

BÀI 1. CƠ BẢN VỀ VISUAL BASIC

I. Giới thiệu về Visual Basic

Visual Basic (Visual Basic) là sản phẩm của Microsoft, một thành phần của bộ Visual Studio.

Chức năng: Là một ngôn ngữ lập trình dùng để xây dựng các ứng dụng chạy trên môi trường Windows.

Đặc điểm: Trực quan, cung cấp các công cụ thuận lợi cho việc tạo các giao diện.

Cài đặt: từ đĩa CD VB6.0, chạy file setup, thực hiện các bước theo hướng dẫn.

Khởi động: Start/Programs/Microsoft Visual Basic 6.0/Microsoft V Basic 6.0

Phiếu New: standard EXE tạo mới một ứng dụng (Project). Phiếu Existing: mở ứng dụng đã có.

Cửa sổ giao diện của Visual Basic thường có các cửa sổ con, qui định việc ẩn hiện bằng các thao tác:

- View/Project Explorer: trình bày các thành phần của một ứng dụng.
- View/Properties Window: trình bày các thuộc tính của đối tượng được chọn.
- View/Form Layout Window: quy định vị trí xuất hiện của cửa sổ kết quả.

Mỗi ứng dụng là một chương trình bao gồm các chương trình con tương ứng với từng sự kiện. Chọn View/Code để viết và xem mã lệnh của các chương trình con này. Chọn View/Object để thiết kế giao diện cho ứng dụng. VB lưu giữ các thông tin của một ứng dụng bằng nhiều tập tin .FRM (nội dung form), .VBP (chương trình chính),... Vì vậy nên tạo thư mục riêng cho từng ứng dụng.

II. Các thao tác cơ bản khi xây dựng ứng dụng

1. Tạo mới một ứng dụng, mở một ứng dụng sẵn có: thao tác như đã nói trong mục trên.
2. Lưu một ứng dụng: chọn biểu tượng Save Project, đặt tên cho các tập tin .FRM, .VBP. Chú ý rằng phục vụ cho cùng một ứng dụng có nhiều tập tin.
3. Tạo một đối tượng (ô điều khiển): chọn loại đối tượng trong Toolbox rồi vẽ lên form.
4. Quy định thuộc tính cho đối tượng: chọn đối tượng, chọn thuộc tính, xác lập giá trị cho thuộc tính trong Properties Window.
5. Viết mã lệnh: nhấp đúp lên đối tượng hoặc View/Code rồi viết mã lệnh tương ứng. Trên cửa sổ Code có thể chọn đối tượng và sự kiện của đối tượng trên các combobox.
6. Chạy chương trình: F5 hoặc chọn Run/start hoặc chọn nút start trên thanh công cụ.
7. Thoát khỏi VB: như các ứng dụng khác trên windows

III. Các khái niệm cơ bản.

1. Đối tượng và các khái niệm liên quan.

Hoạt động của một chương trình VB hầu như đều liên quan đến một số các đối tượng nào đó. Các đối tượng này có thể là Form, có thể là các ô điều khiển như Label, Textbox, Command Button,... Một đối tượng có thể có các thành phần sau:

+ Thuộc tính (property): quy định những tính chất của đối tượng như kích thước, màu sắc, vị trí, giá trị,...

Cú pháp: <Tên_đối_tượng>.<tên thuộc tính>=<giá trị thuộc tính>

Ví dụ: txt1.text="Visual Basic"

Các thuộc tính thông dụng của các đối tượng:

- Name: tên để phân biệt với đối tượng khác, dùng để truy xuất đến các giá trị thuộc tính của đối tượng. Tên không chứa khoảng trống, không gõ dấu tiếng Việt.

Tên của các đối tượng nên đặt kèm theo phía trước là loại của đối tượng đó: Form: frm, TextBox: txt, Command: Cmd, Label: Lbl, ComboBox: Cmb,... để thuận lợi cho việc khai báo biến về sau.

- Caption: Tiêu đề của đối tượng.

- Font: qui định font chữ cho đối tượng.

- BackColor: màu nền của đối tượng.

- Height, Width: chiều cao, độ rộng của đối tượng.

- Left, Top: vị trí từ biên trái và biên trên đến góc trên trái của đối tượng.

- Visible: hiển thị (true) hay không hiển thị (false) đối tượng khi chạy ứng dụng.

+ Phương thức (method): hoạt động chủ động (không có tác động bên ngoài) của bản thân đối tượng như khi chương trình bắt đầu chạy,...

+ Sự kiện (event): hoạt động bị động của đối tượng như xảy ra khi kích chuột,...

Cú pháp <Tên_đối_tượng>.<tên phương thức>

Ví dụ Form1.show

2. Phương pháp lập trình hướng sự kiện.

+ Dùng giao diện để tương tác giữa người dùng và chương trình.

+ Người dùng phải hoạch định thứ tự cho các sự kiện.

+ Thứ tự các đoạn mã lệnh ứng với các sự kiện là không quan trọng.

+ Trên một đối tượng có thể có nhiều sự kiện khác nhau.

IV. Các đối tượng cơ bản.

1. Form

Là đối tượng chứa một số đối tượng khác của một ứng dụng. Khi chạy nó là màn hình giao diện của ứng dụng.

Một số sự kiện của form:

- Initialize: được hệ thống kích hoạt đầu tiên nên có thể dùng để thiết lập các thuộc tính ban đầu cho form.

- Load: xảy ra sau sự kiện trên có thể thiết lập các thuộc tính ban đầu cho các đối tượng của form.

- Click: xảy ra khi người dùng nhấp chuột trên form.

Một số phương thức của form:

- Show: hiển thị form lên màn hình, sau khi show được gọi các phương thức của các ô điều khiển khác trên form mới thực hiện được.

- Hide: che giấu một form nhưng không giải phóng bộ nhớ.

- Load: nạp form vào bộ nhớ nhưng chưa xuất hiện trên màn hình.

- Unload: ngược lại của Load

Có thể dùng tên ngầm định "Me" thay cho tên Form đang xử lý.

2. Label.

Đối tượng dùng để hiển thị thông tin như lời chú giải, lời nhắc (1) cũng có thể được dùng để xuất kết quả (2). Thuộc tính thường dùng là Caption. Những Label (1) thường xác lập thuộc tính trong cửa sổ properties. Các label (2) dùng lệnh dạng

`<Tên label>.Caption = "Nội dung"`

3. TextBox.

Đối tượng dùng để nhập, xuất dữ liệu. Thuộc tính quan trọng nhất là Text, chứa dữ liệu của ô, mặc định có kiểu chuỗi. Vì vậy, cần chuyển đổi kiểu nếu muốn sử dụng dữ liệu ở các kiểu khác. TextBox không có thuộc tính Caption.

Một số thuộc tính, sự kiện khác:

- ScrollBars: thuộc tính qui định thanh cuộn ngang, dọc có hay không.

- Maxlength: thuộc tính qui định chiều dài tối đa của dữ liệu nhập vào.

- Change: sự kiện xảy ra khi dữ liệu của ô bị thay đổi.

- GotFocus: sự kiện xảy ra khi con trỏ được nhảy vào ô.

- LostFocus: sự kiện xảy ra khi con trỏ nhảy ra khỏi ô.

- SetFocus: phương thức nhằm đưa con trỏ vào ô.

3. Command Button

Đối tượng thường dùng để điều khiển việc thực hiện một công việc nào đó của ứng dụng. Sự kiện thường dùng Click để thi hành một đoạn mã lệnh tương ứng.

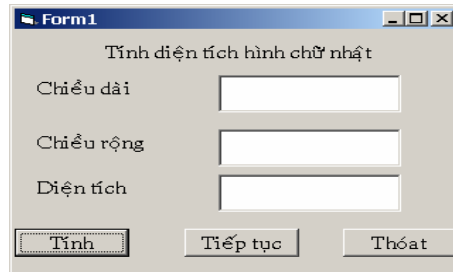
Kí hiệu & trong Caption của một command button có tác dụng tạo phím nóng, người sử dụng gõ ctrl+kí hiệu sau dấu & có tác dụng như nhấp chuột.

V. Ví dụ xây dựng ứng dụng trên VB

Xem hướng dẫn trong phần bài tập

BÀI TẬP SỐ 1

Bài tập 1.1 Tạo ứng dụng tính diện tích hình chữ nhật có giao diện như sau:



Khi chạy ứng dụng, nhập chiều dài, chiều rộng, nhấn nút tính kết quả xuất hiện trong ô diện tích. Khi nhấn nút tiếp tục, chương trình xóa các số cũ để nhập dữ liệu mới. Nhấn nút thoát để đóng Form, quay về VB.

Hướng dẫn:

+ Khởi động VB, tạo các label “Tính diện tích hình chữ nhật”, “chiều dài”, “chiều rộng”, “diện tích”. Nội dung các mục được quy định trong thuộc tính Caption của từng label.

+ Tạo các Textbox để nhập chiều dài, chiều rộng và xuất kết quả là diện tích. Đặt tên lần lượt cho các Textbox trên là a, b, S trong thuộc tính Name của từng Textbox. Để trống thuộc tính Text của các Textbox này.

+ Tạo các Command Button “Tính”, “Tiếp tục”, “Thoát” và nhấp đúp lên nút lệnh để mở cửa sổ Code và viết mã cho các nút lệnh này như sau:

Mã của Command1

```
Private Sub Command1_Click()
    s.Text = a * b
End Sub
```

Mã của Command2

```
Private Sub Command2_Click()
    a.Text = ""
    b.Text = ""
    s.Text = ""
    a.SetFocus
End Sub
```

Mã của Command3

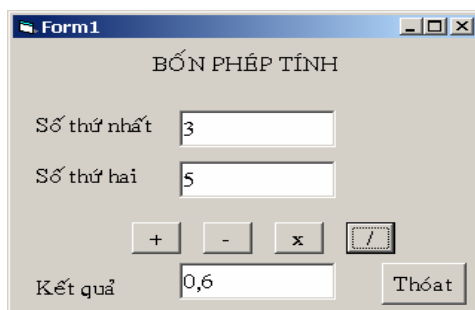
```
Private Sub Command3_Click()
    Unload Form1
End Sub
```

Phương thức a.SetFocus với mục đích đưa con trỏ đến ô để nhập chiều dài. Để làm điều này trong lần chạy đầu tiên ứng với thời điểm Form1 được khởi động, ta nhấp đúp lên Form1 và viết mã lệnh sau:

```
Private Sub Form_Load()
    Show
    a.Text = ""
    a.SetFocus
End Sub
```

Bài 1.2 Tạo ứng dụng tính diện tích hình thang khi nhập đáy lớn, đáy nhỏ, chiều cao.

