Chương 4
LẬP TRÌNH TRÊN VISUAL BASIC

4.1. Khai báo biến

4.1.1. Biến

Biến dùng để chứa dữ liệu tạm thời cho tính toán so sánh và các hoạt động khác ta dùng toán tử (=) để tính toán và chứa giá trị vào biến.

Ví dụ:

```
Applesold = 10
Applesold = Applesold + 1
```

Dấu bằng là toán tử gán, không phải toán tử so sánh (=), giá trị 10 được gán cho biến Applesold.

4.1.2. Khai báo biến

Tuỳ theo phạm vi biến cần sử dụng chúng ta có thể dùng các cấu trúc lệnh khác nhau để khai báo biến.

Dùng lệnh Dim:

```
Dim (tên biến) As (kiểu dữ liệu).
Dim (tên biến) (kiểu gán chuỗi).
```

Ví dụ:

```
Dim a, b As integer
Dim a%, b%
```

Biến khai báo trong thủ tục chỉ tồn tại khi thủ tục thi hành. Nó sẽ biến mất khi thủ tục chấm dứt. Giá trị biến trong thủ tục là cục bộ đối với thủ tục đó. Nghĩa là không thể truy cập biến từ bên ngoài thủ tục. Nhờ đó ta có thể dùng tên biến cục bộ trong thủ tục khác nhau.

Kiểu dữ liệu khai báo trong Dim có thể là kiểu cơ bản như Integer, String hoặc Currentcy.Ta cũng có thể dùng đối tượng của VB (Như Object, Form, TextBox) hoặc của các ứng dụng khác.

Khai báo phần Declaration của một Modul nghĩa là biến đó tồn tại và có tầm hoạt động trong Modul đó.

Khai báo biến với từ khóa Public nghĩa là biến đó tồn tại và có tầm hoạt động trong toàn bộ ứng dụng.

Khai báo biến cục bộ với từ khóa Static, nghĩa là mặc dù biến đó biến mất khi thủ tục chấm dứt, nhưng giá trị của nó vẫn được giữ lại để tiếp tục hoạt động khi thủ tục được gọi trong lần sau.

4.1.3. Khai báo ngầm

Nghĩa là ta không cần khai báo tường minh, trước khi sử dụng biến.

Ví dụ:

Funtion Safe Sqr (num)

Temp Val = Abs (mim)

Safe Sqr = Sqr (Temval)

End Funtion

4.1.4. Khai báo tường minh

Để tránh những lỗi chương trình xảy ra do nhập sai tên biến, chúng ta có thể sử dụng chế độ khai báo tường minh. Với chế độ này, mỗi biến sử dụng cần phải được khai báo trước. Những biến nào chưa khai báo, VisualBasic sẽ báo lỗi khi thực thi chương trình. Ta có thể sử dụng một trong hai cách dưới đây để sử dụng chế độ khai báo biến tường minh.

Cách 1:

Chọn Tools chọn Option, chọn Tab Editor và đánh dấu vào tuỳ chọn Require Variable Decleration.Từ thời điểm này các màn hình lớp hay thư viện được tạo ra sẽ mặc nhiên có sẵn dòng lệnh Option Explicit trong phần Declaration với các màn hình giao tiếp (Form), lớp class hay thư viện (Module) đã được tạo trước đó, chúng ta sẽ tự thêm vào dòng lệnh này như cách 2.

Cách 2:

Trong cửa sổ đặt dòng lệnh sau đây Option Explicit ở đầu phần declarations của màn hình giao tiếp (Form), lớp (Class) hay thư viện Module.

Tầm hoạt động của biến:

	Private	**Public**
Thủ tục	Biến chỉ tồn tại và hoạt động trong thủ tục	Không có
Modul	Biến chỉ tồn tại và hoạt động trong Modul	Biến tồn tại và hoạt động trên Modul

4.1.5. Khai báo biến STATIC

Kiểu dữ liệu	Kích thước		Ký hiệu
Integer	2 byte	-32,768 -> +32,767	%
Long (Long integer)	4 byte	-2,147,483,648 -> 2,147,483,647	&
Single (Single - precision, floating - point)	4 byte	Giá trị aõm: -3.402823E38 -> - 1.401298E -45 Giá trị sử dụng: 1.401298E -> 3.402823E38	!
Double (double-precision, floating-point)	8 byte	- 1.79769313486232 E308-> - 4.94065645841247E - 324 Với các giá trị âm: 4.94065645841247E - 324 1.79769313486232 E308 Với các giá trị dương	#
Currency	8 byte	- 922,337,203,685,477.5808 -> 922,337,203,685,477.5808.	@
String	1 byte mỗi character	0 ->65,500 byte	$
Varriant		+ Dùng để biểu diễn các giá trị số (với tầm vực double) hay các ký tự	

