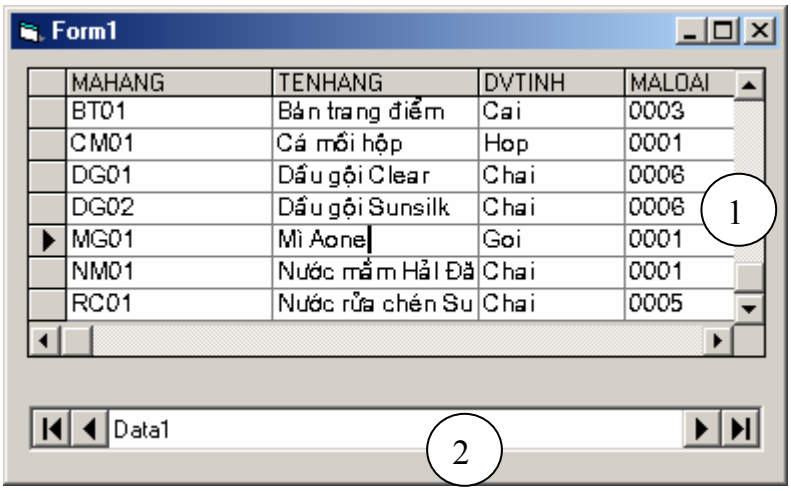
Bài 5-3

ODBC DIRECT

Bước 1: Tạo dự án mới lưu trong Basic\Bt5-3.

Bước 2: Tham chiếu đến điều khiển lưới Microsoft Data Bound Grid Control trong mục Project\Components.

Bước 3: Tạo Form có dạng sau:



Hình V.3: Sử dụng ODBC Direct

1: DBGrid. Name: DbGrid1.

2: Data Control. Name: Data1.

Bước 4: Đổi thuộc tính DefaultType của DataControl là 1 - Use ODBC.

Bước 5: Thuộc tính Connect của Data1 là: ODBC;DSN=DBHH.

Bước 6: Đặt thuộc tính Record Source của Data Control:

Select * From THANGHOA.

Bước 7: Đặt thuộc tính Data Source của DBGrid1 là: Data1.

Bước 8: Lưu dự án và thực thi chương trình.

II. BÀI TẬP TỰ LÀM

Sử dụng CSDL HangHoa.mdb và thư viện đối tượng RDO, anh (chị) hãy:

1) Thiết kế Form nhập liệu cho bảng THANGHOA.
2) Thiết kế Form nhập liệu cho bảng TNHANVIEN.

Chương 6 ĐỐI TƯỢNG DỮ LIỆU ACTIVEX

Mục tiêu:

Chương này gồm các bài tập nhằn rèn luyện cho sinh viên cách thức lập trình cơ sở dữ liệu bằng cách sử dụng thư viện đối tượng ADO, giao diện lập trình phổ biến hiện nay.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

Các thành phần trong mô hình đối tượng ADO gồm có:

- Đối tượng Connection.
- Đối tượng Recordset.
- Đối tượng Command.
- Đối tượng Field.

Cách thức sử dụng các đối tượng này trong ứng dụng viết bằng VB.

Kiến thức có liên quan:

- Giáo trình Visual Basic - Chương 11

Tài liệu tham khảo:

Visual Basic 6 Certification Exam Guide - **Dan Mezick & Scot Hillier** - Chapter 12, Page 345 - **McGraw-Hill** - **1998**.

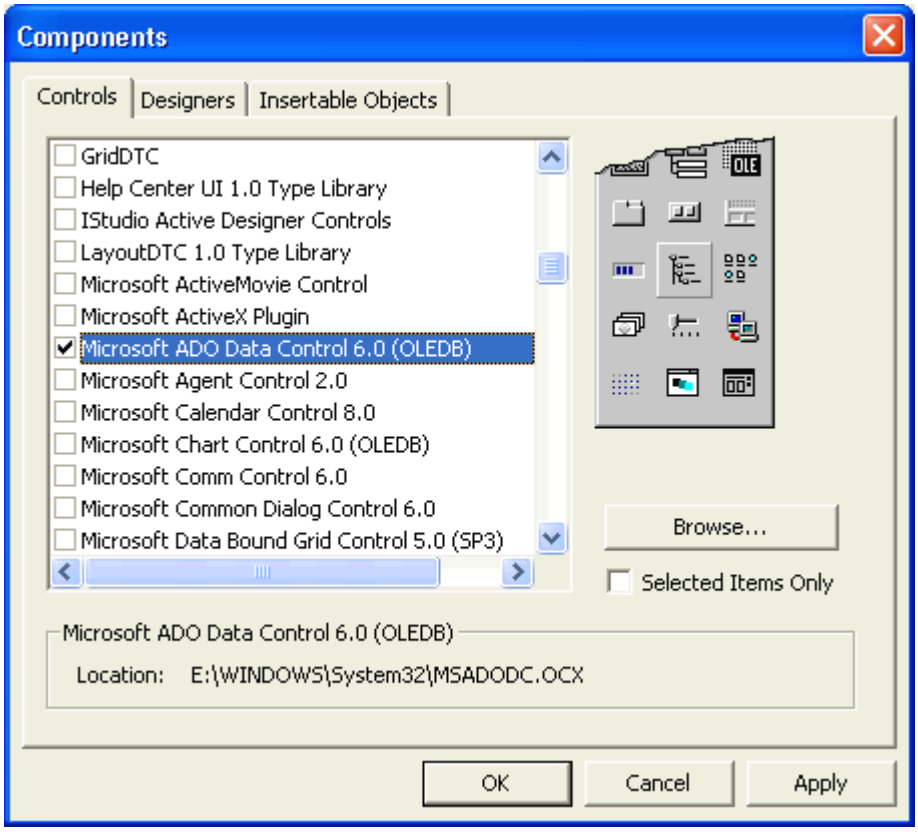
http://www.vovisoft.com/VisualBasic/VB6Chapter16.htm

I. HƯỚNG DẪN

Bài tập 6-1

SỬ DỤNG ADO DATA CONTROL

Bước 1: Khởi động một dự án VB6 mới, thêm điều khiển ***Data ADO*** vào hộp ToolBox nhờ chọn Menu Command **Project | Components...**, rồi **Microsoft ADO Data Control 6.0 (OLEDB)** như dưới đây:



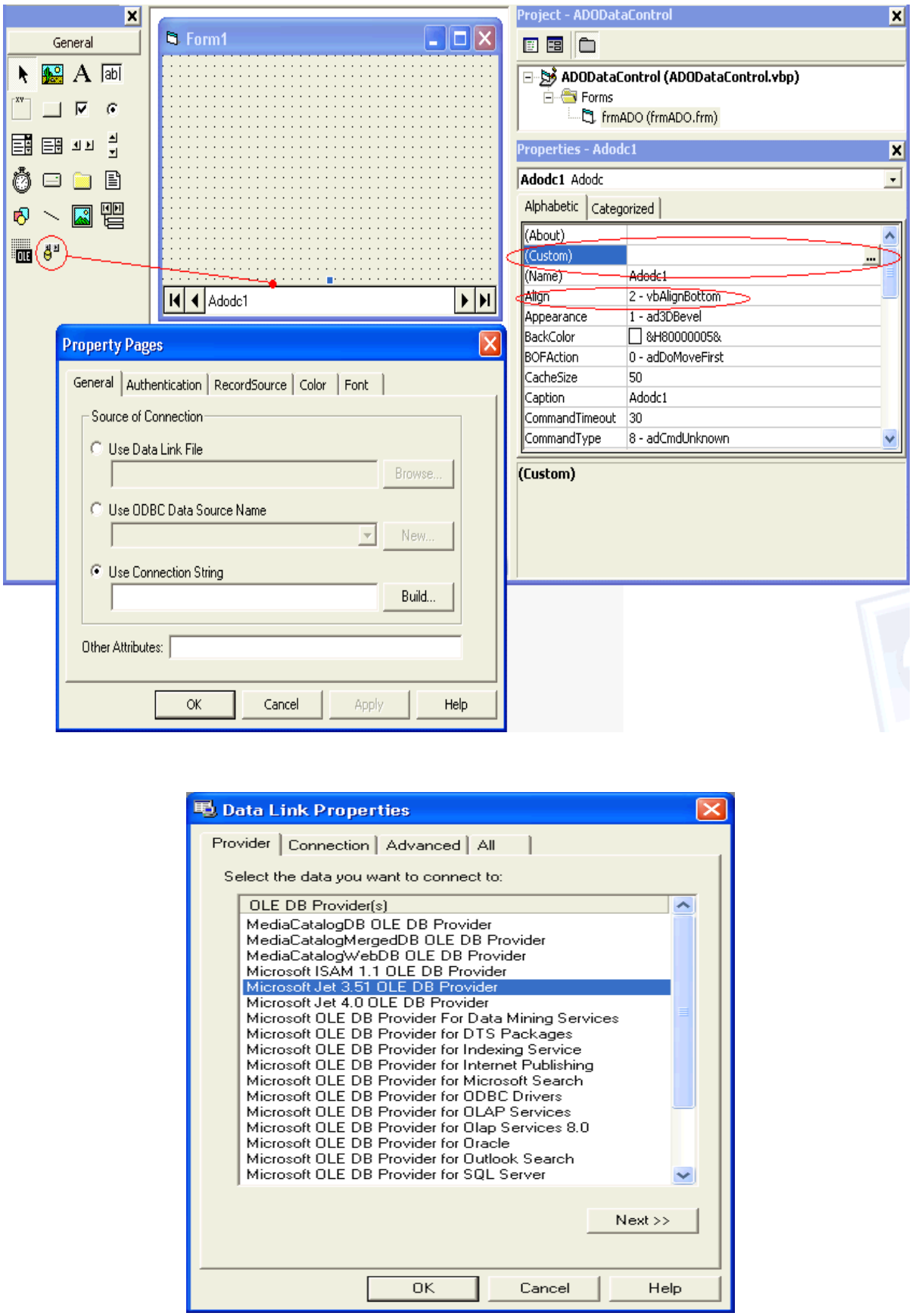
Hình VI.1: Tham chiếu thư viện ADO

Bước 2: Đặt tên dự án là **ADODataControl**. Sửa thuộc tính *Name* của form chính thành **frmADO**, *Caption:* **ADO DataControl Demo**.

Đặt một Control Data ADO tên **Adodc1** lên Form. Muốn cho nó nằm bên dưới Form, thiết lập thuộc tính **Align** của nó trong cửa sổ Properties thành **2 - vbAlignBottom**.

Bước 3: Nhấp bên phải hàng **property (Custom)**, kế đó click lên nút browse có ba chấm để hộp thoại **Property Pages** hiện ra. Trong hộp thoại nầy, trên **Tab General** chọn Radio (Option) Button **Use Connection String** rồi nhấp nút **Build...**.

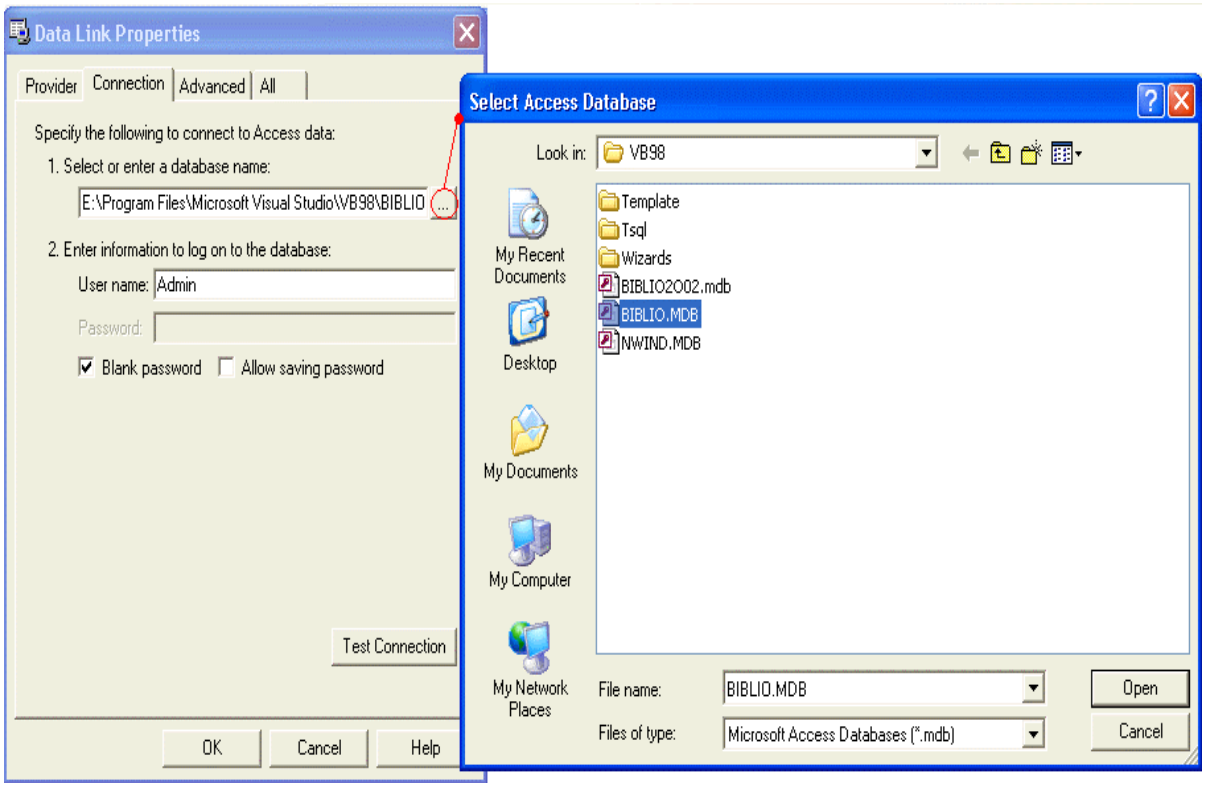
Trong hộp thoại **Data Link Properties**, **Tab Provider**, chọn **Microsoft Jet 3.51 OLE DB Provider**, rồi click nút **Next >>** hay **Tab Connection**.