Bài 1.3 Tạo ứng dụng để nhập vào 2 số nguyên, tính cộng, trừ, nhân, chia 2 số đó với giao diện như sau:



BÀI 2. NGÔN NGỮ LẬP TRÌNH VISUAL BASIC

I. Các kiểu dữ liệu cơ bản.

Mỗi kiểu dữ liệu quy định một tập hợp các giá trị và một tập các phép toán được sử dụng trên tập giá trị đó.

1. Kiểu số nguyên.

Tùy nhu cầu sử dụng số nhỏ hay lớn mà ta dùng kiểu phù hợp trong số các kiểu sau:

- Byte: kích thước 1 byte, phạm vi từ 0 đến 255.
- Integer: kích thước 2 bytes, phạm vi từ -32768 đến 32767.
- Long: kích thước 4 bytes, phạm vi từ -2^{31} đến $2^{31}-1$.

Các phép toán: cộng, trừ, nhân, chia, lũy thừa, div, mod tương ứng với các kí hiệu +, -, *, /, ^, \, mod. Trong đó, div là phép chia lấy phần nguyên, mod là phép chia lấy phần dư.

2. Kiểu số thực.

- Single: kích thước 4 byte, xác định đến 38 chữ số.
- Double: kích thước 8 byte, xác định đến 300 chữ số.

3. Kiểu chuỗi (string)

Chuỗi được đặt giữa hai dấu “ ” có độ dài đủ lớn.

Phép toán: nối chuỗi ứng với kí hiệu & hoặc +. Chú ý những trường hợp kết quả sai do VB chuyển kiểu tự động.

4. Kiểu logic (Boolean)

Chỉ có hai giá trị True, false hoặc 1,0. Các phép toán gồm hội, tuyển, phủ định ứng với kí hiệu and, or, not.

5. Kiểu ngày, giờ (Date)

Kích thước 8 byte ghi được cả ngày lẫn giờ. Thông thường nếu chỉ dùng ngày ta viết giữa hai dấu #.

Ví dụ: #22/12/2007#, #22-12-2007#

Phép toán: cộng, trừ giữa ngày và số; và phép trừ giữa ngày và ngày.

Chú ý: Tất cả các kiểu trên đều có phép so sánh =, <, >, <=>, <=> dựa theo quan hệ thứ tự của từng kiểu. Nên dùng thêm () để chủ động tránh nhầm lẫn thứ tự ưu tiên của các phép toán.

6. Kiểu Variant

Kiểu này để chứa dữ liệu bất kỳ thuộc các kiểu trên chỉ dùng khi nào thực sự cần thiết vì khó kiểm soát sự chuyển đổi kiểu tự động.

II. Hằng, biến.

1. **Hằng** là đại lượng không thay đổi giá trị trong chương trình.

Khai báo: Const <tên hằng> [as <kiểu>] = <giá trị>

Ví dụ: Const pi As Single = 3.1416

2. **Biến** là đại lượng có thể thay đổi giá trị trong chương trình.

Một cách đầy đủ mỗi biến xác định bởi các yếu tố: địa chỉ vật lý, tên, giá trị, thời gian tồn tại và phạm vi hoạt động. Trong đó, tên và giá trị thường được quan tâm đầu tiên.

Tên biến dùng để truy xuất đến giá trị. Tên biến không chứa khoảng trống, không gõ dấu tiếng Việt, không chứa dấu phép toán.

Khai báo: Dim <tên biến> [As <kiểu>]

Biến không khai báo kiểu sẽ là biến kiểu Variant.

Ví dụ. Dim x, y As integer. Khi đó x kiểu Variant, y kiểu integer.

Chú ý:

+ Nên thiết lập chế độ bắt buộc khai báo biến để dễ kiểm soát lỗi của chương trình. Để làm điều này vào Tools/Options/Require variable declaration hoặc khai báo dòng lệnh Option Explicit trong phần General của cửa sổ lệnh.

+ Những biến được khai báo trong phần General là biến toàn cục. Biến khai báo trong thủ tục là biến cục bộ. Biến cục bộ có tầm hoạt động trong thủ tục chứa nó. Biến toàn cục có tầm hoạt động khắp các thủ tục.

+ Phép gán: <tên_biến>=<giá trị> có tác dụng gán <giá trị > cho biến phía bên trái.

III. Hàm chuẩn.

1. Một số hàm toán học.

+ Abs(x): trả về trị tuyệt đối của x.

+ Sqr(x): trả về căn bậc hai của x.

+ Round(x,n): làm tròn x với n số lẻ.

- + Exp(x): e^x
- + Log(x): $\ln(x)$
- + Sin(x), Cos(x), Tan(x): các hàm lượng giác tương ứng.
- + Rnd(): trả về số thực ngẫu nhiên trong đoạn $[0,1]$, kích hoạt trước bằng thủ tục Randomize.

2. Một số hàm thời gian.

- + Day(ngày), Month(ngày), Year(ngày): trả về ngày, tháng, năm của một giá trị ngày.
- + WeekDay(ngày): trả về số nguyên là thứ của ngày trong tuần, ngoại lệ chủ nhật là số 1.

3. Một số hàm kiểu chuỗi.

- + Ucase(chuỗi): trả về chuỗi chữ in. Lcase(chuỗi): trả về chuỗi chữ thường.
- + Ltrim(chuỗi), Rtrim(chuỗi), Trim(chuỗi): cắt bỏ khoảng trống bên trái, bên phải, cả hai bên chuỗi.
- + Left(chuỗi,n), Right(chuỗi,n): lấy n kí tự của chuỗi từ bên trái, bên phải.
- + Len(chuỗi): trả về độ dài của chuỗi.
- + Mid(chuỗi,m,n): trả về n kí tự bắt đầu từ vị trí thứ m.
- + InStr(n, chuỗi 1, chuỗi 2): trả về vị trí xuất hiện của chuỗi 2 trong chuỗi 1 bắt đầu từ vị trí thứ n của chuỗi 1.
- + Replace(chuỗi 1, chuỗi 2, chuỗi 3): thay chuỗi 2 trong chuỗi 1 bằng chuỗi 3.

4. Một số hàm chuyển kiểu.

- + Val(chuỗi) : trả về số tương ứng với chuỗi.
- + CStr(số): trả về chuỗi tương ứng với số.
- + Cdate(chuỗi): trả về ngày tương ứng với chuỗi.

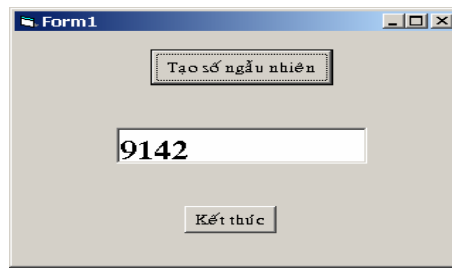
IV. Ví dụ. Bài tập 2.1

BÀI TẬP SỐ 2

Bài tập 2.1 Tạo ứng dụng để bốc thăm số xe bằng cách sinh số ngẫu nhiên có 4 chữ số theo giao diện sau:

- + Số 9142 trong textbox text1 được sinh ngẫu nhiên khi nhấp chuột vào nút lệnh “tạo số ngẫu nhiên”.
- + Chương trình dừng khi nhấp nút “Kết thúc”.
- + Mã lệnh cho nút “tạo số ngẫu nhiên” như sau

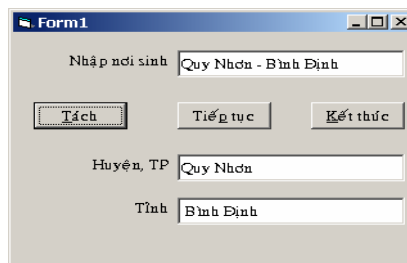

```
Private Sub Command1_Click()
Randomize
Text1.Text = Round(1000 + Rnd() * 8999)
End Sub
```



Cải tiến để sinh số ngẫu nhiên trong đoạn [a,b] với a, b nhập vào từ bàn phím.

Bài tập 2.2 Tạo ứng dụng tính diện tích hình tròn có bán kính nhập vào từ bàn phím, trong đó khai báo hằng pi =3.1416

Bài tập 2.3 Tạo ứng dụng để khi nhập vào nơi sinh của bạn gồm huyện (thành phố), tỉnh cách nhau bởi một dấu gạch nối - , chương trình sẽ tự động tách riêng tên huyện (thành phố), tên tỉnh với giao diện như sau:



Khi nhấp “tiếp tục” xoá trống các ô để thực hiện lượt mới.

Mã lệnh cho nút “Tách”:

```
Private Sub Command1_Click()
    Dim huyen As String, tinh As String
    Dim n As Byte
    n = InStr(1, Text1.Text, "-")
    huyen = Mid(Text1.Text, 1, n - 1)
    tinh = Mid(Text1.Text, n + 1)
    Text2.Text = huyen
    Text3.Text = tinh
End Sub
```

Mã lệnh cho nút “Tiếp tục”:

```
Private Sub Command2_Click()
    Text1.Text = ""
    Text2.Text = ""
    Text3.Text = ""
    Text1.SetFocus
End Sub
```

BÀI 3. CÁC CẤU TRÚC ĐIỀU KHIỂN

I. Cấu trúc rẽ nhánh.

1. IF...THEN...

Công dụng. Lựa chọn một trong hai công việc cần thực hiện.

Cú pháp. If <điều kiện> then
 <công việc 1>
 [Else
 <công việc 2>]
 End If

Điều kiện: Nếu điều kiện thoả mãn thì thực hiện công việc 1 ngược lại thực hiện công việc 2 (nếu có).

Ví dụ. Kiểm tra một số nguyên n là chẵn hay lẻ.

2. SELECT CASE...

Công dụng. Lựa chọn một trong nhiều công việc cần thực hiện.

Cú pháp: Select case <biểu thức>
 Case <danh sách giá trị 1>
 <công việc 1>

 Case <danh sách giá trị n>
 <công việc n>
 [Case Else
 <công việc n+1>]
 End Select

Điều kiện: Lần lượt kiểm tra giá trị của <biểu thức> từ trên xuống, nếu rơi vào danh sách giá trị nào thì thực hiện công việc này rồi thoát khỏi khối lệnh. Công việc n+1 (nếu có) được thực hiện khi giá trị <biểu thức> không rơi vào bất kỳ danh sách giá trị nào ở trên.

Danh sách giá trị có dạng: giá trị 1, giá trị 2,... giá trị n
 hoặc giá trị 1 To giá trị n (nếu các giá trị này liên tục)

Từ khoá Is được dùng trong trường hợp giá trị của biểu thức cần so sánh hơn, kém.