Hình VI.2: Thiết lập Connection

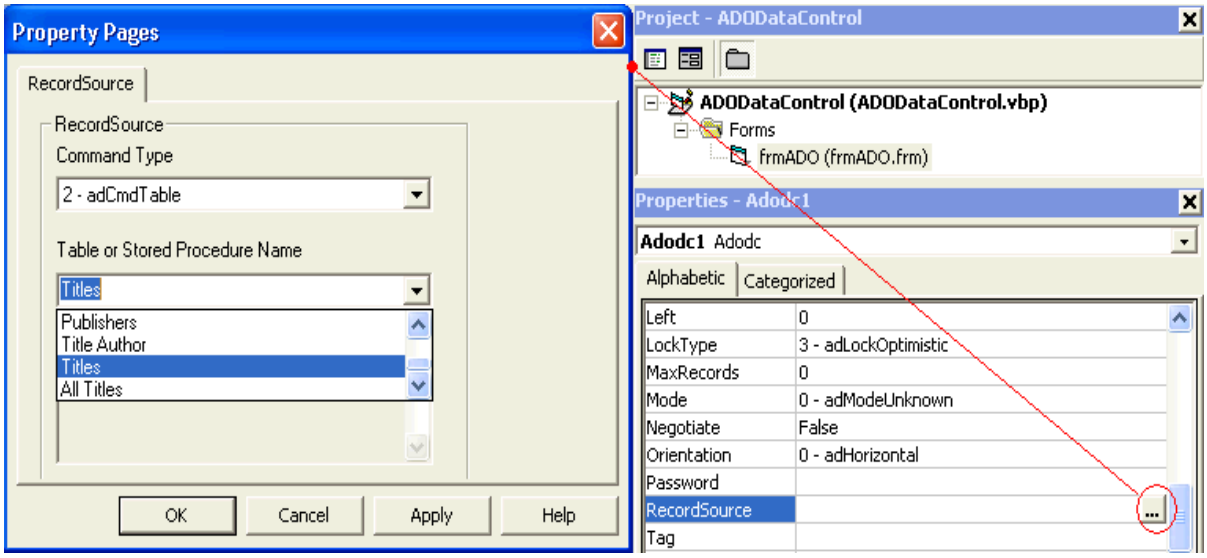
Mục **Select or enter a database name** ta chọn **E:\Program Files\Microsoft Visual Studio\VB98\BIBLIO.MDB**, trên máy tính khác có thể nằm trên ổ **C** hay **D**.

Sau đó, click nút **Test Connection** phía dưới để thử xem connection có được thiết lập tốt không.



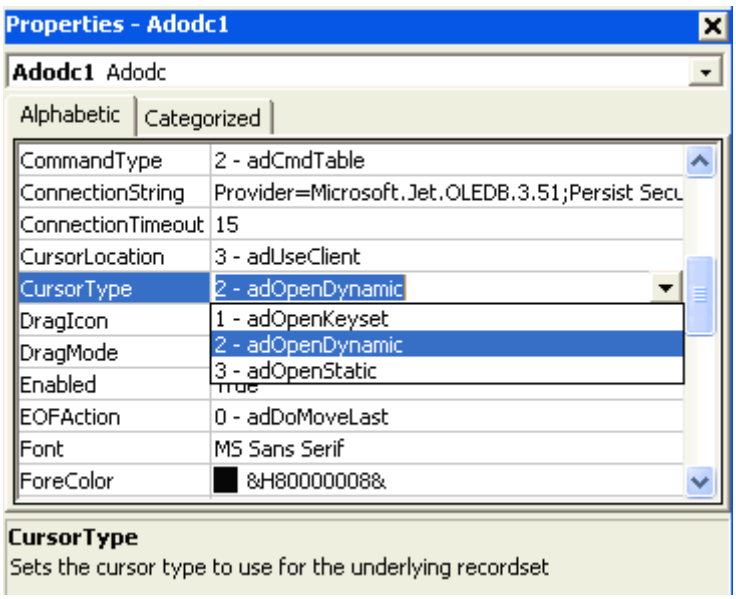
Hình VI.3: Lựa chọn CSDL

Bước 4: Lập connection xong rồi, ta chỉ định muốn lấy gì về làm Recordset bằng cách chọn thuộc tính **Recordsource** của Adodc1. Trong giao diện Property Pages của nó chọn **2-adCmdTable** làm **Command Type**, kế đó mở Combo box cho **Table or Stored Procedure Name** để chọn **table Titles**.



Hình VI.4: RecordSource

Chọn trị số **2-adOpenDynamic** cho thuộc tính Cursor Type của Adodc1:

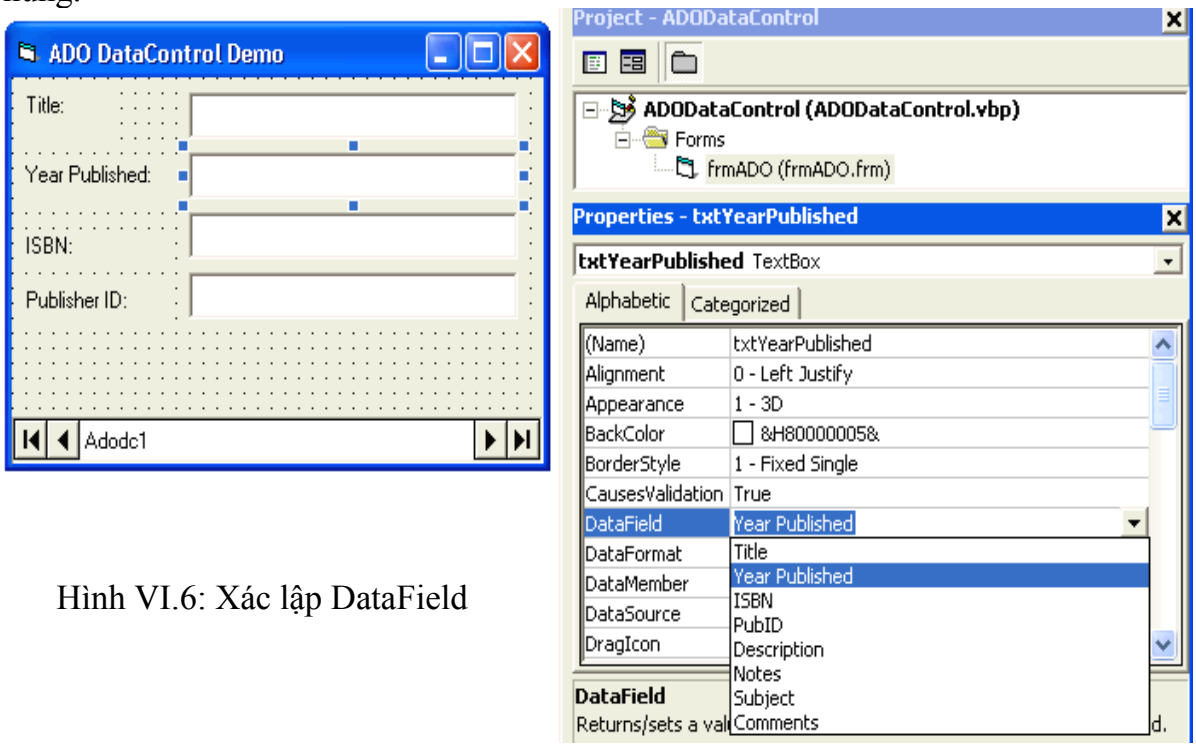


Hình VI.5: CursorType

Bước 5: Bây giờ đặt lên Form 4 điều khiển nhãn với captions: **Title, Year Published, ISBN** và **Publisher ID**. Kế đó cho thêm 4 textboxes tương ứng và đặt tên chúng là **txtTitle, txtYearPublished, txtISBN** và **txtPublisherID**.

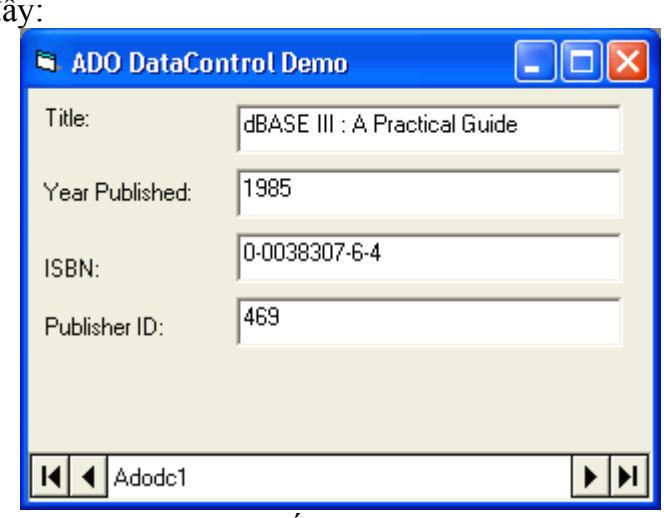
Để thực hiện Data Binding, chọn textbox **txtYearPublished** (năm xuất bản), rồi thiết lập thuộc tính **Datasource** của nó thành **Adodc1**. Khi chọn thuộc tính **DataField** của txtYearPublished và mở ComboBox ra ta sẽ thấy liệt kê tên các Fields trong table Titles. Đó là vì Adodc1 được coi như trung gian lấy table Titles từ database. Ở đây ta sẽ chọn cột Year Published.

Lặp lại công việc này cho 3 textboxes kia, và chọn các cột Title (Tiêu đề), ISBN (số lý lịch trong thư viện quốc tế), và PubID (số lý lịch nhà xuất bản) làm DataField cho chúng.



Hình VI.6: Xác lập DataField

Đến đây, mặc dù chưa viết một dòng lệnh nào, bạn có thể chạy chương trình và nó sẽ hiển thị như dưới đây:



Hình VI.7: Kết quả ứng dụng

Bài tập 6-2

TẠO TẬP TIN LIÊN KẾT DỮ LIỆU (DATALINK FILE)

Tập tin liên kết dữ liệu là tập tin Windows dùng để chứa các thông tin về chuỗi kết nối đến một CSDL nào đó. Chúng ta sẽ sử dụng tập tin này trong việc thiết lập đối tượng kết nối (Connection) đến CSDL.

Bước 1: DataLink File được tạo trong cửa sổ Windows Explorer. Do đó khởi động Windows Explorer, vào thư mục \Program Files\Common Files\System\Ole DB\Data Links. Đây là nơi thường chứa các tập tin DataLink.
Bước 2: Chọn New\Microsoft Data Link từ Windows Explorer.
Bước 3: Đổi tên tập tin này thành Biblio.UDL.
Bước 4: Nhấp chuột phải lên tập tin mới này và chọn **PROPERTIES** từ menu.
Bước 5: Nhấp chuột chọn mục **PROVIDER** & chọn **Microsoft JET 3.51 OLEDB Provider**. Nhấp Next.
Bước 6: Chọn đường dẫn đến tập tin BIBLIO.MDB (C:\Program Files\Microsoft Visual Studic\VB98\Biblio.mdb) trong mục **Select or Enter Database Name**.
Bước 7: Nhấp mục **Test Connection** để kiểm tra kết nối có thành công hay không?
Bước 8: Chọn OK để lưu lại kết nối dữ liệu này.

Bài tập 6-3

ĐỐI TƯỢNG ADO RECORDSET

Bước 1: Trước khi bắt đầu, cần kiểm tra tập tin DataLink của bài 6-2 có được tạo ra hay là không?
Bước 2: Tạo một dữ án mới trong VB ở thư mục Basic\Bt6-3.