Ví dụ: Xếp loại sinh viên theo điểm trung bình

```
Select case dtb
Case Is>=9
    Loai="Xuất sắc"
Case Is>=8
    Loai="Giỏi"
Case Is>=7
    Loai="Khá"
Case Is>=6.5
    Loai="TB khá"
Case Is>=5
    Loai="TB"
Case Else
    Loai="Yếu"
End Select
```

Chú ý. Điều khiển để chương trình nhảy đến một vị trí bất kỳ bằng lệnh

Goto <nhãn>

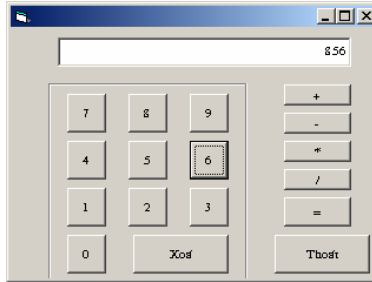
Trong đó nhãn có dạng <tên nhãn:>

3. Ví dụ. Xem bài tập 3.1

BÀI TẬP 3

Bài tập tập 3.1

Tạo ứng dụng máy tính cho phép nhập số và thực hiện các phép tính cộng, trừ, nhân chia với các số nguyên với giao diện như sau:



Hướng dẫn:

- + Tạo textbox txtso để xuất hiện các số được chọn và kết quả.
- + Tạo các Command ứng với các số, các phép toán,..
- + Viết các mã lệnh gợi ý như sau, chú ý rằng các biến pt (phép toán), so1 giữ số thứ nhất của phép toán phải là các biến toàn cục

```
Option Explicit
Dim pt As String
Dim so1 As Integer

Private Sub cmd0_Click()
txtso = txtso & "0"
End Sub

Private Sub cmd1_Click()
txtso = txtso & "1"
End Sub

.....
Private Sub cmd9_Click()
txtso = txtso & "9"
End Sub

Private Sub cmdcong_Click()
so1 = Val(txtso)
txtso.Text = ""
pt = "+"
End Sub

Private Sub cmdtru_Click()
so1 = Val(txtso)
txtso.Text = ""
pt = "-"
End Sub

Private Sub cmdnhan_Click()
so1 = Val(txtso)
```

```

txtso.Text = ""
pt = ""
End Sub

Private Sub cmdchia_Click()
so1 = Val(txtso)
txtso.Text = ""
pt = "/"
End Sub

Private Sub cmdbang_Click()
Dim kq As Single, so2 As Single
so2 = Val(txtso.Text)
Select Case pt
Case "+": kq = so1 + so2
Case "-": kq = so1 - so2
Case "*": kq = so1 * so2
Case "/"
If so2 = 0 Then
txtso.Text = "không chia được"
GoTo abc
Else
kq = so1 / so2
End If
End Select
txtso.Text = kq
abc:
End Sub

Private Sub cmdxoa_Click()
txtso = ""
End Sub

Private Sub cmdthoat_Click()
Unload Me
End Sub

```

Bài tập 3.2 Tạo ứng dụng giải phương trình bậc hai $ax^2 + bx + c = 0$.

Bài tập 3.3 Tạo ứng dụng để khi nhập vào một tháng của năm nào đó, chương trình cho biết số ngày của tháng này. Biết rằng các tháng 1,3,5,7,8,10,12 có 31 ngày; các tháng 4,6,9,11 có 30 ngày; tháng 2 năm nhuận có 29 ngày, năm không nhuận có 28 ngày. Năm nhuận là năm chia hết cho 400 hoặc là năm chia hết cho 4 mà không chia hết cho 100.

BÀI 4. CÁC CẤU TRÚC LẶP

I- FOR...NEXT

Công dụng. Thực hiện công việc lặp lại với số lần lặp biết trước.

Cú pháp. For <biến_điều_khiển=giá_trị_đầu> To <giá_trị_cuối> [step n]
<công việc>

Next

Điều khiển:

+ B1. Biến_điều_khiển nhận giá_trị_đầu

+ B2. Nếu biến điều_khiển nhỏ hơn hoặc bằng giá trị_cuối thì thực hiện công việc rồi tăng biến điều_khiển lên n rồi quay lại B2.

+ B3. Nếu biến điều_khiển lớn hơn giá trị_cuối thì kết thúc điều khiển.

Exit For : được dùng trong <công việc> khi nào muốn thoát khỏi vòng lặp ngay lập tức (không chờ biến điều khiển lớn hơn giá trị cuối).

Ví dụ. Đoạn chương trình sau tính tổng các số nguyên từ 1 đến n (nhập trong txtn)

```
Dim S As Integer
Dim i As Integer, n As Integer
S = 0
n = Val(txtn.Text)
For i = 1 To n
    S = S + i
Next
lblkq.Caption = "Kết quả là" & S
```

II. DO WHILE...LOOP

Công dụng. Thực hiện công việc lặp lại với số lần lặp không xác định trước.

Cú pháp 1. Do while <điều kiện>

<công việc>

Loop

Điều khiển: Nếu điều kiện thoả mãn thì thực hiện công việc và lặp lại cho đến khi điều kiện không thoả mãn thì thoát khỏi cấu trúc này.

Cú pháp 2. Do

<công việc>

Loop Until <điều kiện>

Điều khiển: Tương tự như cú pháp 1 nhưng kiểm tra điều kiện sau khi thực hiện công việc.

Ví dụ. Tính số tháng cần thiết gửi ngân hàng với số vốn là $v = 100.000.000$ đồng, mức lãi suất $k = 0.65\%$ tháng để có được cả vốn lẫn lãi số tiền là $t = 115.000.000$ đồng (biết rằng không rút lãi hàng tháng).

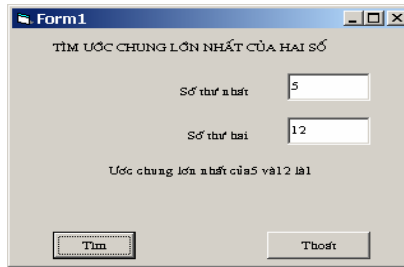
```
Private Sub Tinh_Click()
Dim v As Single, k As Single, st As Single, thang As Byte
v = Val(txtv)
k = Val(txtk)
st = Val(txtst)
thang = 0
Do While v < st
    thang = thang + 1
    v = v * (1 + k)
Loop
txtkq.Text = thang
End Sub
```

BÀI TẬP

Bài tập 4.1 Lập ứng dụng tính $n!$ với n nguyên dương nhập từ bàn phím.

Bài tập 4.2 Lập ứng dụng tìm ước chung lớn nhất của 2 số nguyên dương nhập từ bàn phím.

Hướng dẫn: Tạo giao diện như sau



- + Textbox txta: để nhập số thứ nhất
- + Textbox txtb: để nhập số thứ hai
- + Label lblkq: thông báo kết quả

```
Private Sub Command1_Click()
    Dim a As Integer, b As Integer, r As Integer
    a = Val(txta.Text)
    b = Val(txtb.Text)
    Do While b <> 0
        r = a Mod b
        a = b
        b = r
    Loop
    lblkq.Caption = "Ước chung lớn nhất của " & txta.Text & " và " & txtb.Text & " là" & a
End Sub

Private Sub Command2_Click()
    Unload Me
End Sub
```

BÀI 5. BIẾN MẢNG

I. Biến mảng.

Khái niệm: Mảng là tập hợp hữu hạn các phần tử cùng kiểu, cùng tên và phân biệt theo chỉ số.

Khai báo: Dim Tên_mảng(n) [As kiểu_phần_tử]

Sẽ tạo mảng n phần tử được đánh chỉ số từ 0 đến n-1.

Ví dụ: Dim A(5) As Integer

Khai báo: Dim Tên_mảng(n₁ to n₂) [As kiểu_phần_tử]

Sẽ tạo mảng gồm các phần tử được đánh chỉ số từ n₁ đến n₂.

Ví dụ: Dim A(3 to 5) As Integer

Truy xuất: Tên_mảng(i) dùng để truy xuất đến giá trị của phần tử có chỉ số i của mảng.

Chú ý:

+ Hàm Lbound(Tên_mảng): trả về chỉ số nhỏ nhất của mảng,

Ubound(Biến_mảng): trả về chỉ số lớn nhất của mảng.

+ Khai báo mảng hai chiều:

Dim Tên_mảng(m,n) [As kiểu_phần_tử]

Dim Tên_mảng(n₁ to n₂, n₃ to n₄) [As kiểu_phần_tử]

+ Truy xuất mảng hai chiều: Tên_mảng(i,j) dùng để truy xuất đến giá trị của phần tử nằm trên dòng i, cột j của mảng.

+ Mảng có số phần tử không xác định trước (mảng động):

Khai báo: Dim Tên_mảng() [As kiểu_phần_tử]

Thao tác: Redim Dim Tên_mảng(n): thực sự tạo mảng n phần tử.

+ Mảng gán giá trị trước:

Chỉ dùng đối với mảng động và có kiểu phần tử là Variant, chỉ số từ 0 trở đi.

Ví dụ Dim a()

A=Array("Một", "hai", "ba")

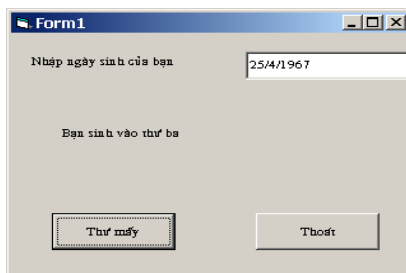
II. Mảng các ô điều khiển

Trường hợp dùng mảng ứng với các ô điều khiển trên Form (thường là các TextBox) thuận tiện nhất là dùng mảng các ô điều khiển. Mảng các ô điều khiển là một biến mảng đặc biệt mà mỗi phần tử của mảng ứng với một ô điều khiển.

Khi tạo ra các ô điều khiển này ta đặt tên trùng nhau, tên này ứng với tên của mảng; chỉ số trong thuộc tính Index khác nhau ứng với chỉ số của phần tử.

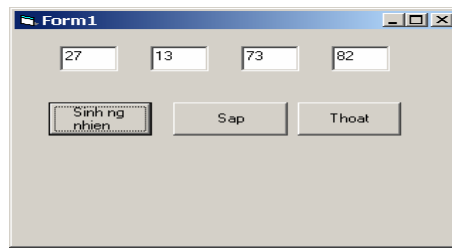
BÀI TẬP 5

Bài tập 5.1 Nhập vào ngày sinh của bạn, thông báo ngày này rơi vào thứ mấy (bằng tiếng Việt).



```
Private Sub Command1_Click()
Dim TenThu()
TenThu = Array("Chủ nhật", "thứ hai", "thứ ba", "thứ tư", "thứ năm", "thứ sáu", "thứ bảy")
Dim ngay As Date, thu As Byte
ngay = CDate(txtngay)
thu = Weekday(ngay)
If thu = 1 Then
    lblkq.Caption = "Bạn sinh vào " & TenThu(0)
Else
    lblkq.Caption = "Bạn sinh vào " & TenThu(thu - 1)
End If
End Sub
```

Bài tập 5.2 Sinh ngẫu nhiên 4 số nguyên dương trong đoạn từ 0 đến 100. Sắp xếp các số này thành một dãy số tăng.



```
Option Explicit
Const n As Byte = 4

Private Sub Command3_Click()
Dim i As Integer, j As Integer
Randomize
For i = 0 To n - 1
    txta(i).Text = Round(Rnd() * 100, 0)
Next
End Sub

Private Sub Command1_Click()
Dim i As Integer, j As Integer
Dim tg As Integer
For i = 0 To n - 2
    For j = i + 1 To n - 1
        If Val(txta(i).Text) > Val(txta(j).Text) Then
            tg = Val(txta(i).Text)
            txta(i).Text = txta(j).Text
            txta(j).Text = tg
        End If
    Next
Next
Next
End Sub
```

BÀI 6. CHƯƠNG TRÌNH CON

I. Khái niệm – Phân loại

Khái niệm: Một chức năng tương đối độc lập được sử dụng lặp lại nhiều lần được tổ chức thành một đoạn mã lệnh được gọi là chương trình con (CTC).