Bước 3: Tham chiếu đến thư viện ADO bằng cách chọn Project\References\ActiveX Data Object 2.0 Library. Chọn OK.
Bước 4: Đặt một điều khiển ListBox lên Form (Name: lstName)
Bước 4: Xử lý sự kiện Form_Load như sau:

```
Set m_Connection = New ADODB.Connection
m_Connection.ConnectionString = _
      "File Name=C:\Program Files\Common Files\System\Ole DB\Biblio.udl"
m_Connection.Open

Set m_RecordSet = New ADODB.Recordset
m_RecordSet.Open "Select Name FROM Publishers", m_Connection
Do While Not m_RecordSet.EOF
      lstName.AddItem m_RecordSet!Name
      m_RecordSet.MoveNext
Loop
```

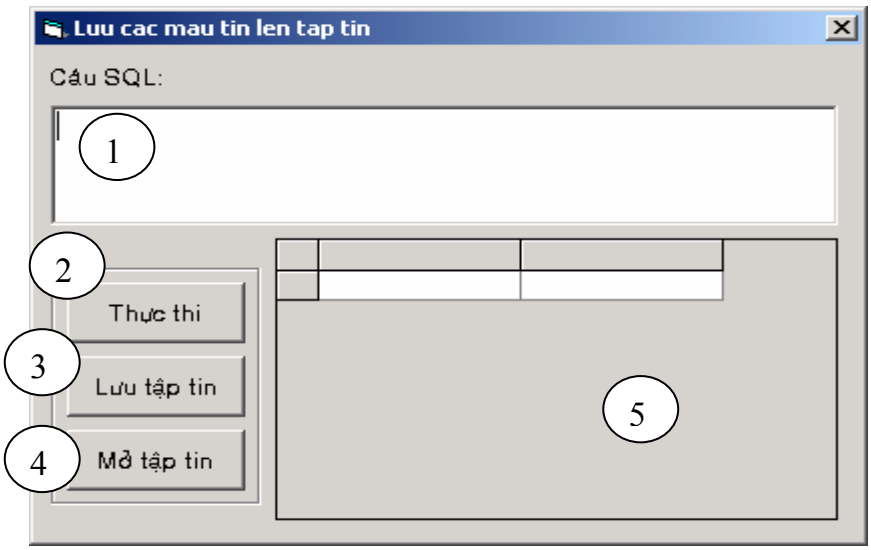
Bước 5: Chạy chương trình, điều gì xảy ra?
Đoạn mã trên là một ví dụ điển hình về việc sử dụng phương thức Open để nhận về một tập hợp các Records từ cơ sở dữ liệu.

Bước 6: Thay vì khởi tạo đối tượng *Connection* nhờ đặt thuộc tính *ConnectionString* như trên (dùng *tập tin DataLink*), hãy khởi tạo đối tượng này như trong phần mô tả ở lý thuyết (sử dụng thuộc tính *Provider & ConnectionString*). Sau đó chạy chương trình. Nhận xét.

Bài tập 6-4

LƯU RECORSET RA TẬP TIN

Bước 1: Trước khi bắt đầu, cần kiểm tra tập tin DataLink của bài 6-2 có được tạo ra hay là không?
Bước 2: Tạo một dự án mới trong VB ở thư mục Basic\Bt6-4.
Bước 3: Tham chiếu đến thư viện ADO bằng cách chọn Project\References\ActiveX Data Object 2.0 Library. Chọn OK.
Bước 4: Thêm một điều khiển mở rộng vào hộp công cụ Toolbox nhờ chọn Project\Components. Chọn Microsoft Common Dialog Control 6.0 và Microsoft DataGrid Control 6.0. Chọn OK.
Bước 5: Tạo giao diện như sau:



Hình VI.8: Lưu Recordset ra tập tin

1: TextBox:
Name: txtSQL; MultiLine: True.
2: CommandButton:
Name: cmdTT; Caption: Thực thi.
3: CommandButton:
Name: cmdLuu; Caption: Lưu tập tin
4: CommandButton:
Name: cmdMo; Caption: Mở tập tin
5: DataGrid: Name: grdResult

Ngoài ra, chương trình còn có sử dụng hộp thoại mở & lưu tập tin; do đó, ta thêm vào một Common Dialog vào ứng dụng với thuộc tính Name: dlgFile.

Bước 6: Khi chương trình thực thi, mỗi khi người dùng nhập vào một câu lệnh SQL vào TextBox rồi nhấp chọn *Thực thi*, câu SQL này sẽ thực thi và hiển thị kết quả ở lưới bên phải. Do đó sự kiện cmdTT_Click dược xử lý như sau:

```
Private Sub cmdTT_Click()
   Dim m_RecordSet As ADODB.Recordset
   Set m_RecordSet = New ADODB.Recordset
   m_RecordSet.CursorLocation = adUseClient
   m_RecordSet.CursorType = adOpenStatic
   m_RecordSet.Open txtSQL.Text, _
      "File Name=C:\Program Files\Common Files\System\OLE DB\Biblio.udl"
   Set grdResult.DataSource = m_RecordSet
End Sub
```

Bước 7: Lưu RecordSet vào tập tin sẽ được thực hiện nhờ hàm Save. Ở đây, ta sử dụng hộp thoại CommonDialog để mở và lưu tập tin. Sự kiện cmdLuu_Click được xử lý:

```
Private Sub cmdLuu_Click()
   On Error GoTo xuly
   Dim m_RecordSet As ADODB.Recordset
   Set m_RecordSet = grdResult.DataSource
```

```
   Set grdResult.DataSource = Nothing

   Dim strFileName As String
   dlgFile.Filter = "Record Set Files (*.dat)|*.dat"
   dlgFile.ShowSave
   strFileName = dlgFile.FileName

   ' Luu cac mau tin
   m_RecordSet.Save strFileName
   Exit Sub

xuly:
   MsgBox Err.Description, vbCritical + vbSystemModal, "Loi"
End Sub
```

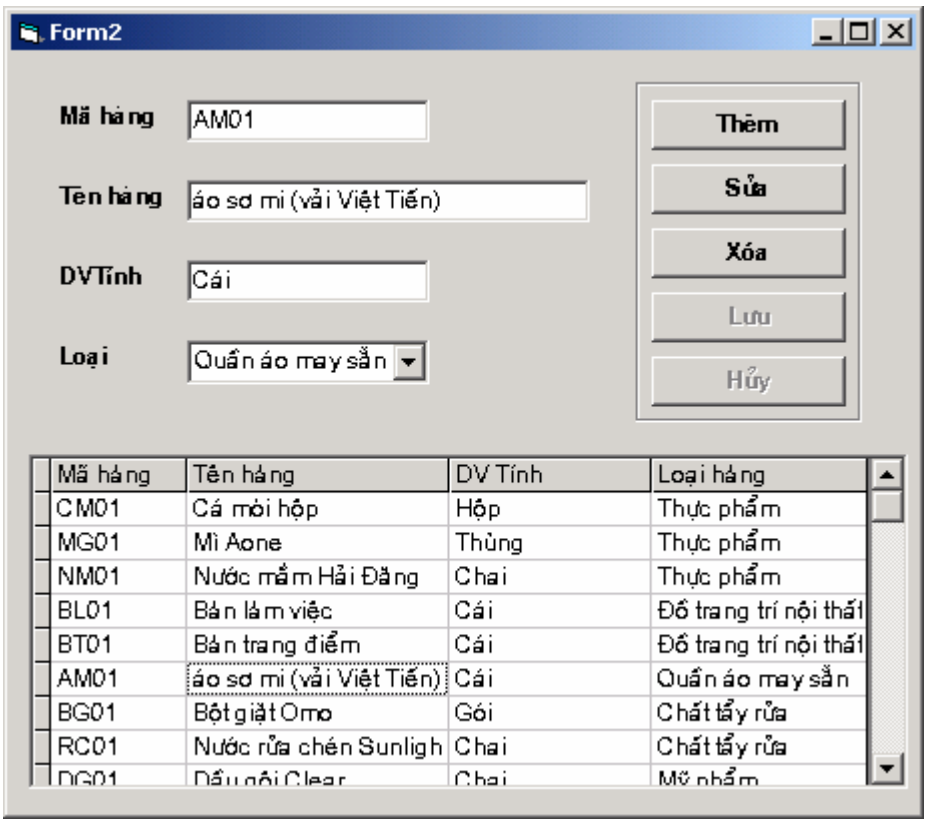
Bước 8: Mỗi khi tập hợp mẩu tin được lưu lên tập tin, chúng sẽ không phụ thuộc vào vào các nối kết với nguồn dữ liệu. Để mở dữ liệu được lưu, sử dụng hàm Open với tên tập tin là đối số. Sự kiện cmdOpen_Click được xử lý:

```
Private Sub cmdMo_Click()
   On Error GoTo xuly
   Dim strFileName As String
   dlgFile.Filter = "Record Set Files (*.dat)|*.dat"
   dlgFile.ShowOpen
   strFileName = dlgFile.FileName
   Dim m_RecordSet As ADODB.Recordset
   Set m_RecordSet = New ADODB.Recordset
   m_RecordSet.Open strFileName
   Set grdResult.DataSource = m_RecordSet
   Exit Sub
xuly:
   MsgBox Err.Description, vbCritical + vbSystemModal, "Loi"
End Sub
```

Bước 9: Chạy chương trình, nhập câu lệnh SQL vào TextBox, nhấp Thực thi. Sau đó lưu tập mẩu tin này lên đĩa. Mỗi khi muốn mở lại tập tin nào đó, sử dụng *Mở tập tin*.

II. BÀI TẬP TỰ LÀM

1) Sử dụng ADO, thiết kế Form nhập liệu cho bảng THangHoa (hình dưới). Ở đây thay vì hiển thị MaLoai, ta lại hiển thị TenLoai:



Hình VI.9: Form nhập liệu

2) Sử dụng ADO, thiết kế Form nhập liệu cho bảng TNhanVien.

3) Sử dụng ADO, thiết kế Form cho phép nhập (sửa, xóa) thông tin về một phát sinh về một mặt hàng nào đó trong ngày. *Lưu ý*: Trường STT là kiểu AutoNumber (Access), Ngay: lấy ngày hệ thống (hàm Now).

Chương 7 MÔI TRƯỜNG DỮ LIỆU

Mục tiêu:

Chương này gồm các bài tập nhằm rèn luyện cho sinh viên cách thức sử dụng môi trường dữ liệu (Data Environment) của VB để lập trình CSDL.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

Sử dụng thành thạo môi trường dữ liệu gồm:

- Tạo đối tượng Connection.
- Tạo đối tượng Command.
- Viết mã lệnh thao tác với môi trường dữ liệu.

Kiến thức có liên quan:

- Giáo trình Visual Basic, Chương 12.

Tài liệu tham khảo:

- **Visual Basic 6 Certification Exam Guide** – Chapter 10, Page 277 - **Dan Mezick & Scot Hillier** - **McGraw-Hill** - **1998**.
- **Tự học Lập trình cơ sở dữ liệu với Visual Basic 6.0 trong 21 ngày** (T1) - Chương 9, trang 395 - **Nguyễn Đình Tê** (chủ biên) – **Nhà xuất bản Giáo dục** - **2001**.

I. HƯỚNG DẪN

Bài 7-1

DATA ENVIRONMENT

Data Environment cho phép chúng ta tạo ứng dụng cơ sở dữ liệu với OLEDB một cách nhanh chóng và hiệu quả. Trong bài tập này, ta sẽ tìm hiểu về Data Environment và Report Designer của VB.

Bước 1: Tạo thư mục Basic\Bt7-1. Tạo một dự án kiểu Stardard EXE lưu vào trong thư mục đó.

Bước 2: Nếu mục Data Environment không có sẵn trong Project Explorer, ta chọn Project\Components…, đánh dấu vào mục Data Environment trong tùy chọn Designers, nhấp OK. Chọn Project\More ActiveX Designers… để thêm Data Environment vào môi trường soạn thảo.

Bước 3: Trong Data Environment, nhấp chuột phải vào đối tượng **Connection1**, chọn **Properties…**, chọn **Microsoft Jet 3.51 OLE DB Provider**.

Bước 4: Trong mục chọn *Connection*, chọn cơ sở dữ liệu mình muốn thao tác trong mục *Select or Enter a Database Name Box*; ở đây ta chọn CSDL BIBLIO.MDB (thường ở đường dẫn C:\Program Files\Microsoft Visual Studio\VB98\BIBLIO.MDB). Nhấp nút **Test Connection** để kiểm tra nối kết với CSDL có bị lỗi hay không? Ta sẽ nhấp OK nếu nối kết này thành công (nếu không ta phải kiểm tra lại).

Bước 5: Trong Data Environment, nhấp chuột phải vào đối tượng Connection1 và chọn *RENAME* để đổi tên thành BIBLIO.

Bước 6: Nhấp chuột phải vào BIBLIO và chọn *ADD COMMAND* trên menu, một đối tượng command được tạo ra với tên là Command1 trong Data Environment. Nhấp chuột phải vào đối tượng mới tạo này, chọn RENAME để đổi tên thành Publishers.

Bước 7: Nhấp chuột phải vào Publishers và chọn mục Properties, một hộp thoại quy định các thuộc tính cho đối tượng Publishers được mở ra. Trong mục chọn General, chọn Source Data là SQL Statement và ta nhập câu SQL sau vào khung nhập:

SELECT PubID, Name FROM Publishers WHERE Name LIKE ?