Ví dụ. Tìm UCLN của hai số nguyên được sử dụng trong quy đồng mẫu số, tối giản phân số nên thường được thiết kế thành một chương trình con.

Phân loại: + Hàm: CTC tính toán trả về một giá trị.

+ Thủ tục: CTC thực hiện một công việc không trả về giá trị nào.

Phạm vi: + CTC viết trong Code của Form nào chỉ hoạt động được cho Form đó.

+ CTC viết trong Modules sẽ hoạt động được trong toàn Project (Chọn Project/Add Modules, viết chương trình con, khi lưu một Modules (đơn thể) sẽ tạo ra một tập tin . Bas tương ứng. Tập tin .Bas có thể thêm vào (Add) vào một Project khác. Trong một Modules có thể chứa nhiều CTC.

II. Cấu trúc của chương trình con

1. Hàm Function <Tên_hàm> ([ds tham số]) As <kiểu_kết_quả>

Các lệnh

Tên_hàm = <giá trị>

End Function

- + Tên hàm: tự đặt theo quy tắc như quy tắc đặt tên biến.
- + ds tham số: là nhiều tham số khai báo như danh sách biến.
- + <giá trị> là kết quả trả về của hàm.
- + Trong hàm có thể dùng **Exit Function** để thoát khỏi hàm

Ví dụ: Hàm tính UCLN của hai số nguyên

```
Function UCLN(a As Integer, b As Integer) As Integer
Do While b <> 0
    r = a Mod b
    a = b
    b = r
Loop
UCLN = a
End Function
```

2. Thủ tục Sub <Tên_thủ_tục> ([ds tham số])

Các lệnh

End Sub

Ví dụ. Hoán vị giá trị của hai biến

```
Sub hoanvi(a As Integer, b As Integer)
Dim tg As Integer
tg = a
a = b
b = tg
End Sub
```

III. Truyền tham số cho chương trình con

Thông thường, CTC đều có tham số. Gọi CTC là thực hiện chính CTC đó với những giá trị tham số thực sự. Cách gọi như sau;

+ Đối với thủ tục: Tên_thủ_tục [ds_tham_số_thực_sự]

```
Ví dụ    Dim x as Integer, y as Integer
         x=1
         y=2
         Hoanvi x,y
```

+ Đối với hàm: Tên_hàm([ds tham số thực sự])

```
Ví dụ    UCLN(3,7)
```

Chú ý:

+ Truyền theo trị và truyền theo biến: Xét đoạn chương trình sau:

```
x = 3
y = 7
z = UCLN(x, y)
Text1.Text = x
```

Giá trị của x sau câu lệnh cuối cùng là bao nhiêu. Câu trả lời là 1. Biến x này đã bị thay đổi trong quá trình tính ước chung lớn nhất của x và y. Đây là điều không nên. Để điều này không xảy ra, ta sửa lại tiêu đề của hàm này như sau:

```
Function UCLN(byval a As Integer, byval b As Integer) As Integer
```

Truyền tham số x, y cho a,b lúc này là truyền theo trị (by value). Hoạt động của CTC sẽ không làm thay đổi giá trị của biến x, y.

Truyền tham số trong trường hợp trước (tham số khi khai báo không có từ khoá Byval hoặc khai báo bằng từ khoá ByRef) là truyền theo biến. Hoạt động của CTC sẽ làm thay đổi giá trị của biến x, y.

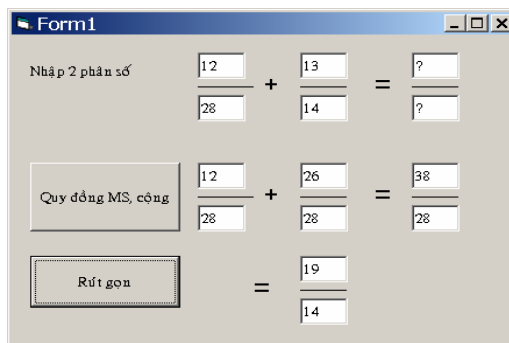
Vì vậy, nên hạn chế việc truyền tham số theo biến vì khó quản lý những hiệu ứng xảy ra khi CTC thực hiện.

+ Khi truyền tham số CTC sẽ kiểm tra chặt chẽ kiểu của tham số vì vậy phải chú ý sự phù hợp giữa kiểu dữ liệu của tham số thực sự và tham số hình thức.

Chú ý: Những chương trình con được viết trong VB có thể bổ sung vào cho các cơ sở dữ liệu trong Access và dùng như một hàm chuẩn trong Access. Xem hướng dẫn ở bài tập 6.2.

BÀI TẬP 6

Bài tập 6.1 Tạo ứng dụng hướng dẫn cho học sinh phổ thông kiểm tra việc cộng hai phân số.



Các đối tượng chính: txta, txtb, txtc, txtd, txta1, txtb1, txtc1, txtd1, txte1, txtf1, txtg, txtq;

Mã lệnh cho nút Quy đồng mẫu số, cộng:

```
Private Sub Command1_Click()
    Dim msc As Integer, a As Integer, b As Integer, c As Integer, d As Integer
    a = Val(txta.Text)
    b = Val(txtb.Text)
    c = Val(txtc.Text)
    d = Val(txtd.Text)
    msc = b * d \ ucln(b, d)
    Txta1.Text = a * msc \ b
    txtb1.Text = msc
    txtc1.Text = c * msc \ d
    txtd1.Text = msc
    txte1.Text = (Txta1.Text) + Val(txtc1.Text)
    txtf1.Text = msc
End Sub
```

Mã lệnh cho nút rút gọn

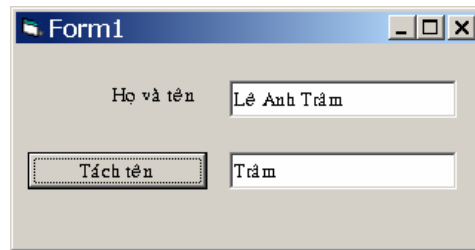
```
Private Sub Command2_Click()
    Dim tg As Integer, e As Integer, f As Integer
    tg = ucln(Val(txte1.Text), Val(txtf1.Text))
    e = Val(txte1.Text) \ tg
    f = Val(txtf1.Text) \ tg
```

```
If (f = 1) Or (e = 0) Then
    txtp.Text = e
Else
    txtp.Text = e
    txtq.Text = f
End If
End Sub
```

Cần thiết phải có hàm tính UCLN cụ thể đã nói ở trên với tiêu đề cho hàm như sau:

```
Function UCLN(byval a As Integer, byval b As Integer) As Integer
    Do While b <> 0
        r = a Mod b
        a = b
        b = r
    Loop
    UCLN = a
End Function
```

Bài tập 6.2 Viết hàm tách tên trong chuỗi họ và tên. Thiết kế ứng dụng sử dụng hàm này như sau:



Đối tượng chính txthoten, txtkq; nút lệnh Tách tên Command1

Hàm và mã lệnh:

```
Function Tachten(ByVal S As String) As String
    Dim i As Byte
    S = RTrim(S)
    i = Len(S)
    Do While Mid(S, i, 1) <> " "
        i = i - 1
    Loop
    Tachten = Mid(S, i + 1)
End Function

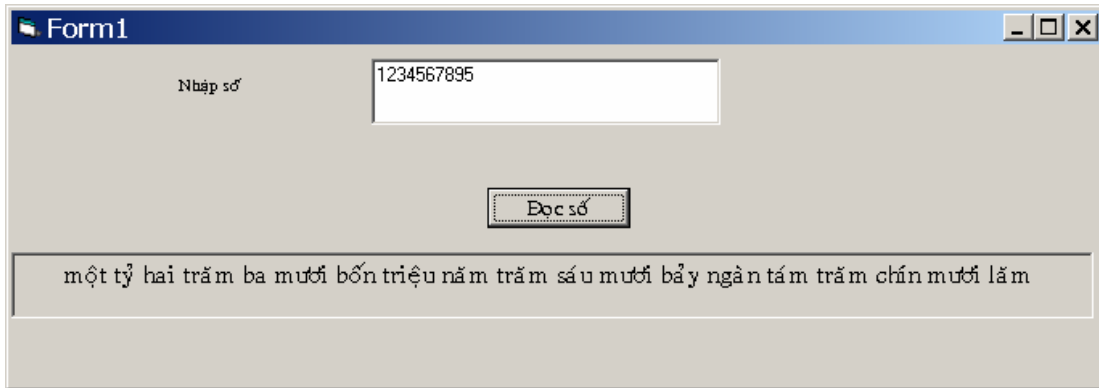
Private Sub Command1_Click()
    Txtkq.Text = Tachten(Txthoten.Text)
End Sub
```

Bây giờ ta sẽ dùng hàm này để tách riêng tên của khách hàng trong bảng Nhập Xuất Vật Tu (tập tin QUANLY.MDB) đã thực hành trong môn Access trước đây:

- + Mở CSDL QUANLY.MDB trong Access
- + Chọn Modules/New
- + Chép hàm Tachten trong VB vào Modules này, lưu lại
- + Trong Access, tạo Query lấy dữ liệu từ bảng Nhập Xuất Vật Tu có cột thứ nhất là HoTenKH, cột thứ hai là TenKH, là một biểu thức dùng hàm Tachten([HoTenKH]).

Tương tự, bạn có thể viết hàm đổi họ tên khách hàng thành dạng chuẩn để sử dụng cho bài tập số 2 (phần Form trong Access): khi nhập xong họ và tên khách hàng trên Form thì tự động đổi thành dạng chuẩn.

Bài tập 6.3 Tham khảo hàm đọc số thành chữ sử dụng trong các hoá đơn, phiếu tính tiền sau đây:



Function doc3so(n As Integer) As String

Dim chu(1 To 9) As String

chu(1) = "một "

chu(2) = "hai "

chu(3) = "ba "

chu(4) = "bốn "

chu(5) = "năm "

chu(6) = "sáu "

chu(7) = "bảy "

chu(8) = "tám "

chu(9) = "chín "

Dim ht, hc, dv As Byte

Dim s As String

ht = n \ 100

dv = n Mod 10

hc = ((n - dv) \ 10) Mod 10

s = ""

If ht > 0 Then

 s = chu(ht) + "trăm "

End If

Select Case hc

Case 0:

 If (ht > 0 And dv <> 0) Then

 s = s + "lẻ "

 End If

Case 1:

 s = s + "mười "

Case Else

 s = s + chu(hc) + "mười "

End Select

Select Case dv

Case 0:

Case 1:

 If (hc = 0 or hc=1) Then

 s = s + "một "

 Else

 s = s + "mốt "

 End If

Case 5:

 If hc = 0 Then

 s = s + "năm "

 Else

 s = s + "lăm "

 End If

Case Else

```

    s = s + chu(dv)
End Select
doc3so = s
End Function

Function docnso(n As Long) As String
Dim t As String
Dim s As String
Dim s1 As String
Dim i As Byte
Dim j As Byte
Dim n1 As Integer
Dim ch(1 To 4) As String
ch(1) = ""
ch(2) = "ngàn "
ch(3) = "triệu "
ch(4) = "tỷ "
t = Trim(Str(n))
Do While (Len(t) Mod 3 <> 0)
    t = "0" + t
Loop
s = ""
j = 1
For i = Len(t) - 2 To 1 Step -3
    s1 = Mid(t, i, 3)
    n1 = Val(s1)
    s = doc3so(n1) + ch(j) + s
    j = j + 1
Next
docnso = s
End Function

Private Sub Command1_Click()
Dim n As Long
Dim s As String
n = Val(so.Text)
s = docnso(n)
kq = s
End

```

BÀI 7. LISTBOX, COMBOBOX, SCROLLBAR

I. LISTBOX

ListBox là loại đối tượng điều khiển cho phép người dùng chọn một số mục trong các mục được liệt kê bởi một danh sách. Một số thuộc tính cơ bản:

+ List: chứa nội dung các mục của ListBox bằng một mảng, mỗi phần tử là một chuỗi có nội dung tương ứng với các mục. Mảng này được đánh chỉ số bắt đầu từ 0. Như vậy, để lấy mục thứ i ta dùng <Tên ListBox>.<List>(i).

Nhập danh sách: chọn thuộc tính List, gõ nội dung, ctrl+Enter; kết thúc bằng Enter.

+ ListIndex: cho biết số thứ tự của đang chọn (tính từ 0)

+ ListCount: cho biết số mục của danh sách

Một số phương thức cơ bản (được dùng khi viết mã lệnh)

+ AddItem: Thêm một mục “Nội dung” vào danh sách tại vị trí i, theo cú pháp

<Tên_ListBox>.<AddItem> <Nội dung> [i]

+ Remove: Xoá một mục thứ i khỏi danh sách, theo cú pháp

<Tên_ListBox>.<RemoveItem> <i>

+ Clear: Xoá toàn bộ danh sách, theo cú pháp

<Tên_ListBox>.<Clear> <i>

Ví dụ. Tạo một ứng dụng để chọn các môn học. Xem bài tập 7.1

II. COMBOBOX

Tương tự như ListBox nhưng Combobox cho nhập thêm dữ liệu.

Một số thuộc tính cơ bản: Combobox có tất cả những thuộc tính và phương thức của ListBox, ngoài ra nó có những thuộc tính, sự kiện đặc trưng sau:

+ Text: thuộc tính này chứa nội dung của mục được chọn trên Combobox

+ Click: sự kiện xảy ra khi người sử dụng nhấp chuột chọn một mục.

+ Change: sự kiện xảy ra khi người dùng nhập dữ liệu hoặc thay đổi nội dung của Combobox

Ví dụ: Xem bài tập 7.2

III. HSCROLLBAR, VSCROLLBAR

Các đối tượng này cho phép chọn một giá trị trong một khoảng xác định trước bằng các thanh cuộn ngang, thanh cuộn dọc.

Một số thuộc tính:

+ Value: giá trị số hiện thời, dùng để xử lý.

+ Min, Max: giá trị nhỏ nhất, lớn nhất của khoảng.

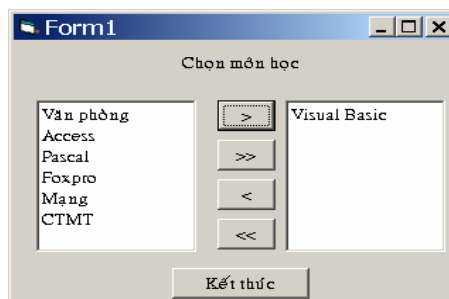
+ SmallChange: bước nhảy khi nhấp chuột vào đầu thanh cuộn

+ LargeChange: bước nhảy khi nhấp chuột vào giữa thanh cuộn

Ví dụ: Xem bài tập 7.2, 7.3

BÀI TẬP 7

Bài tập 7.1 Tạo ứng dụng chọn các môn học theo giao diện sau:



Các đối tượng chính: List1, List2, các nút lệnh

Mã lệnh cho nút >

```
Private Sub Command1_Click()
    Dim i As Byte
    i = List1.ListIndex
    If i >= 0 Then
        List2.AddItem List1.List(i)
        List1.RemoveItem i
    End If
End Sub
```

Mã lệnh cho nút >>

```
Private Sub Command2_Click()
    Dim i As Byte
    For i = 0 To List1.ListCount - 1
        List2.AddItem List1.List(i)
    Next
    List1.Clear
End Sub
```

Lúc bắt đầu, mặc nhiên chọn mục đầu tiên trong List1:

```
Private Sub Form_Load()
    List1.ListIndex = 0
End Sub
```

Các nút lệnh còn lại các bạn tự viết mã lệnh.

Bài tập 7.2 Tạo ứng dụng gồm một Combobox để chọn tháng và một HScrollBar chọn một năm từ 2000 đến 2020, chương trình cho biết số ngày trong tháng?



Các đối tượng chính: cmbThang, txtNam, HscNam, lblkq và hàm tính số ngày sau:

```
Function songay(t As Integer, n As Integer) As Integer
    Select Case t
        Case 1, 3, 5, 7, 8, 10, 12: songay = 31
        Case 4, 6, 9, 11: songay = 30
        Case 2:
            If (n Mod 400 = 0) Or (n Mod 4 = 0 And n Mod 100 <> 0) Then
                songay = 29
            Else
                songay = 28
            End If
    End Select
End Function

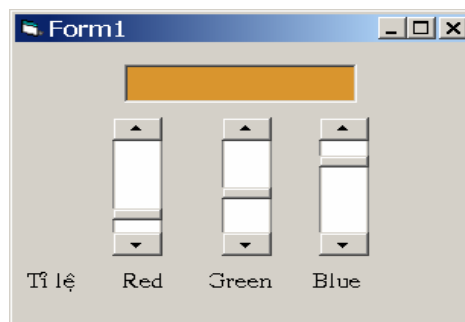
Private Sub Form_Load()
    Dim i As Byte
    For i = 0 To 11
        CmbThang.List(i) = i + 1
    Next
    CmbThang.ListIndex = 0
```

```
HScNam.Min = 2000
HScNam.Max = 2020
HScNam.SmallChange = 1
End Sub
```

```
Private Sub CmbThang_Click()
lblkq = "Tháng này có " & songay(Val(CmbThang.Text), Val(txtnam.Text))
End Sub
```

```
Private Sub HScNam_Change()
txtnam.Text = HScNam.Value
lblkq = "Tháng này có " & songay(Val(CmbThang.Text), Val(txtnam.Text))
End Sub
```

Bài tập 7.3 Một màu nào đó được tạo thành do sự pha trộn 3 màu cơ bản Red, Green và Blue. Trong VB, hàm RGB(x,y,z) để tính giá trị khi pha ba màu tương ứng nói trên với tỉ lệ x, y, z. Xây dựng một ứng dụng để xem mẫu màu khi pha bằng cách thay đổi giá trị của x,y,z bằng các VScrollBar.



Các đối tượng chính: Vscr1, Vscr2, Vscr3, txtkq

Chuẩn bị các giá trị

```
Private Sub Form_Load()
VScr1.Min = 0
VScr1.Max = 255
VScr1.Value = 127
VScr2.Min = 0
VScr2.Max = 255
VScr2.Value = 127
VScr3.Min = 0
VScr3.Max = 255
VScr3.Value = 127
End Sub
```

Xử lý khi dùng thanh trượt

```
Private Sub VScr1_Change()
txtkq.BackColor = RGB(VScr1.Value, VScr2.Value, VScr3.Value)
End Sub
Private Sub VScr2_Change()
txtkq.BackColor = RGB(VScr1.Value, VScr2.Value, VScr3.Value)
End Sub
Private Sub VScr3_Change()
txtkq.BackColor = RGB(VScr1.Value, VScr2.Value, VScr3.Value)
End Sub
```

BÀI 8. IMAGE, DRIVERLISTBOX, DIRLISTBOX, FILELISTBOX

I. IMAGE

Đối tượng cho phép hiển thị các tập tin ảnh.

Một số thuộc tính:

+ Picture: tên của tập tin ảnh cần hiển thị

+ Stretch: quy định có hay không việc co giãn cho vừa với kích thước đã định

Để xác lập tập tin cần hiển thị:

- Đặt trực tiếp vào thuộc tính Image hoặc

- Dùng lệnh: <tên_đtượng>.Picture = LoadPicture(“Đường dẫn\tên tập tin ảnh”)

Ví dụ: Xem bài tập 8.1

II. FILELISTBOX

Là một ListBox để hiển thị danh sách các tập tin của thư mục được chọn, mặc nhiên là của thư mục hiện hành.

Một số thuộc tính đặc trưng là:

+ Path: chứa đường dẫn đến thư mục được chọn trong FileListBox hay nói cách khác danh sách cây thư mục được thể hiện trong FileListBox do tham số này quy định.

+ Pattern: qui định dạng của tập tin được hiển thị. Ví dụ: Pattern=“*.jpg; *.bmp”

+ Filename: tên của tập tin được chọn (không kèm theo đường dẫn)

Chọn một tập tin là sự kiện Click

III. DIRLISTBOX

Là một ListBox dùng để hiển thị cấu trúc cây thư mục của ổ đĩa được chọn (mặc nhiên là ổ đĩa hiện hành). Ngoài các thuộc tính như của một ListBox, nó có thuộc tính đặc trưng là:

+ Path: chứa đường dẫn đến thư mục được chọn trong DirListBox hay nói cách khác danh sách cây thư mục được thể hiện trong DirListBox do tham số này quy định.