Câu SQL trên phải nhận vào một tham số từ chương trình gọi nó (dấu chấm hỏi). Nghĩa là để câu SQL này thực thi được, ta cần cung cấp một tham số đầu vào cho nó.

Bước 8: Để định nghĩa tham số, ta chuyển sang mục chọn Parameters, ta thấy có một tham số đã được định nghĩa tên là Param1. Mặc nhiên của tham số này là kiểu số, tuy nhiên ta sẽ đổi chúng thành kiểu chuỗi với các thuộc tính như sau:

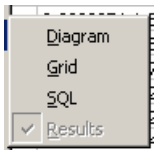
Data Type	adVarChar
Host Data Type	String
Size	255

Bước 9: Tạo một recordset con cho đối tượng Publishers command bằng cách nhấp chuột phải vào vào đối tượng này và chọn ADD CHILD COMMAND trên menu. Các

câu truy vấn này dùng để tìm thông tin về các sách dựa vào nhà xuất bản nào đó. Khi một command con được thêm vào, nó có tên là COMMAND1.

Bước 10: Trong command con này, nó cần phải có một câu SQL để truy xuất thông tin. Câu truy vấn này có nhiệm vụ kết nối thông tin từ các bảng Title và Author đối với từng loại nhà xuất bản.

Bước 11: Mở cửa sổ Data View bằng cách chọn View\Data View Window trên menu. Trong Data View, xác định thư mục Data Environment Connections. Đối tượng connection BIBLIO được hiển thị trong thư mục này. Mở đối tượng connection và xác định thư mục Tables. Trong thư mục này, chọn bảng Titles. Nhấp đúp vào bảng Titles này để hiển thị nội dung của bảng.

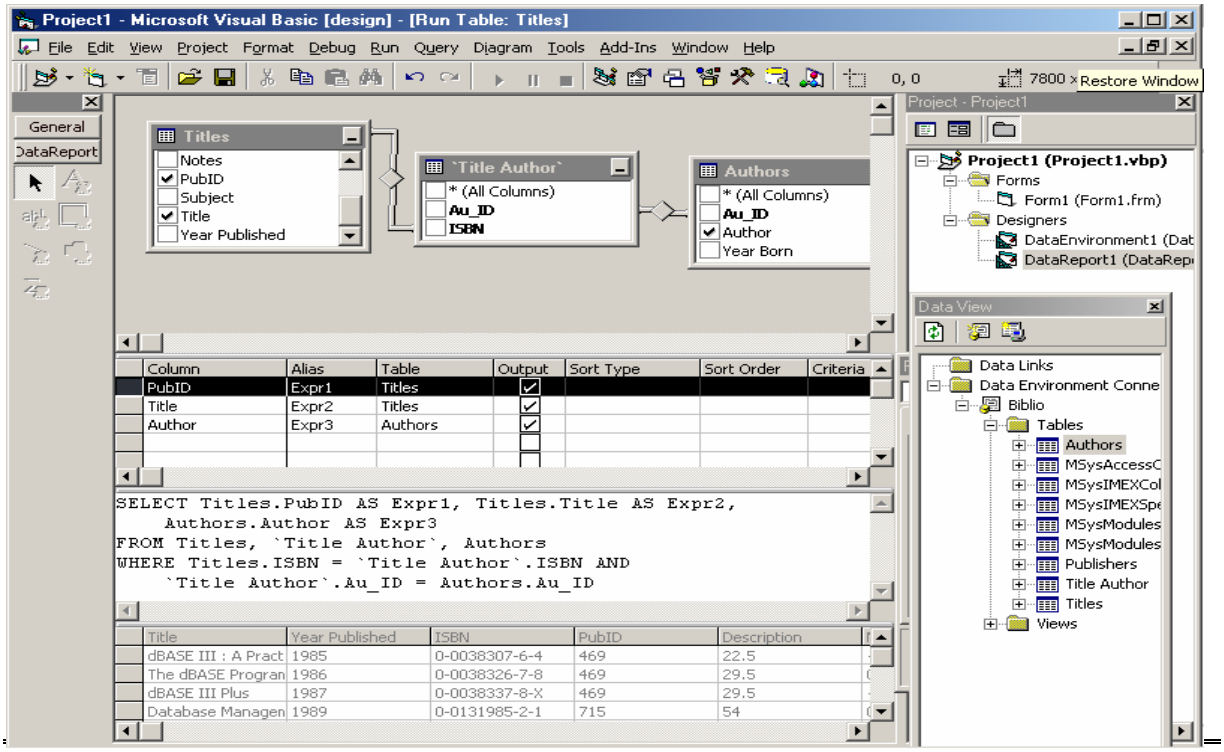


Bước 12: Khi đã hiển thị nội dung của bảng, ta mở Data Tools bằng cách chọn VIEW\SHOW PANES trên menu. Chọn tất cả các mục trong phần này (Diagram, Grid, SQL, Results).

Bước 13: Khi tất cả các mục của Data Tool được chọn, ta kéo các bảng Title Author và Authors vào khung diagram của Data Tools. Sau đó ta phải kiểm tra các liên kết của các bảng: bảng Titles có liên kết với bảng Title Author bởi trường ISBN không? Bảng Title Author có liên kết với bảng Authors bởi trường Au_ID không? Nếu không có các mối liên kết này ta phải chọn đúng trường trong bảng Title Author và kéo chúng vào trường tương ứng trong 2 bảng Titles và Authors.

Bước 14: Để tạo ra câu truy vấn, ta đánh dấu chọn vào trường PubID, Title trong bảng Titles; trường Author trong bảng Authors. Kế tiếp, kiểm tra khung Grid ta thấy có một dòng có dấu * (như hình dưới) biểu thị là ta đã chọn tất cả các trường của tất cả các bảng. Ta không cần chọn tất cả thông tin, do đó ta nhấp vào dòng có dấu * và bấm phím Delete để xóa chúng. Lúc đó trong khung SQL ta thấy có câu SQL sau:

```
SELECT Titles.PubID AS Expr1, Titles.Title AS Expr2,
        Authors.Author AS Expr3
FROM Titles, `Title Author`, Authors
WHERE Titles.ISBN = `Title Author`.ISBN AND
        `Title Author`.Au_ID = Authors.Au_ID
```



Hình VII.1: SQL Builder

Bước 15: Chọn câu SQL hiển thị bên trên rồi nhấp Edit\Copy.

Bước 16: Nhấp đúp trở lại vào Data Environment, nhấp chuột phải vào command con đã có và chọn PROPERTIES trên menu. Sau đó đổi tên của command trong mục General thành Titles. Chọn Source Data là SQL Command; sau đó dán nội dung câu SQL đã copy ở trên vào khung nhập câu SQL (chọn Edit\Paste).

Bước 17: Chọn mục RELATION , ta thấy trường PubID của cả hai bảng Publishers và Titles được hiển thị. Nhấp nút ADD để xác định liên kết giữa câu command cha và command con.

Bước 18: Đóng các cửa sổ và lưu dự án lại.

XÂY DỰNG GIAO DIỆN CHƯƠNG TRÌNH

Bước 19: Thêm vào một điều khiển vào dự án bằng cách chọn Project\Components trên menu; tìm đến mục Microsoft Hierachial Flexgrid Control 6.0 (OLEDB). Đánh dấu tùy chọn này và nhấp OK.

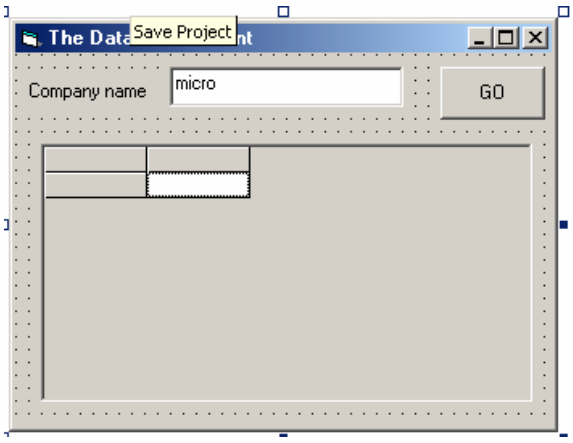
Bước 20: Mở Form1, tạo giao diện cho chương trình như dạng sau (hình bên dưới):

Item 1: Label
- Name: lblCompany
- Caption: Company Name

Item 2: TextBox
- Name: txtCompany
- Text: micro

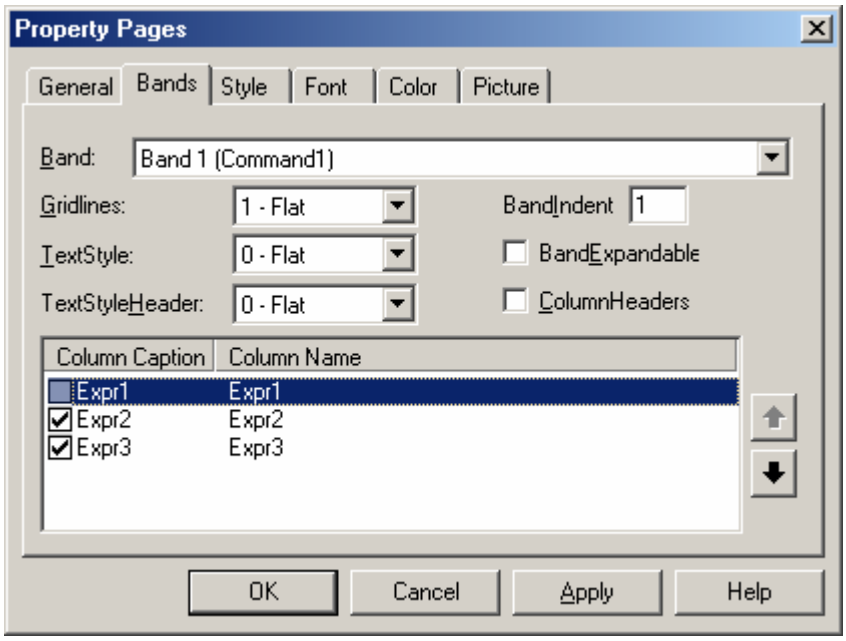
Item 3: CommandButton
- Name: cmdGO
- Caption: GO
- Default: TRUE

Item 4: Hierarchial FlexGrid
- Name: grdTitles
- AllowUserResizing: 1-flexResizeColumns
- DataSource: DataEnvironment1
- DataMember: Publishers
- FixedCols: 0



Hình VII.2: Giao diện ứng dụng

Bước 21: Nhấp chuột vào phải vào Hierarachical FlexGrid trên Form1, hộp thoại thuộc tính của Hierarachical được mở ra. Chọn mục BANDS trong hộp thoại này. Chọn *Band0* (Publishers) sau đó không đánh dấu vào trường PubID (với tên là Expr1) để nó không hiển thị khi thực hiện câu SQL. Cũng vậy, không đánh dấu vào trường PubID của Band1 (hình dưới):



Hình VII.3: Chọn trường hiển thị trên lưới

Bước 22: Đổi tên của các cột trong Band1 từ Expr2, Expr3 thành Title và Author bằng cách: Nhấp chuột 2 lần vào mục cần đổi tên trong *Column Caption*, nhập tên mới vào; sau đó chọn OK.

Bước 23: Mục đích của chương trình này là: Khi chương trình thực thi, trong TextBox có một từ là *micro*, từ được đề nghị tìm kiếm. Khi nhấp chuột vào nút nhấn; đoạn văn bản trong TextBox được dùng để thực thi câu SQL Publishers; câu SQL này sẽ truy tìm tất cả các công ty mà tên của chúng chứa chuỗi được nhập vào trong TextBox; kết quả trả về được hiển thị trên lưới. Do đó, ta sẽ xử lý sự kiện cmdGO_Click bằng đoạn mã sau:

```
MousePointer = vbHourglass
With DataEnvironment1
        .rsPublishers.Close
        .Publishers "%" & txtCompany.Text & "%"
End With
' Thiet lap tren luoi
Set grdTitles.DataSource = DataEnvironment1
grdTitles.CollapseAll
MousePointer = vbDefault
```

Bước 24: Lưu dự án và chạy chương trình. Sử dụng *micro* để tìm kiếm (ta có thể thử với từ khác như: *hill* hay *mill*)

TẠO BÁO CÁO (REPORT)

Bước 25: Nếu Data Report Designer không hiển thị dưới menu PROJECT, mở hộp thoại COMPONENTS, chọn *Data Report Designer* trong mục chọn Designers. Sau đó ta thêm Data Report Designer vào dự án bằng cách chọn PROJECT\ADD DATA REPORT trên menu.

Bước 26: Liên kết Data Report Designer với Data Environment nhờ việc thiết lập thuộc tính *DataSource* của Data Report là *DataEnvirronment1* và thuộc tính *DataMember* là *Publishers*.

Bước 27: Nhấp chuột phải trên Data Report Designer và chọn RETRIEVE STRUCTURE trên menu. Một hộp thoại xác nhận hiện lên và ta chọn YES.

Bước 28: Ta nhận thấy trên report gồm các phần: report header, page header, publisher information, title information, page footer, và report footer.