Chọn một thư mục là sự kiện Change

IV. DRIVERLISTBOX

Đối tượng này là một Combobox dùng để liệt kê các ổ đĩa hiện có trong máy tính.

Một số thuộc tính:

+ Drive: tên của ổ đĩa được chọn, mặc nhiên là ổ đĩa hiện hành (khi chưa chọn).

+ List: danh sách các ổ đĩa trong máy

+ ListIndex: số thứ tự của ổ đĩa được chọn (tính từ 0,...)

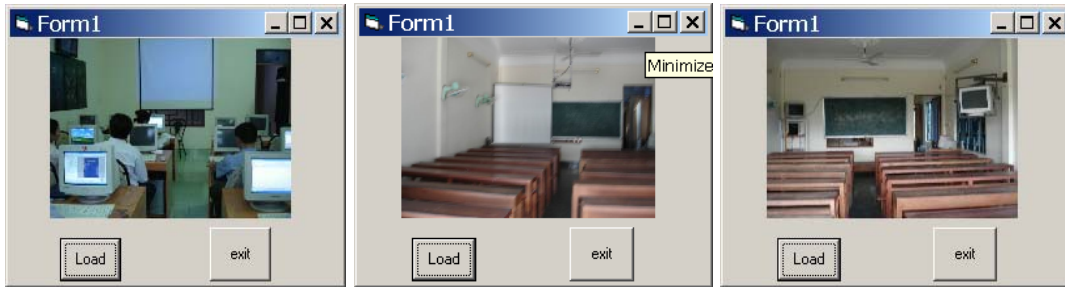
+ ListCount: tổng số các ổ đĩa.

Chọn một ổ đĩa là sự kiện Change

Ví dụ. Xem bài tập 8.2

BÀI TẬP SỐ 8

Bài tập 8.1 Tạo ứng dụng để xem ảnh các lớp học của TTTH 14-Bà Triệu, khi nhấp “Load” lần lượt xem các ảnh sau:



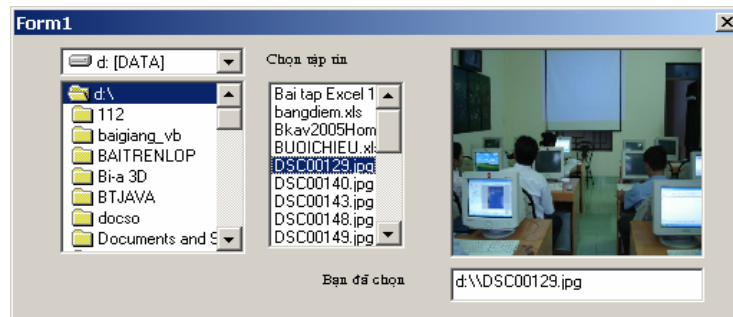
Các đối tượng chính: ImgAnh, command1, command2. Các tập tin ảnh có tên trong chương trình phải có sẵn trên đĩa, có thể dùng tập tin khác để thay thế.

```
Dim n As Byte
Private Sub Form_Load()
    n = 0
End Sub

Private Sub Command1_Click()
    If n < 3 Then
        n = n + 1
    Else
        n = 1
    End If
    Select Case n
        Case 1: ImgAnh.Picture = LoadPicture("C:\dsc00129.jpg")
        Case 2: ImgAnh.Picture = LoadPicture("C:\dsc00148.jpg")
        Case 3: ImgAnh.Picture = LoadPicture("C:\dsc00149.jpg")
    End Select
End Sub

Private Sub Command2_Click()
    ImgAnh.Picture = LoadPicture("")
End Sub
End Sub
```

Bài tập 8.2 Tạo một ứng dụng cho phép lựa chọn tập tin ảnh trong một thư mục của ổ đĩa nào đó trong máy và load tập tin ảnh đó lên form theo giao diện như sau:



Các đối tượng chính Drive1, Dir1, File1, image1, txtkq

```
Private Sub Drive1_Change()
Dir1.Path = Drive1.Drive
End Sub

Private Sub Dir1_Change()
File1.Path = Dir1.Path
File1.Pattern = "*.jpg;*.bmp"
End Sub

Private Sub File1_Click()
Dim s As String
s = Dir1.Path & "\" & File1.FileName
Image1.Picture = LoadPicture(s)
txtkq = s
End Sub
```

BÀI 9. ỨNG DỤNG CÓ NHIỀU FORM, TẠO MENU

I. ỨNG DỤNG CÓ NHIỀU FORM.

Một ứng dụng có thể chứa nhiều Form, có thể chuyển giao dữ liệu từ form này qua form khác hoặc đứng ở form này điều khiển form kia. Các thao tác thường gặp bao gồm:

+ Thêm một form vào ứng dụng: Project/Add Form; sau đó ta có thể chọn để thêm một form mới (New) hoặc thêm một form sẵn có (Existing).

+ Quy định form được mở đầu tiên trong ứng dụng: Project/ Project properties; trong mục Startup Object chọn tên form được mở đầu tiên.

+ Mở một form khác (form2): dùng <Tên_Form2>.Show

+ Đóng một form khác: dùng Unload <Tên_Form2>

+ Truy xuất dữ liệu của form khác: <Tên_Form2>.<đối_tượng>.<thuộc_tính>

Ví dụ. Xem bài tập 9.1

II. TẠO MENU

Một menu thường chứa nhiều mục chọn, mỗi mục chọn là một đối tượng có hai thành phần cơ bản là Name và Caption. Ở đây, Name là thuộc tính quan trọng hơn, liên quan đến mã lệnh; Caption đơn thuần chỉ là thuộc tính để xuất hiện nội dung của mục chọn không liên quan đến mã lệnh. Vì vậy, khi thay đổi Name phải sửa lại mã lệnh còn thay đổi Caption thì không cần.

Một số thuộc tính khác:

+ Enabled: cho phép chọn (true) hay không (false) mục này trên menu.

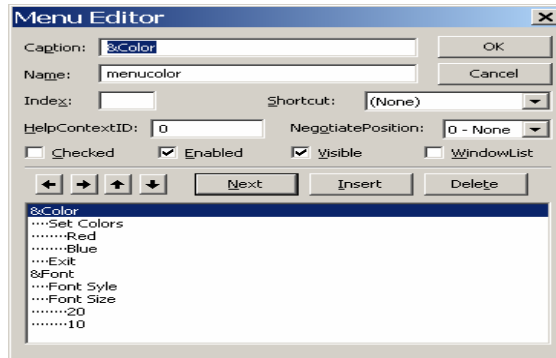
+ Visible: cho phép hiển thị hay không mục này trên menu.

Phương thức duy nhất cho mục chọn là Click.

Cách thiết kế menu: Menu có thể có nhiều cấp. Tổ chức các cấp của menu như thế nào phụ thuộc lúc thiết kế giao diện cho menu. Thiết kế bằng Menu Editor như sau:

+ Tools/Menu Editor

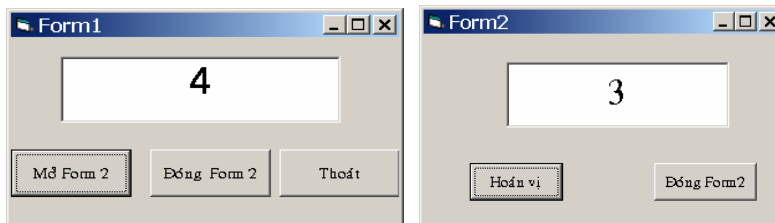
+ Khai báo thuộc tính cho từng mục chọn. Các cấp được thiết kế như phần dưới của bảng:



Ví dụ. Xem bài tập 9.2

BÀI TẬP SỐ 9

Bài tập 9.1 Tạo ứng dụng gồm hai form mỗi form có textbox để nhập dữ liệu. Form1 có chức năng đóng, mở form2. Form 2 có chức năng hoán vị giá trị đã nhập.



Các đối tượng chính:

+ Form1: text1, các nút lệnh Command1, Command2, Command3 ứng với mở form2, đóng form2 và thoát.

+ Form2: text1, các nút lệnh Command1, Command2 ứng với hoán vị, đóng form2.

Mã lệnh trên Form1

```
Private Sub Command1_Click()
    Form2.Show
End Sub
```

```
Private Sub Command2_Click()
    Unload Form2
End Sub
```

```
Private Sub Command3_Click()
    Unload Me
End Sub
```

Mã lệnh trên Form2:

```
Private Sub Command1_Click()
    Dim tg As String
    tg = Form1.text1.Text
    Form1.text1.Text = Form2.text1.Text
    Form2.text1.Text = tg
End Sub
```

```
Private Sub Command2_Click()
    Unload Me
End Sub
```

Bài tập 9.2 Thiết kế Menu trên form để chọn màu nền Blue, Red cho form; chọn kích thước cho dòng chữ văn bản mẫu là 10,20 như sau:



Dùng Menu Editor để thiết kế như hình vẽ trong bài lý thuyết.

Mã lệnh (để ý trong chương trình có một mục có Name là MnuRed, tên này do người thiết kế tự đặt)

```
Private Sub Menu10_Click()
    menu20.Enabled = True
    Menu10.Enabled = False
    txt1.FontSize = 10
End Sub
```

```
Private Sub menu20_Click()
    menu20.Enabled = False
    Menu10.Enabled = True
    txt1.FontSize = 20
End Sub
```

```
Private Sub menuBlue_Click()
    menuBlue.Enabled = False
    MnuRed.Enabled = True
    Form1.BackColor = vbBlue
End Sub
```

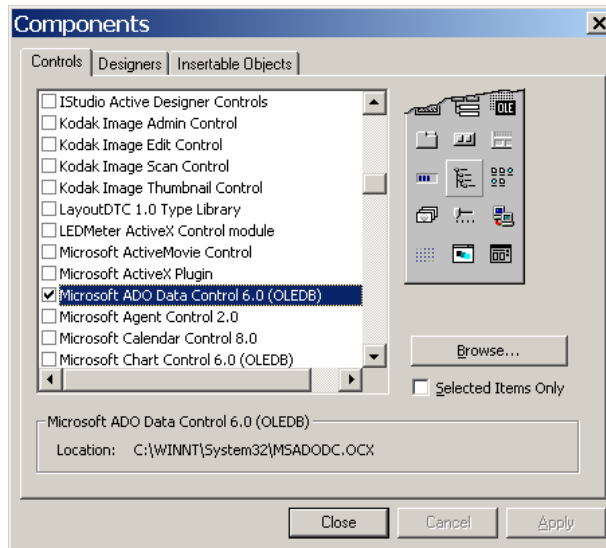
```
Private Sub MnuRed_Click()
    MnuRed.Enabled = False
    menuBlue.Enabled = True
    Form1.BackColor = vbRed
End Sub
Private Sub MenuExit_Click()
    End
End Sub
```

BÀI 10. ĐỐI TƯỢNG ĐIỀU KHIỂN ADO

I. Bổ sung điều khiển ADO vào hộp công cụ.

ADO (ActiveX Data Object) là điều khiển dùng để kết nối giữa một form trong VB với một cơ sở dữ liệu. ADO cung cấp các chức năng duyệt qua các mẫu tin và truy xuất dữ liệu trên cơ sở dữ liệu.