Bước 29: Chọn phần Report Header, trong phần này, thêm một Report Label vào, đặt Caption là *Book Report*; ta có thể đổi Font chữ và kích thước Font tương ứng.

Bước 30: Chọn *Group* header, trong phần này thêm một Report TextBox vào; đặt thuộc tính *DataMember* là *Publishers* và *DataField* là *Name*.

Bước 31: Chọn phần *Detail* của Report. Trong phần này, thêm 2 Report TextBox vào, đặt 2 điều khiển này cạnh nhau trong phần *Detail*; chọn TextBox thứ nhất và đặt thuộc tính *DataMember* là *Titles* và *DataField* là *Expr2*, đối với TextBox thứ hai: *DataMember*: *Titles, DataField: Expr3*.

Bước 32: Lưu dự án lại.

Bước 33: Chọn Form1 trong môi trường soạn thảo; tạo một menu trong Form1 bằng cách chọn *Tools\Menu Editor*; trong *Menu Editor*, tạo một Report menu với các phần tử là *Preview* và *Print* với các thuộc tính sau:

Item 1 – Menu
    Name: mnuReport
    Caption: Report
Item 2 – Menu
    Name: mnuPreview
    Caption: Preview
Item 3 – Menu
    Name: mnuPrint
    Caption: Print

Bước 34: Ta có thể xem báo cáo trước khi in nhờ hàm Show của đối tượng Report. Trong hàm xử lý sự kiện mnuPreview, thêm đoạn mã sau:

```
DataReport1.Show
```

Bước 35: In báo cáo được thực hiện nhờ hàm PrintReport của đối tượng Report. Trong hàm xử lý sự kiện mnuPrint, thêm đoạn mã sau:

```
DataReport1.PrintReport
```

Bước 36: Lưu và chạy chương trình. Thử hiển thị và in report.

Bài 7-2

BIỂU MẪU NHẬP LIỆU VỚI DATA ENVIRONMENT

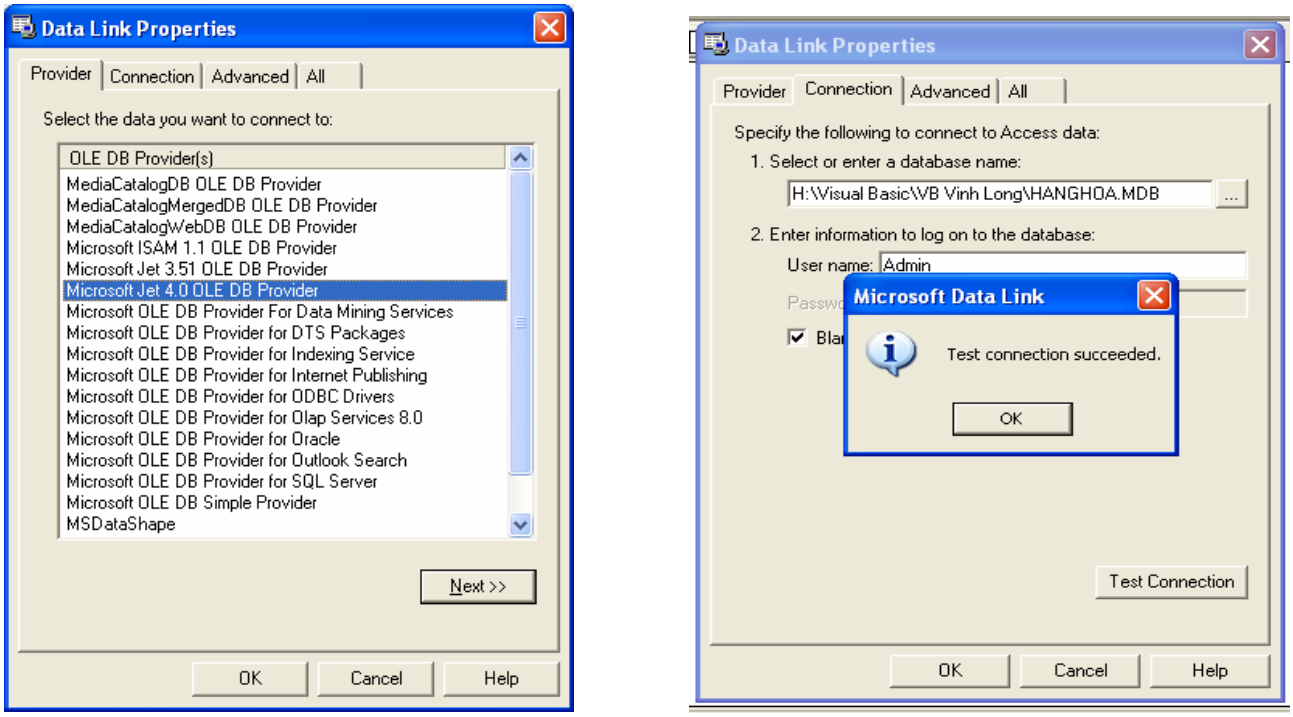
o Tạo một dự án mới; bổ sung Data Enviroment vào dự án của ta nhờ chọn Project/Add Data Environment. Khi lựa chọn mục này, môi trường DED sẽ hiển thị; sử dụng cửa sổ Properties để thiết lập thuộc tính Name là: datHH. Ở đây, ta sẽ sử dụng DED để kết nối với CSDL HANGHOA.MDB.

o Sửa lại thuộc tính Name của Connection1 là conHH; sau đó chuột phải lên conHH, chọn Properties.

Ở hộp thoại đầu tiên, ta phải chọn một trình cung cấp dữ liệu, ở đây chọn *Microsoft Jet 4.0 OLE DB Provider*, nhấn Next để tiếp tục.

Tiếp theo ta cần nhập chính xác đường dẫn đến tập tin CSDL, chẳng hạn ở đây là: H:\Visual Basic\HangHoa.Mdb.

Cuối cùng, nhấn nút Test Connection để kiểm tra việc nối kết dữ liệu chính xác hay không?



Hình VII.4: Tạo Connection

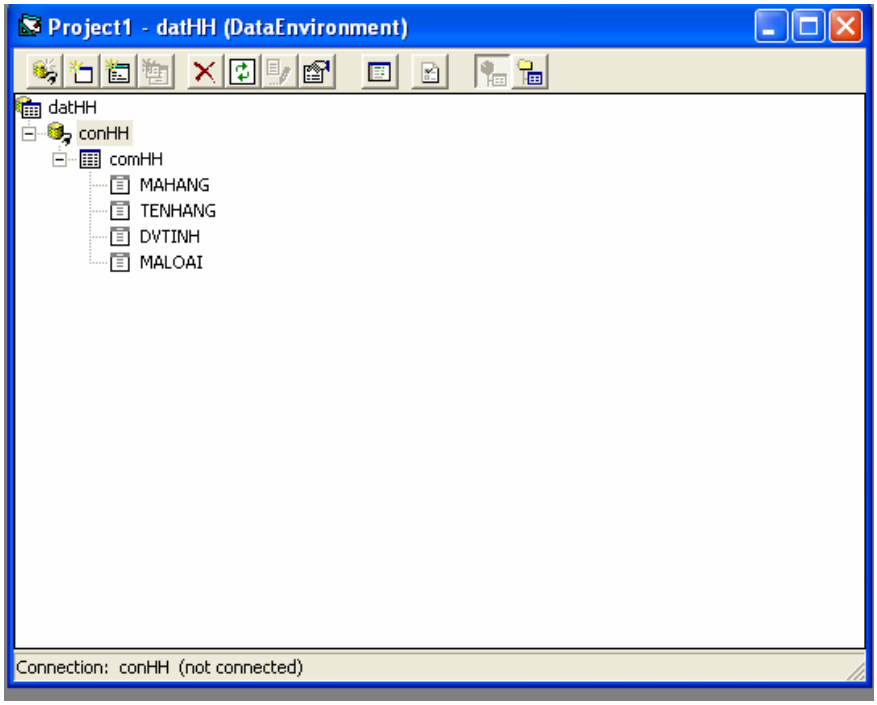
Tạo đối tượng Command:

o Xây dựng một đối tượng Command kết nối trực tiếp với Table (bảng) THANGHOA trong file dữ liệu HangHoa.mdb.

o Nhấp chuột phải trên kết nối dữ liệu conHH & chọn Add Command; sửa Command Name là: comHH; chọn Table từ Combo Box Database Object, chọn THANGHOA từ Combo Box Object Name.

o Trước khi đóng hộp thoại này, ta chuyển qua nhãn Advanced & thiết lập LockType là 3 – Optimistic (mặc nhiên là 1 – Read Only); Cursor Location: Use client-side cursor. Nhờ vậy ta mới có thể cập nhật Record Set từ chương trình của ta.

o Trở lại giao diện DED, ta được:



Hình VII.5: Đối tượng Command

Tạo một ứng dụng nhập liệu với DED

o Ở môi trường DED, ta kéo các trường của Command comHH vào Form1, chỉnh sửa lại cho thích hợp.

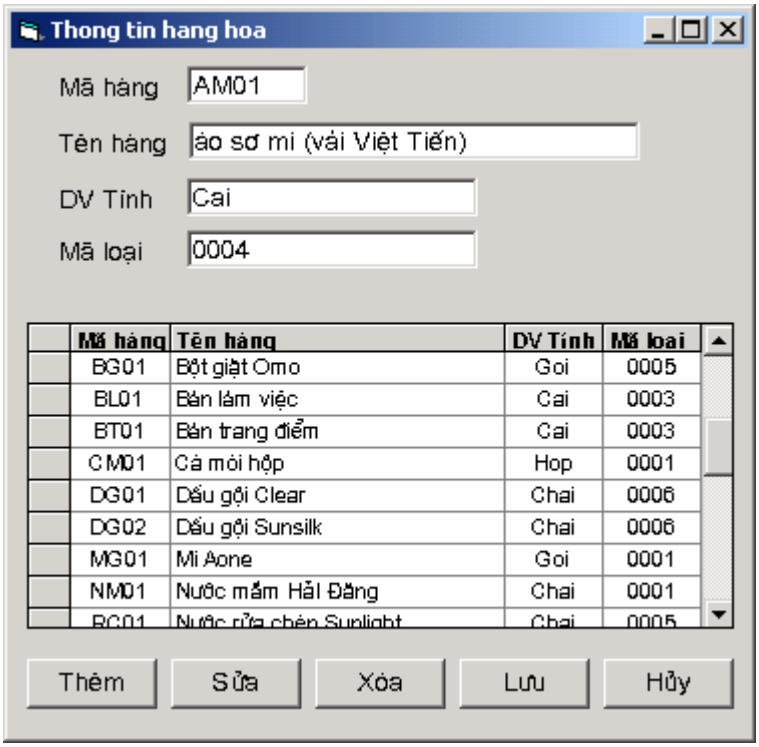
o Ở đây ta có sử dụng một lưới để hiển thị dữ liệu; do vậy ta chọn Project\Component; chọn Microsoft DataGrid Control 6.0 (OLE DB); sau đó kéo điều khiển này vào Form, thiết lập các thuộc tính cho thích hợp.

Name: grdHH.
DataSource: datHH
DataMember: comHH

o Nhấp chuột phải lên điều khiển DataGrid, chọn Retrieve Structure. Sau đó, lưu dự án & chạy chương trình ta được:

Hình VII.6: Kết quả thực thi ứng dụng

o Thêm các nút hành động (Thêm, Sửa, Xóa,…). Chẳng hạn các sự kiện cmd_Them_Click, cmdXoa_Click, cmdLuu_Click, cmdHuy_Click được xử lý:



Hình VII.7: Giao diện đầy đủ

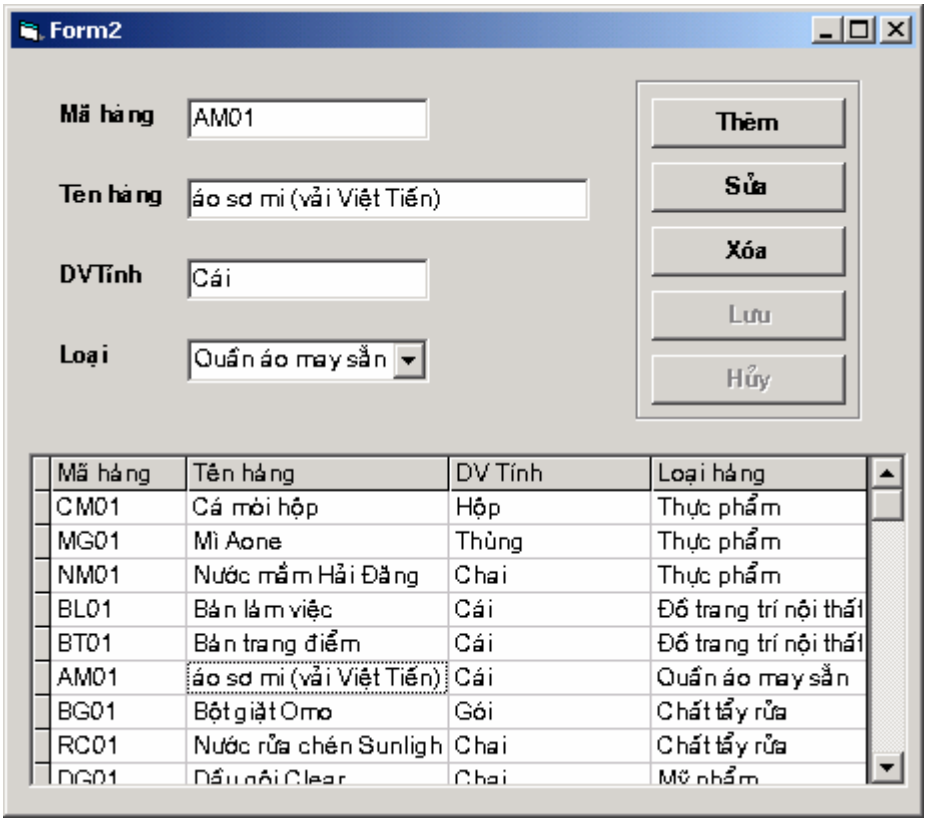
```
Private Sub cmdThem_Click()
    With datHH.rscomHH
        .AddNew
    End With
End Sub

Private Sub cmdXoa_Click()
    With datHH.rscomHH
        .Delete
       .Update
        Me.Refresh
    End With
End Sub

Private Sub cmdHuy_Click()
    With datHH.rscomHH
        .CancelUpdate
        Me.Refresh
    End With
End Sub

Private Sub cmdLuu_Click()
    On Error GoTo Xuly
    With datHH.rscomHH
        .Update
```

```
    End With
    Me.Refresh
    Exit Sub
Xuly:
    MsgBox Err.Description, vbCritical + vbSystemModal,
"Error"
End Sub
```

Như vậy, ta đã thiết kế xong một Form cho phép hiển thị thông tin các hàng hóa, Form này cho phép sửa đổi, thêm mới các mẩu tin trong bảng THANGHOA của CSDL HANGHOA.MDB.

II. BÀI TẬP TỰ LÀM

1) Sử dụng DataEnviroment, thiết kế Form nhập liệu cho bảng THangHoa (hình dưới). Ở đây thay vì hiển thị MaLoai, ta lại hiển thị TenLoai:



Hình VII.8: Form nhập liệu

2) Sử dụng DataEnviroment, thiết kế Form nhập liệu cho bảng TNhanVien.

3) Sử dụng DataEnviroment, thiết kế Form cho phép nhập (sửa, xóa) thông tin về một phát sinh về một mặt hàng nào đó trong ngày. *Lưu ý*: Trường STT là kiểu AutoNumber (Access), Ngay: lấy ngày hệ thống (hàm Now).

Chương 8 THIẾT LẬP BÁO CÁO VÀ XUẤT THÔNG TIN

Mục tiêu:

Chương này nhằm giới thiệu cho sinh viên cách thức để tạo báo biểu, thành phần không thể thiếu trong các ứng dụng cơ sở dữ liệu.

Học xong chương này, sinh viên phải nắm bắt được các vấn đề sau:

Sử dụng Data Report để tạo các loại báo biểu sau:

- Báo cáo dạng bình thường.
- Báo cáo có phân nhóm dữ liệu.
- Báo cáo dạng phân cấp.

Kiến thức có liên quan:

- Giáo trình Visual Basic, chương 13.

Tài liệu tham khảo:

- **Visual Basic 6.0 và Lập trình cơ sở dữ liệu** - Chương 21, trang 636 - **Nguyễn Ngọc Mai** (chủ biên) – **Nhà xuất bản Giáo dục** - **2000**.

- **Tự học Lập trình cơ sở dữ liệu với Visual Basic 6.0 trong 21 ngày** (T1) - Chương 5, trang 183 - Nguyễn Đình Tê (chủ biên) - **Nhà xuất bản Giáo dục** - **2001**.

I. HƯỚNG DẪN

TẠO KẾT NỐI VỚI CƠ SỞ DỮ LIỆU

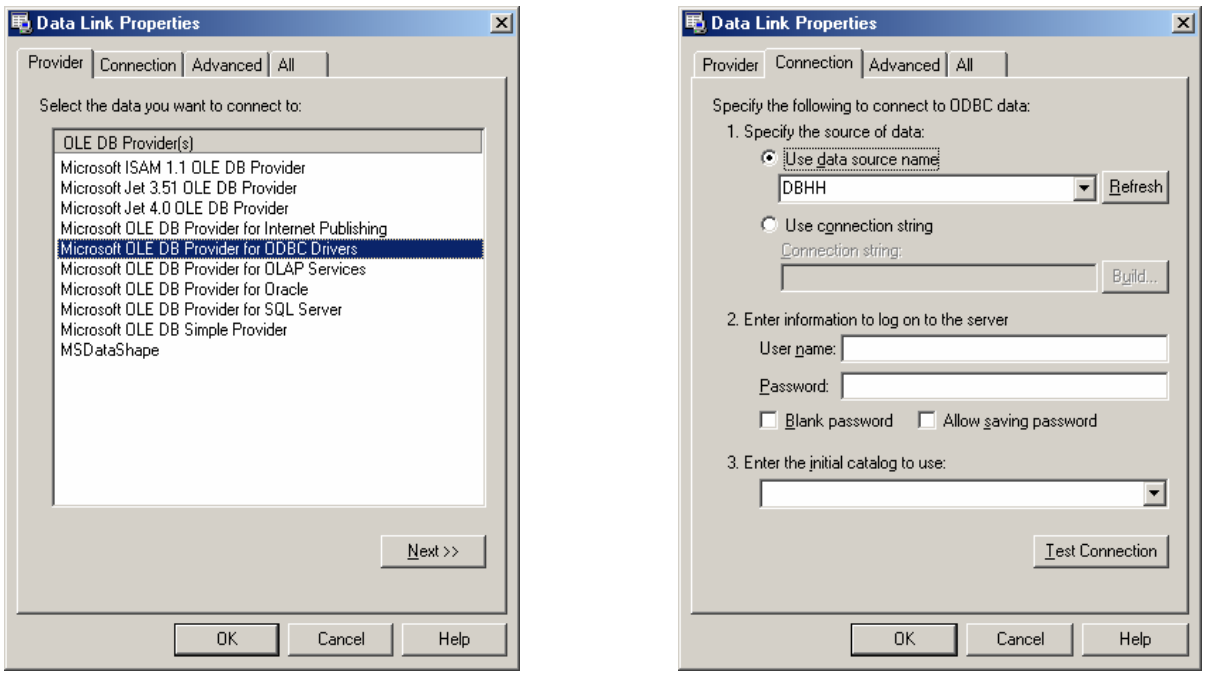
Bước 1: Tạo một dự án trong thư mục Basic\Bt8-1.

Bước 2: Để tạo một report, ta phải xác định dữ liệu nào sẽ sử dụng trong report, đó là nhiệm vụ của Data Environment.

Ta chèn một Data Environment vào dự án nhờ việc chọn Project\More ActiveX Designer\Data Environment.

Ta sử dụng Data Environment để lưu trữ các tập dữ liệu sẽ dùng trong report.

Bước 3: Nhấp đúp vào trình thiết kế Data Environmenr trong cửa sổ Project Explorer để mở trình thiết kế. Sau đó nhấp vào Connection1 và thay đổi thuộc tính Name của nó thành cnHangHoa; nhấp chuột phải vào cnHangHoa và chọn mục Properties, hộp thoại Properties hiện ra:



Hình VIII.1: Tạo nối kết đến CSDL

Bước 4: Ta sẽ tạo kết nối đến cơ sở dữ liệu DBHH.MDB có sẵn trên máy. Do đó ta nhấp nút Next sau khi chọn Microsoft OLEDB Provider for ODBC. Trong tùy chọn này ta chọn mục Use data source name; rồi chọn nguồn dữ liệu DBHH đã tạo ra từ trước đó.

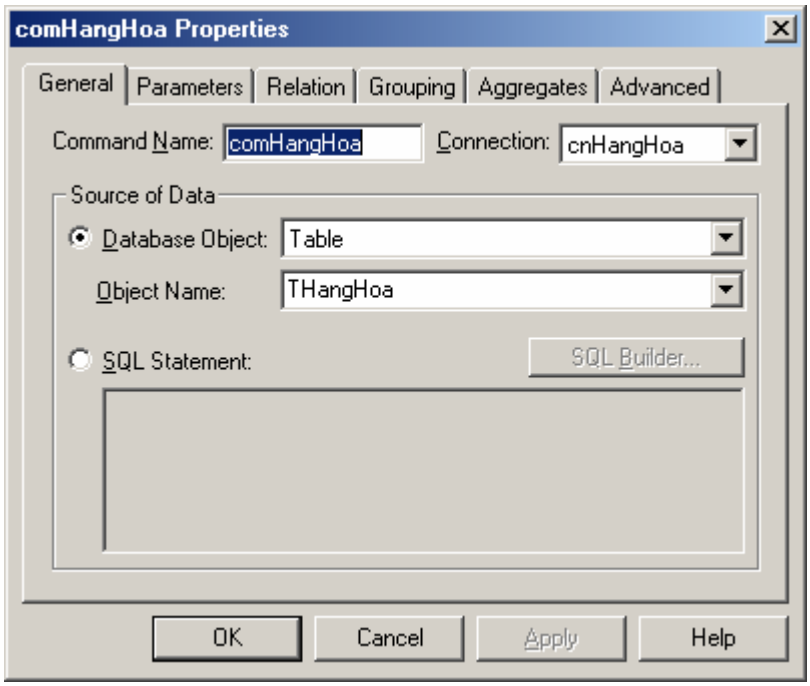
Bước 5: Nhấp nút Test Connection để kiểm tra xem nối kết có thành công hay không?

TẠO ĐỐI TƯỢNG COMMAND

Bước 6: Sau khi đã xác định Data Environment Connection, ta có thể xác định Command, Command lưu kết nối tới bảng dữ liệu và những trường sẽ được dùng trong

report. Nhấp nút phải chuột vào Command trong cửa sổ thiết kế Data Environment và chọn mục Add Command, một hộp thoại xuất hiện như hình bên dưới.

Bước 7: Thiết lập thuộc tính Name của Command là comHangHoa, chọn Connection là cnHangHoa. Trong phần DataBase Object chọn kiểu lấy dữ liệu là Table; sau đó trong ComboBox Object Name chọn Table THANGHOA.

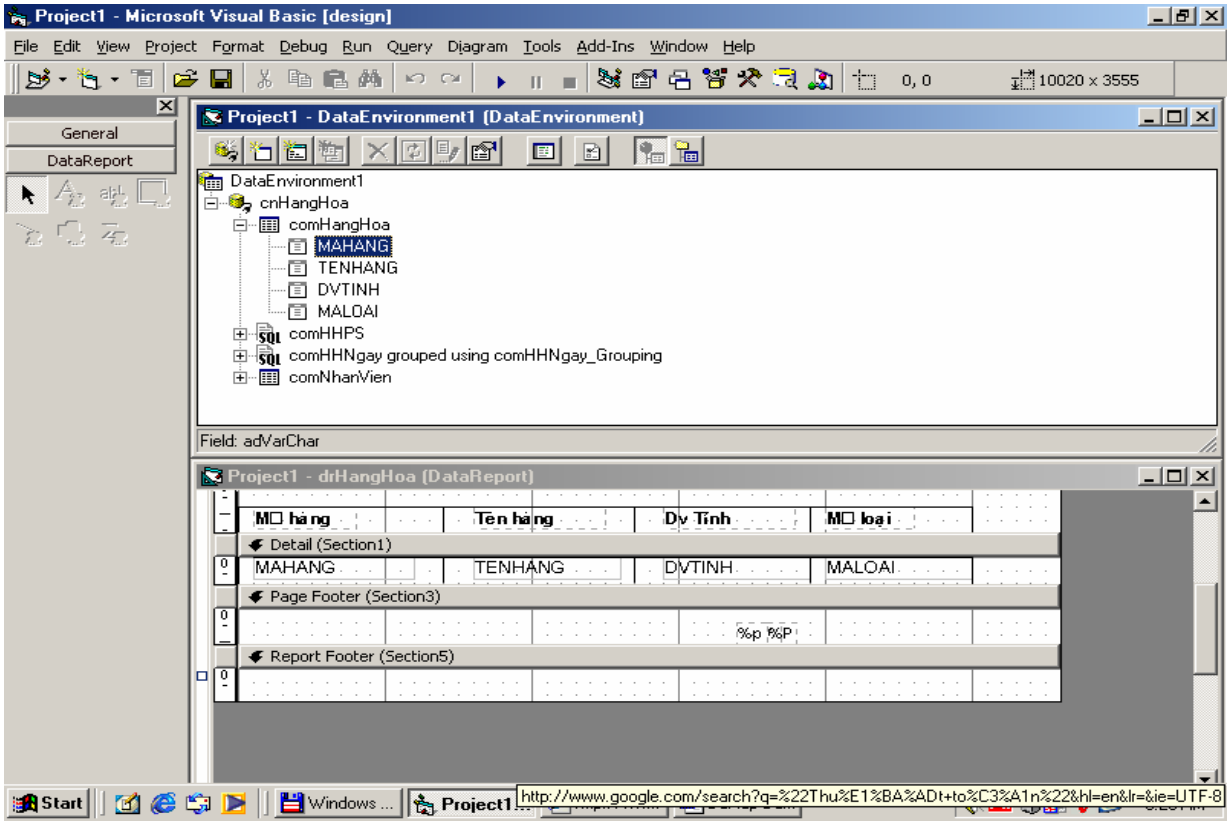


Hình VIII.2: Đối tượng Command

XÂY DỰNG MỘT BÁO CÁO (REPORT) ĐƠN GIẢN

Bước 8: Bây giờ ta sẽ tạo một báo cáo để hiển thị dữ liệu đã xác định trong Data Environment. Để làm điều này, nhấp phải chuột vào Project trong cửa sổ Project Explorer; rồi chọn Add\Data Report; sau đó đặt tên của report này lại là drHangHoa, DataSource: DataEnvironment1, DataMember: cnHangHoa (trong cửa sổ Properties). Bước kế tiếp ta đóng tất cả các cửa sổ ngoại trừ 2 cửa sổ DataEnvironment và drHangHoa.