Thêm đối tượng ADO hay đầy đủ là ADODC (ADO Data Control) vào hộp công cụ: Project/Components/Microsoft ADO Data Control 6.0



Kết quả: xuất hiện biểu tượng adodc trong hộp công cụ. Chọn công cụ này và vẽ lên form ta sẽ có đối tượng ADODC trên form.

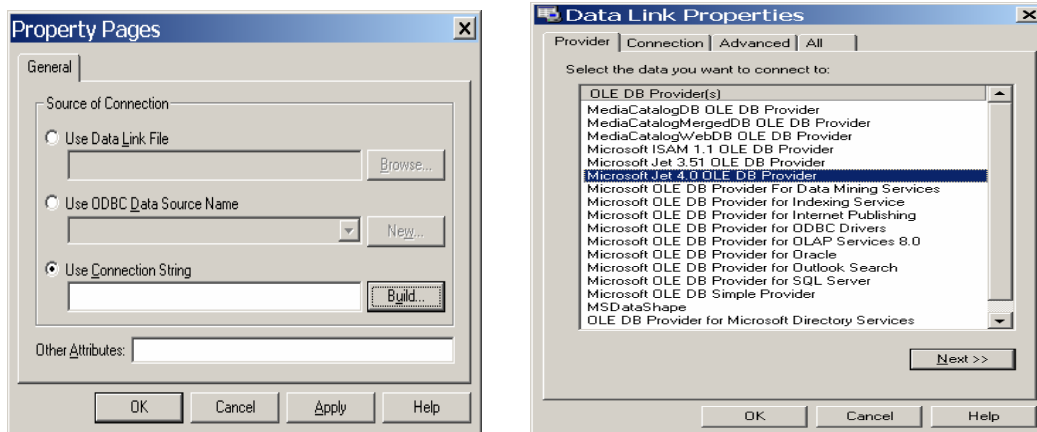
II. Các thuộc tính cơ bản của ADODC.

1. Thuộc tính ConnectionString: quy định kết nối CSDL với đối tượng ADODC với hai thông tin quan trọng gồm: kiểu kết nối và CSDL được kết nối, xác lập qua các bước sau:

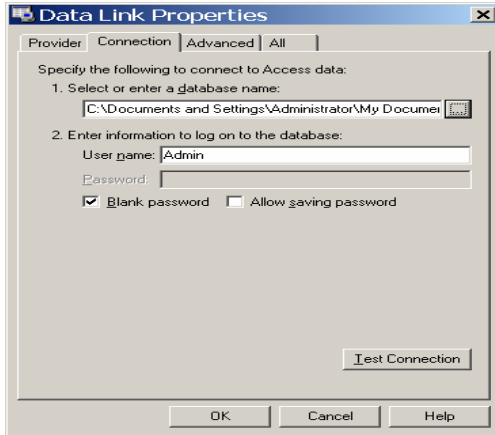
- + Kích chuột vào thuộc tính ConnectionString, xuất hiện hộp thoại Property Page.
- + Chọn Build...(H.1)
- + Chọn Microsoft Jet 4.0 OLE DB Provider để kết nối với CSDL trong Access 2002 (H.2)
- + Chọn tập tin .mdb cần kết nối (H.3)

Kết quả: Dòng “Provider=Microsoft.Jet.OLEDB.4.0;Data Source=C:\Documents and Settings\Administrator\My Documents\nhanvien.mdb;Persist Security Info=False” xuất hiện trong Use Connection String nếu kết nối với tập tin nhanvien.mdb.

2. Thuộc tính RecordSource: quy định cụ thể Table, Query của tập tin .mdb (hoặc câu lệnh SQL) sẽ dùng để lấy dữ liệu.

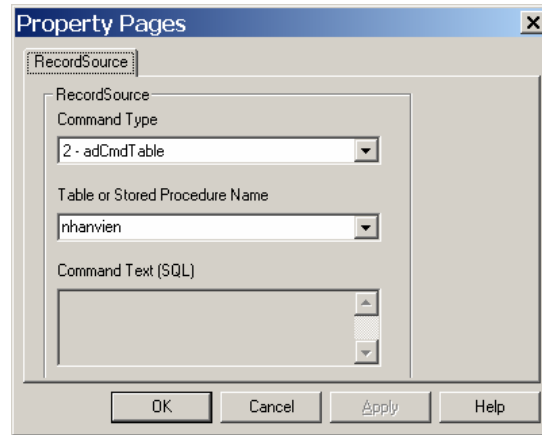


H.1



H.3

H.2



H.4

- + Chọn RecordSource trong cửa sổ property, xuất hiện Property Page (H.4)
- + Chọn 2-adCmdTable dữ liệu là Table hoặc Query trong Access (chọn 1-adCmdText dữ liệu lấy bằng một câu lệnh Sql).
- + Chọn tên Table cần kết nối.

III. Truy xuất dữ liệu

Xong mục II, chúng ta đã qui định việc kết nối dữ liệu đến một Table hay một Query cụ thể của một tập tin .mdb, công việc tiếp theo là truy xuất dữ liệu đến từng trường trên Table (query) này. Đơn giản nhất là dùng các đối tượng điều khiển đã biết như Textbox,...liên kết với các trường. Tạo Textbox và qui định hai thuộc tính sau của Textbox:

- + Thuộc tính DataSource: chọn ADODC ứng với tập .mdb cung cấp dữ liệu
- + Thuộc tính DataField: chọn trường truy xuất dữ liệu.

Ví dụ. Xem bài tập 10.1

IV. Một số phương thức và sự kiện của ADODC

- + Refresh: cập nhật ngay những thuộc tính của ADODC.

Các phương thức sau đây thuộc thành phần RecordSet của ADODC:

- + Tên_ADODC.RecordSet.MoveFirst: chuyển đến mẫu tin đầu tiên.
- + Tên_ADODC.RecordSet.MoveLast: chuyển đến mẫu tin cuối cùng.
- + Tên_ADODC.RecordSet.MoveNext: chuyển đến mẫu tin kế tiếp.
- + Tên_ADODC.RecordSet.MovePrevious: chuyển đến mẫu tin phía trước.
- + Tên_ADODC.RecordSet.Bof: kiểm tra mẫu tin hiện hành có là mẫu tin đầu tiên hay không, nếu có thì trả về giá trị true và ngược lại.
- + Tên_ADODC.RecordSet.Eof: kiểm tra mẫu tin hiện hành có là mẫu tin cuối cùng hay không, nếu có thì trả về giá trị true và ngược lại.
- + Tên_ADODC.RecordSet.RecordCount: cho biết tổng số mẫu tin.

+ Tên_ADODC.RecordSet.AbsolutePosition: chỉ biết vị trí hiện tại của mẫu tin hiện hành.

+ Tên_ADODC.RecordSet.AddNew: thêm mẫu tin trống vào bảng

+ Tên_ADODC.RecordSet.Update: Cập nhật dữ liệu vào bảng.

+ Tên_ADODC.RecordSet.Delete: Xoá mẫu tin hiện hành.

Các sự kiện sau đây thường dùng khi lập trình:

+ Tên_ADODC.RecordSet.MoveComplete: xảy ra khi một mẫu tin nào đó trở thành mẫu tin hiện hành (sau khi thực hiện các phương thức di chuyển mẫu tin hiện hành)

+ Tên_ADODC.RecordSet.WillChangeRecord: xảy ra trước khi di chuyển mẫu tin hiện hành hoặc Update dữ liệu (thường dùng để kiểm tra tính hợp lệ của dữ liệu trước khi cập nhật hoặc thực hiện một thao tác khác)

+ Tên_ADODC.RecordSet.RecordChangeComplete: xảy ra sau khi một mẫu tin nào đó hoàn tất việc thay đổi dữ liệu trong CSDL (dùng để thay đổi giao diện liên quan đến mẫu tin mới, cập nhật thông tin vừa thay đổi trong CSDL lên form)

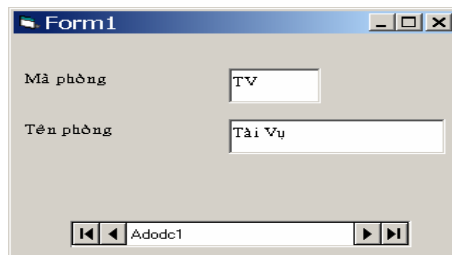
Ví dụ. Xem bài tập 10.2

BÀI TẬP SỐ 10

Bài tập 10.1 Mở bảng Nhân viên trong bài tập môn Access, thêm vào vùng MaPhong. Tạo thêm bảng DMPHong như sau:

| | |
|---------|------------|
| Maphong | Tenphong |
| HC | Hành chính |
| KD | Kinh doanh |
| TV | Tài vụ |

Tạo ứng dụng liên kết để xem thông tin trong bảng Danh mục phòng theo giao diện như sau:



Hướng dẫn: Thực hiện các bước như trong bài học. Sau đó tạo Form và các đối tượng chính: txtmaphong, txttenphong liên kết với các trường mã phòng, tên phòng của bảng.

Bài tập 10.2 Tạo ứng dụng để thực hiện các thao tác trên bảng nhân viên như đã gợi ý trên form sau đây:

Các đối tượng chính:

- + adodc1 liên kết với bảng nhân viên, thuộc tính Visible bằng false.
- + txtho, txtten, txtngsinh, chkgioitinh, cmbmaphong lấy dữ liệu tại các trường tương ứng với tên của các đối tượng.
- + Các nút lệnh CmdDau, CmdCuoi, CmdTruoc, CmdSau, CmdThem, CmdLuu, CmdXoa, CmdThoat; các nút này ứng với các Command1, command2,...