Bước 9: Bước này chỉ đơn giản là nhấp chọn và kéo các trường từ Data Environment, mục command comHangHoa vào Data Report. Đối với report này, nhấp chọn và kéo các trường tên MaHang, TenHang, DVTinh, MaLoai. Khi thực hiện xong, màn hình tương tự như hình dưới:



Hình VIII.3: Thiết kế Report

Ở đây, ta chỉ kéo chọn các trường vào mục Detail của report, còn các tiêu đề cột (tương ứng với các trường); ta đặt chúng trong phần Page Header của report. Có một ghi nhận rằng, khi ta kéo thả các trường vào trong phần Detail của report, một report Label chứa tên trường và một report TextBox cho phép nhập liệu xuất hiện trong phần Detail. Do đó, trong phần Detail ta chỉ giữ lại TextBox, còn report Label ta chuyển chúng lên phần Page Header. Vì vậy coi như ta đã thiết lập xong phần hiển thị của report. Vấn đề còn lại chỉ là trang trí sao cho đạt yêu cầu về mặt thẩm mỹ.

CHÈN CÁC TIÊU ĐỀ ĐẦU TRANG VÀ CUỐI TRANG

Bước 10: Tất cả các report hoàn chỉnh đều phải có một số đặc điểm nào đó; nhất là report phải có một tên, số trang của report được hiển thị rõ ràng. Các mục này đặt tốt nhất là trong phần Report Header và Report Footer.

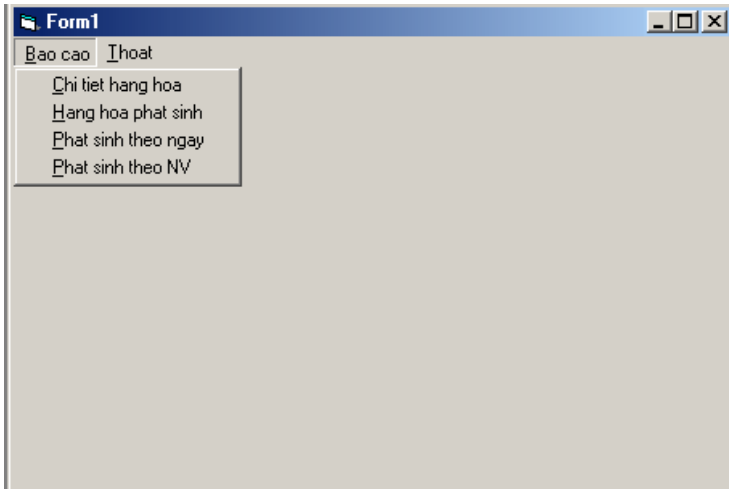
Ta cần phải thêm Report Header và Report Footer vào trong report của mình; đơn giản chỉ cần nhấp chuột phải vào report và chọn Show Report Header | Footer. Hai phần mới này sẽ được tạo ra.

Bước 11: Thêm một report Label vào phần Header của report; sau đó thay đổi thuộc tính Caption của nó thành *Chi tiết hàng hóa các loại*. Ở đây có thể chọn Font chữ với kích thước lớn (VD: 20) để tiêu đề của report được hiển thị lớn.

Bước 12: Nhấp chuột phải vào mục report Footer và chọn tùy chọn Insert Control | Current Page Number để chèn số trang vào; tiếp đến chọn Insert Control | Total Number of Pages để chèn tổng số trang của report.

Bước 13: Đến đây, phần thiết kế report coi như hoàn tất; ta lưu report lại.

Bước 14: Tạo giao diện cho chương trình như sau:



Hình VIII.4: Giao diện chính

Bước 15: Khi chọn Chi tiết hàng hóa, report vừa thực hiện sẽ được hiển thị; do đó ta xử lý sự kiện này như sau:

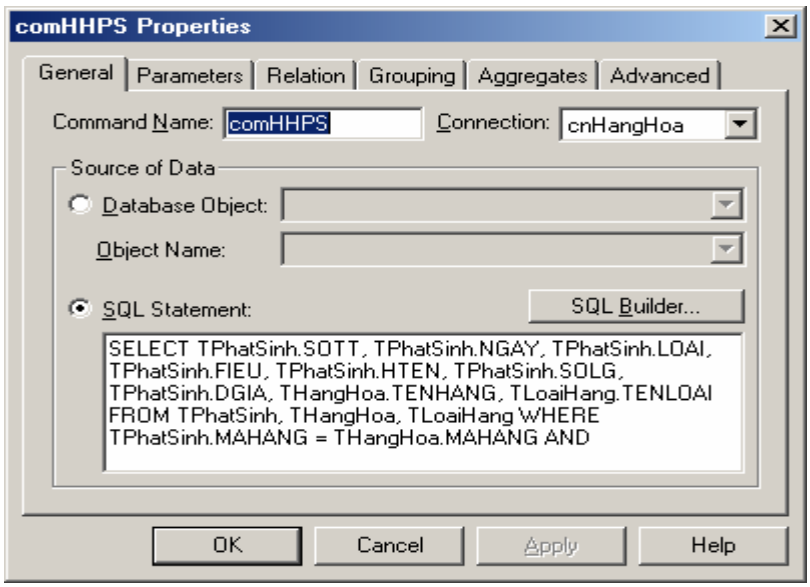
```
Private Sub mnuChiTiet_Click()
   drHangHoa.Show
End Sub
```

Bước 16: Lưu dự án và chạy chương trình.

TẠO REPORT BẰNG CÂU LỆNH SQL

Bước 17: Data Report cho phép ta tạo các report bằng cách sử dụng các câu lệnh SQL như là nền tảng của đối tượng Command. Trong phần này ta sẽ sử dụng câu lệnh SELECT để lấy dữ liệu từ các bảng TPHATSINH, THANGHOA, TLOAIHANG và chọn những trường sau: PhatSinh.SOTT, TPhatSinh.NGAY, TPhatSinh.LOAI, TPhatSinh.FIEU, TPhatSinh.HTEN, TPhatSinh.SOLG, TPhatSinh.DGIA, THangHoa.TENHANG, TLoaiHang.TENLOAI.

Bước 18: Chọn mục Data Enviroment1 trong Project Explorer, sau đó nhấp chuột phải vào đối tượng cnHangHoa, chọn Add Command. Ta điền vào các thuộc tính như hình dưới:



Hình VIII.5: Command là SQL

Bước 19: Trong mục Source of Data, thay vì chọn DatabaseObject như bài trước, ở đây ta chọn SQL Statement và nhập câu SQL tương ứng vào, rồi chọn OK.

Bước 20: Thêm một DataReport vào dự án với Name: drHHPS, DataSource: DataEnvironment, DataMember: comHHPS. Sau đó kéo thả các trường tương ứng vào report rồi trang trí lại chúng.

Bước 21: Lưu report lại. Ta sẽ gọi thực thi report này trong phần xử lý tình huống khi chọn mục Hàng Hóa Phát Sinh của menu.

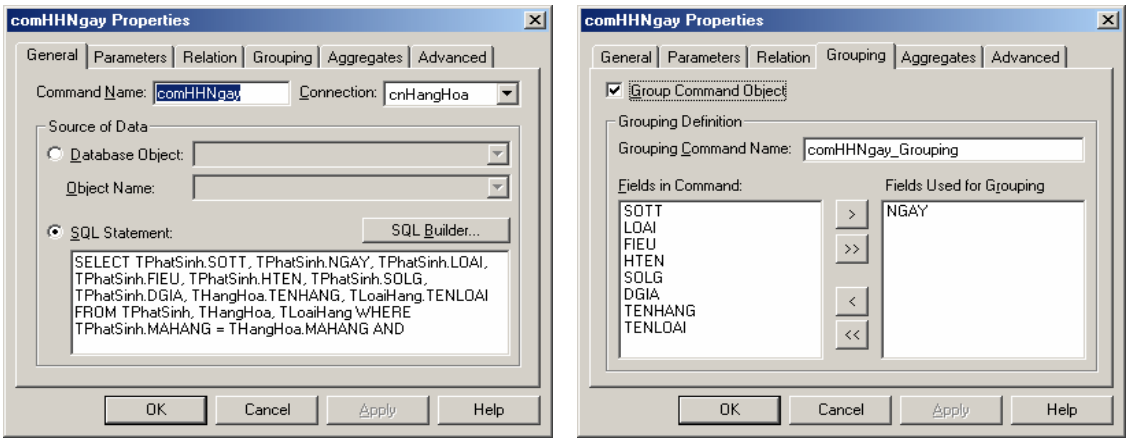
```
Private Sub mnuHHPS_Click()
   drHHPS.Show
End Sub
```

Bước 22: Lưu dự án và chạy chương trình.

NHÓM DỮ LIỆU

Bước 23: Ta có thể nhóm dữ liệu lại theo một nội dung nào đó. Ở đây, trong DataEnvironment1, thêm vào một đối tượng Command mới tên comHHNgay. Trong phần này ta sẽ lấy dữ liệu từ các bảng TPhatSinh, ThangHoa, TLoaiHang và lấy ra các trường TPhatSinh.SOTT, TPhatSinh.NGAY, TPhatSinh.LOAI, TPhatSinh.FIEU, TPhatSinh.HTEN, TPhatSinh.SOLG, TPhatSinh.DGIA, THangHoa.TENHANG, TLoaiHang.TENLOAI.

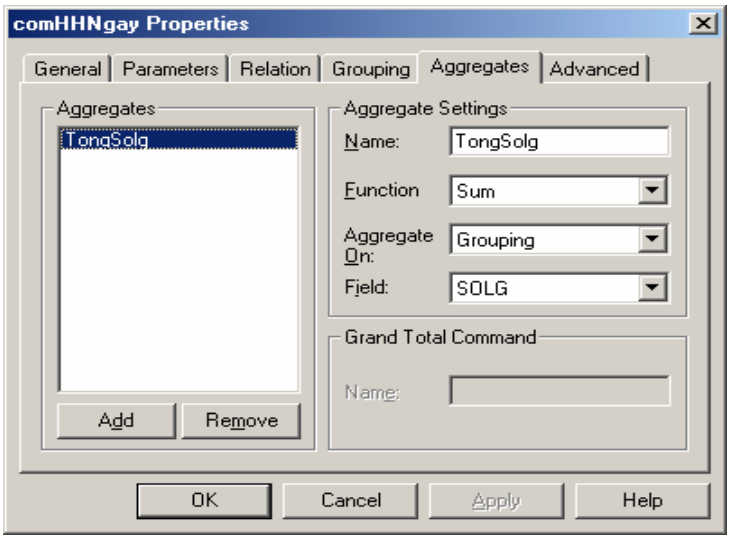
Bước 24: Ta sẽ nhóm dữ liệu trên lại theo từng ngày, do đó ta làm theo các bước như trong hình dưới:



Hình VIII.6: Nhóm dữ liệu

Chọn Grouping, chọn trường để nhóm lại là Ngay; sau đó nhấp OK.

Bước 25: Trong phần Aggregates, ta điền thông tin như hình:

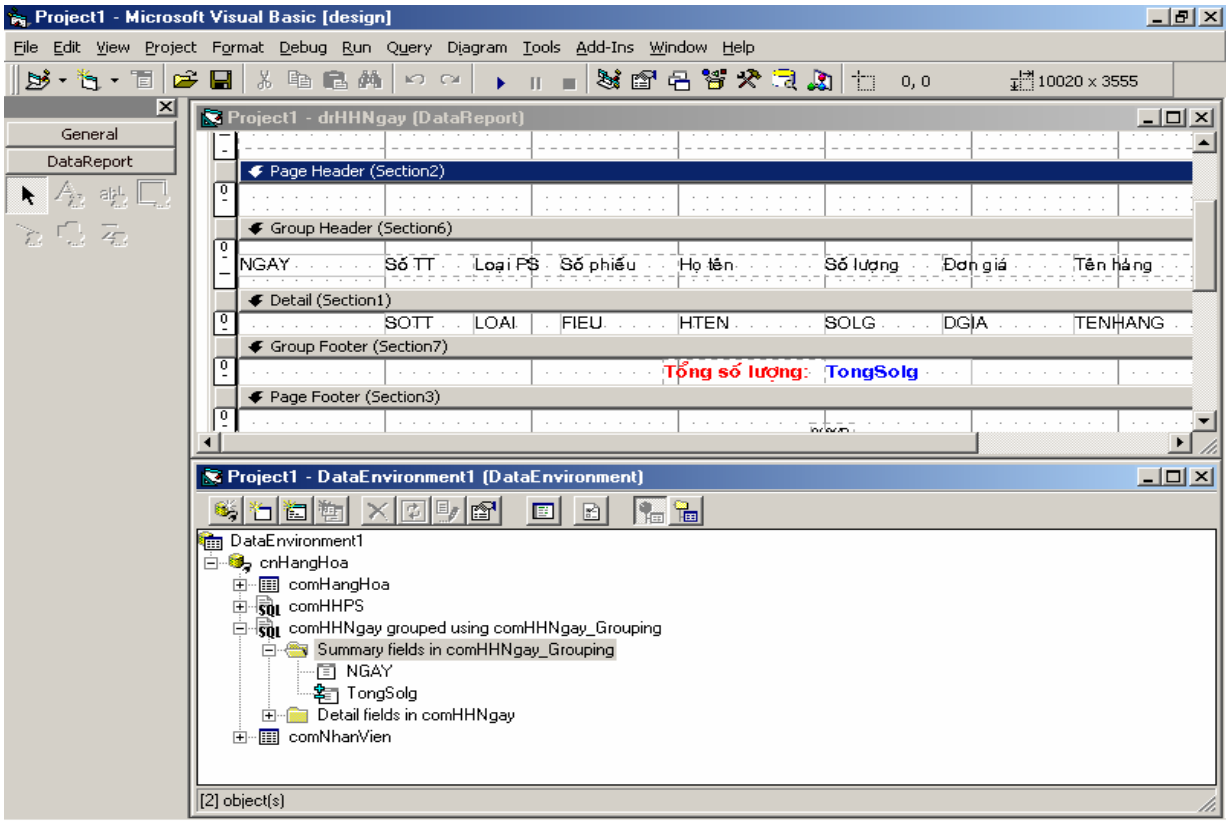


Hình VIII.8: Hàm thao tác trên nhóm

Ở đây ta sử dụng các hàm tập hợp để tính tổng số lượng bán được trong ngày. Do đó ta chọn mục Aggregate, chọn mục thích hợp và nhấp Add để thêm vào một hàm tính trên nhóm.

Bước 25: Chèn một DataReport mới vào tên drHHNgay, DataSource: DataEnvironment1, DataMember: comHHNgay_Grouping. Sau đó nhấp chuột phải vào Report và chọn mục Show Group Header | Footer.

Bước 26: Ta kéo trường ngày trong phần Sumary Fields in comHHNgay_Grouping vào phần Group Header. Trong phần Detail ta kéo thả các trường khác. Còn phần Group_Header, ta lại kéo TongSolg và đặt vào.



Hình VIII.9: Thiết kế Report

Bước 27: Lưu report lại. Ta sẽ gọi report hiển thị khi nhấp vào mục Phat sinh theo ngay trên menu:

```
Private Sub mnuPsngay_Click()
    drHHNgay.Show
End Sub
```

Bước 28: Lưu dự án lại, chạy chương trình, ta có thể in thử ra giấy.

Bước 29: Còn một report nữa ta sẽ thực hiện với điều kiện là tìm các phát sinh hàng hóa theo từng nhân viên (tương tự như ở report trên) và ta sẽ gọi thực thi ở mục menu còn lại.

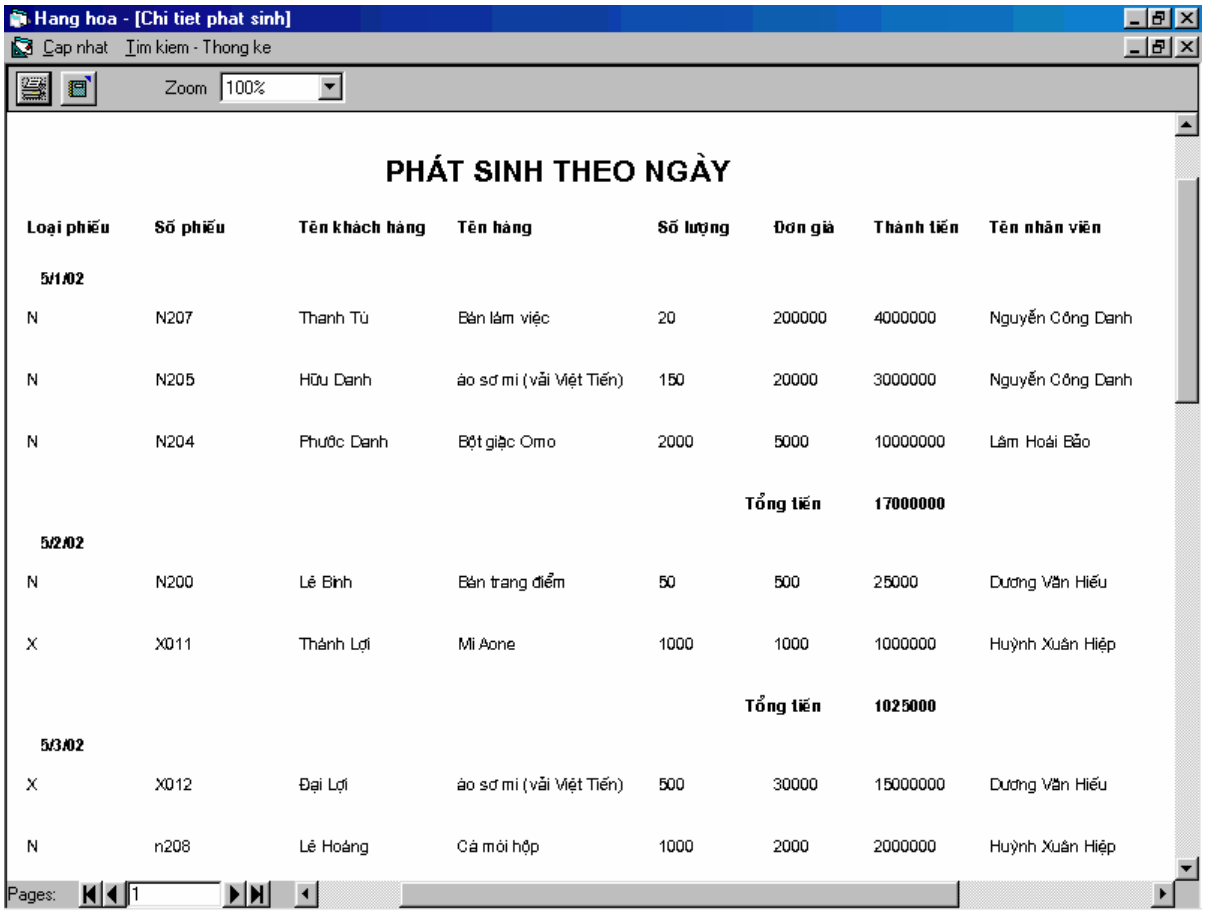
Bước 30: Xử lý sự kiện mnu_Thoat:

```
Private Sub mnuThoat_Click()
    End
End Sub
```

II. BÀI TẬP TỰ LÀM

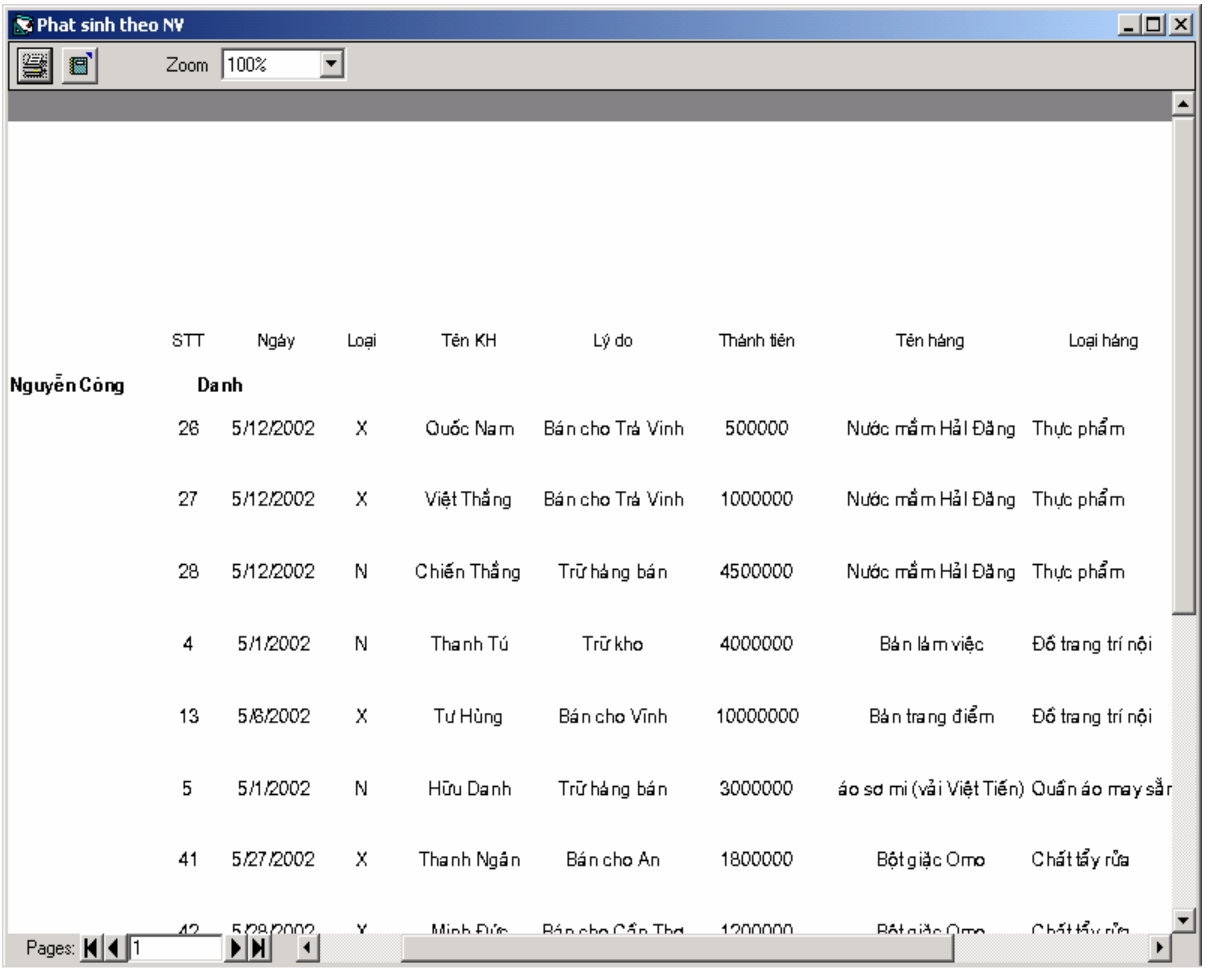
1) **Tạo báo cáo (Report)** cho phép in ra giấy thông tin của các **hàng hóa**, các thông tin này bao gồm: *Mã hàng hóa, tên hàng hóa, đơn vị tính, tên loại hàng* tương ứng.

2) **Tạo báo cáo (Report)** cho phép in ra giấy thông tin chi tiết về các **phát sinh theo từng ngày**, các thông tin này bao gồm: *Ngày lập, loại, Fieu, họ tên khách hàng, lý do, số lượng, đơn giá, thành tiền (với Thành tiền = Số lượng * Đơn giá), tên hàng hóa và họ tên nhân viên tương ứng*. (Hình VIII.10)



Hình VIII.10: Phát sinh theo ngày

3) **Tạo báo cáo (Report)** cho phép in ra giấy thông tin chi tiết về các **phát sinh theo từng nhân viên**, các thông tin này bao gồm: *Ngày lập, loại, Fieu, họ tên khách hàng, lý do, thành tiền (với Thành tiền = Số lượng * Đơn giá), tên hàng hóa và loại hàng tương ứng*. (Hình dưới)



Hình VIII.11: Phát sinh theo nhân viên

LỜI KẾT

Chương Thiết lập báo cáo cũng là chương kết thúc của giáo trình TT. Visual Basic. Tuy nhiên *lập trình sự kiện và lập trình cơ sở dữ liệu* với VB chỉ là một phần trong những khả năng mà VB mang lại. Hy vọng chúng tôi sẽ gặp lại bạn đọc trong những chuyên đề khác của VB.