Mã lệnh cụ thể cho các nút:

```
Private Sub Command1_Click()
    Adodc1.Recordset.MoveFirst
End Sub
```

```
Private Sub Command2_Click()
    Adodc1.Recordset.MoveLast
End Sub
```

```
Private Sub Command3_Click()
    If Not Adodc1.Recordset.BOF() Then
        Adodc1.Recordset.MovePrevious
    Else
        Adodc1.Recordset.MoveLast
    End If
End Sub
```

```
Private Sub Command4_Click()
    If Not Adodc1.Recordset.EOF() Then
        Adodc1.Recordset.MoveNext
    Else
        Adodc1.Recordset.MoveFirst
    End If
End Sub
```

```
Private Sub Command5_Click()
    Adodc1.Recordset.AddNew
    txtho.SetFocus
End Sub
```

```
Private Sub Command6_Click()
    Adodc1.Recordset.Update
End Sub
```

```
Private Sub Command7_Click()
    Adodc1.Recordset.Delete
    Adodc1.Recordset.MoveNext
End Sub
```

```
Private Sub Adodc1_MoveComplete(ByVal adReason As ADODB.EventReasonEnum, ByVal
pError As ADODB.Error, adStatus As ADODB.EventStatusEnum, ByVal pRecordset As
ADODB.Recordset)
Adodc1.Caption = "Hồ sơ thứ " & Adodc1.Recordset.AbsolutePosition & "/" &
Adodc1.Recordset.RecordCount
End Sub
```

BÀI 11. MỘT SỐ HỖ TRỢ KHÁC CHO QUẢN TRỊ CSDL

I. DATAGRID

Ô điều khiển này dùng để thể hiện dữ liệu của các mẫu tin cùng lúc dưới dạng bảng.

Các thuộc tính cơ bản:

- + DataSource: quy định nguồn dữ liệu lấy từ ADODC nào.
- + AllowAddNew: cho phép (true) hay không (false) được thêm mẫu tin từ DataGrid
- + AllowDelete: cho phép (true) hay không (false) được xoá mẫu tin.
- + AllowUpdate: cho phép (true) hay không (false) được sửa đổi dữ liệu cho mẫu tin từ DataGrid.

Thao tác: không có sẵn trên ToolBox, ta phải thêm vào bằng dãy thao tác sau

- + Project/Components/Microsoft DataGrid Control 6.0
- + Chọn và vẽ một DataGrid lên form, trước đó trên form phải xác lập một ADODC liên kết với một CSDL
- + Xác lập thuộc tính DataSource
- + Thêm tên các trường: chuột phải lên DataGrid chọn Retrieve Fields
- + Sửa đổi: chuột phải lên DataGrid chọn Edit
- + Những chi tiết khác có thể thay đổi bằng cách chuột phải lên DataGrid chọn Properties.

Ví dụ. Xem bài tập 11.1

II. Lấy dữ liệu thông qua câu lệnh SQL

1. Khái niệm, cú pháp:

SQL (Structured Query Language) là ngôn ngữ để định nghĩa và thao tác dữ liệu trong đa số hệ QTCSDL.

SQL có chức năng rất rộng, tuy nhiên trong khuôn khổ của giáo trình này chúng ta chỉ quan tâm đến khả năng chọn lọc để rút ra dữ liệu thoả mãn một số điều kiện.

Cú pháp. SELECT <ds trường> FROM <bảng> [WHERE <điều kiện>]

Cho phép lấy dữ liệu gồm các <ds trường> của các mẫu tin từ <bảng> với điều kiện các mẫu tin này thoả mãn <điều kiện>

Nếu cần lấy hết tất cả các trường thì thay <ds trường> bằng dấu *

Ví dụ: SELECT * FROM nhanvien WHERE maphong = "TV"

Chú ý:

+ Trường có thể thay bằng biểu thức dạng <biểu thức> [As <Tên>]

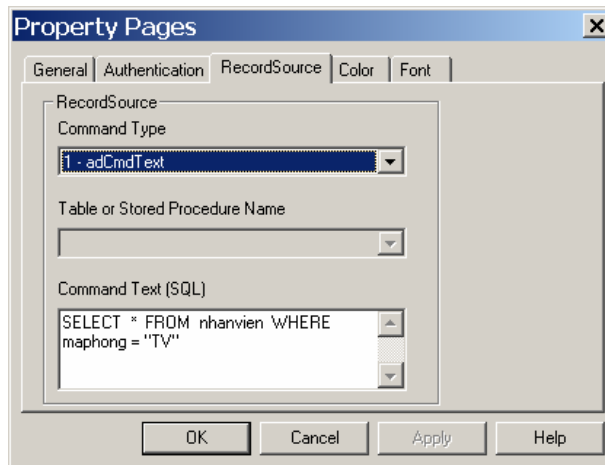
Ví dụ. SELECT ho, ten, luong*0.05 As BaoHiem FROM nhanvien

+ Ngoài các phép toán thường dùng như >, <, >=, <=, =, <>; các phép toán logic AND, OR, NOT trong điều kiện người ta thường dùng BETWEEN, LIKE có cú pháp giống như trong Access. Để ý rằng dấu % là kí tự đại diện thay thế cho một nhóm kí tự tại vị trí của nó. Ví dụ: Ten LIKE "% n" để chọn những người có kí tự cuối trong tên là "n".

2. Quy định dữ liệu lấy qua SQL

Thay vì lấy trực tiếp dữ liệu từ bảng, linh hoạt hơn trong trường hợp cần lọc dữ liệu chúng ta lấy dữ liệu thông qua câu lệnh SQL.

Cách 1: Kích chuột phải lên ADODC chọn Properties, đặt Command Type là 1-adCmdText, câu lệnh SQL được gõ vào như hình vẽ.



Cách 2: Quy định hai thông tin trên bằng mã lệnh

Ví dụ:

```
Adodc1.CommandType = adCmdText
```

```
Adodc1.RecordSource = "select * from nhanvien where maphong='"&"TV" &"'"
```

Chú ý dấu nháy đơn và nháy kép trong cách viết ='"&"TV" &"' ' ' "

Câu lệnh SQL là một chuỗi, vì vậy khá linh hoạt để có thể biểu diễn điều kiện phức tạp. Xem bài tập 11.3

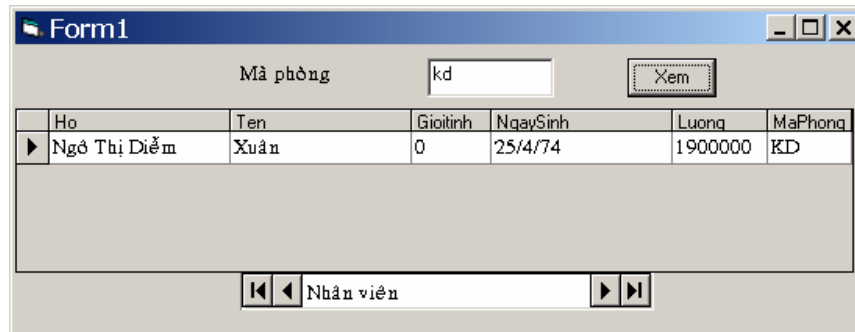
BÀI TẬP SỐ 11

Bài tập 11.1 Tạo một ứng dụng để liệt kê danh sách các phòng trong tập tin danh mục phòng ở dạng bảng như sau:

Hướng dẫn: Lần lượt thực hiện các thao tác như trong phần lý thuyết



Bài tập 11.2 Tạo ứng dụng để liệt kê danh sách nhân viên trong tập tin nhân viên ở dạng bảng theo mẫu sau: Khi chạy chương trình nhập mã phòng, chỉ xuất hiện các nhân viên trong phòng cần xem.



Các đối tượng chính:

- + Adodc1 liên kết với bảng nhân viên như bài tập 11.1
- + Tạo một DataGrid1 hiển thị các trường của bảng nhân viên
- + Tạo txttk để nhập mã phòng cần xem
- + Nút lệnh “xem” ứng với Command1 và mã lệnh sau:

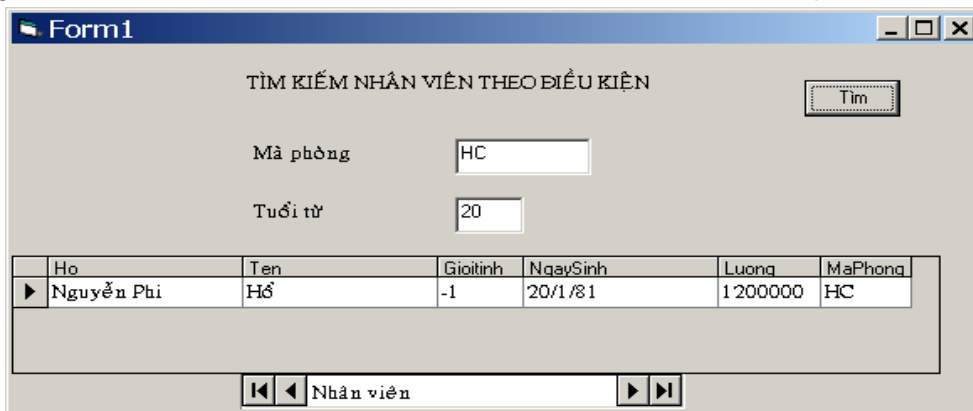

```
Private Sub Command1_Click()
    Adodc1.CommandType = adCmdText
    Adodc1.RecordSource = "select * from nhanvien where maphong LIKE '" & "%" & txttk.Text & """"
    Adodc1.Refresh
End Sub

Private Sub Form_Load()
    Form1.Show
    txttk.SetFocus
End Sub
```

Bài tập 11.3 Tạo ứng dụng “tìm kiếm nhân viên” theo điều kiện. Điều kiện gõ vào ô nào thì kết hợp để tìm nhân viên thỏa các điều kiện đó.

Các đối tượng chính:

- + Adodc1 liên kết với bảng nhân viên như bài tập 11.1
- + Tạo một DataGrid1 hiển thị các trường của bảng nhân viên
- + Tạo txtmaphong để nhập mã phòng cần xem
- + Tạo txttuoi1 để nhập độ tuổi
- + Nút lệnh “Tìm” ứng với Command1 và mã lệnh sau:



```

Private Sub Command1_Click()
Dim dk As String, dk1 As String, dk2 As String, sql As String
dk = ""
Adodc1.CommandType = adCmdText
dk1 = "maphong = " & txtmaphong.Text & ""
dk2 = " year(date())-year(ngaysinh)>=" & Val(txttuoi1.Text) & ""
dk = dk1 & " and " & dk2
sql = "select * from nhanvien where " & dk
Adodc1.RecordSource = sql
Adodc1.Refresh
End Sub

Private Sub Form_Load()
Form1.Show
txtmaphong.SetFocus
End Sub
    
```

Tương tự như vậy, các bạn có thể bổ sung thêm các điều kiện khác.

Công cụ của Microsoft Visual Studio cho phép tạo một bộ cài đặt hoàn chỉnh cho một ứng dụng tùy ý. Vào Start/Programs/ Microsoft Visual Studio 6.0/ Microsoft Visual Studio 6.0 Tools/ Package and Deployment Wizard và làm theo hướng dẫn.

TÀI LIỆU THAM KHẢO

1. Giáo trình Lập trình Visual Basic, TS. Trần Thiên Thành, ĐH Quy Nhơn.
2. Giáo trình Lập trình ứng dụng Visual Basic, Đặng Thế Khoa, ĐHQG TP HCM
3. Giáo trình điện tử của Bộ GD-ĐT <http://ebook.edu.net.vn>