- Để khai báo tất cả các biến cục bộ trong một thủ tục Static, ta đặt từ khoá Static vào thủ tục.
- Static Function Runningtotal (mim) VB sẽ hiểu là tất cả biến khai báo trong thủ tục này đều là Static, là cho chúng được khai báo là Private, là Dim hoặc thậm chí là khai báo ngầm.
- Từ khóa Static có thể đặt ở đầu thủ tục Sub hoặc Funtion, kể cả thủ tục xử lý sự kiện hoặc các hàm Private.

4.1.6. Các toán tử cơ sở

4.1.6.1. Toán tử gán

Đây là toán tử cở sở của hầu hết các ngôn ngữ lập trình.

Toán tử dùng để gán giá trị cho các biến có kiểu dữ liệu cơ sở trong VisualBasic là dấu (=), cú pháp chung lệnh gán có dạng sau:

<Tên biến> = <Biểu thức>

Biểu thức ở bên phải của cú pháp trên có thể là một giá trị hằng, một biến hay một biểu thức tính toán. Khi đó VisualBasic sẽ thực hiện việc tính giá trị của biểu thức trước rồi sau đó mới gán giá trị có được cho biến.

Ví dụ: Dòng lệnh gán sau đây sẽ tăng giá trị biến k lên thêm một (k=k+1).

Thông thường, giá trị của biểu thức và biến trong cú pháp lệnh gán phải cùng kiểu dữ liệu, tuy nhiên chúng ta vẫn có thể gán biểu thức số vào một biến kiểu chuỗi. Trong trường hợp này, VisualBasic sẽ tự động đổi giá trị biểu thức thánh chuỗi sau đó mới gán vào biến.

Với các biến có kiểu dữ liệu tổng quát, để gán giá trị cho biến chúng ta phải dùng lệnh Set theo cú pháp dưới đây:

Set <Tên biến>= <Biểu thức>

4.1.6.2. Toán tử tính toán

Những toán tử trình bày trong phần này là những toán tử cơ sở liên quan đến các giá trị số, bao gồm số nguyên và số thực.

- **Các toán tử số nguyên**

+ : Cộng (Cộng hai toán hạng)

- : Trừ (Trừ hai toán hạng)

***** : Nhân (Dùng để nhân hai toán hạng)

\ : Chia (Chia nguyên bỏ phần thập phân)

Ví dụ 1: X=5\2 thì X sẽ nhận giá trị là 2

/: Chia làm tròn

Ví dụ 2: Dim x as Double

X=3/2

^ : Luỹ thừa (Dùng để tính luỹ thừa)

MOD: Chia lấy phần dư

- **Các toán tử số thực**

  +: Cộng (cộng hai toán hạng)

  - : Trừ (Trừ hai toán hạng)

  *: Nhân (Dùng để nhân hai toán hạng)

  / : Chia (chia hai toán hạng)

  ^: Luỹ thừa (Dùng để tính luỹ thừa)

4.1.6.3. Toán tử nối chuỗi

Đây là toán tử cơ sở dùng để nối các dữ liệu lại với nhau.

Ví dụ: Trong dòng lệnh dưới đây:

S = "Viasual" & "Basic"

Biến chuỗi S sẽ có giá trị là "VisualBasic".

Tương tự như lệnh gán chuỗi với các biểu thức số, Visualbasic sẽ tự động thực hiện việc chuyển dữ liệu chuỗi thành số trước rồi sau đó mới nối với dòng tiếp theo.

S=S & 1 =>Giá trị của biến S sau lệnh gán sẽ là "VisualBasic 1".

4.1.6.4. Toán tử so sánh

Như mọi ngôn ngữ lập trình khác,VisualBasic cũng sử dụng các toán tử so sánh như:

Kí hiệu	Ý nghĩa	Ví dụ
=	So sánh bằng	A=B
<	So sánh nhỏ hơn	A<B
<=	So sánh nhỏ hơn hay bằng	A<=B
<>	So sánh khác	A<>B
>	So sánh lớn hơn	A>B
>=	So sánh lớn hơn hay bằng	A=B

Giá trị kết quả của một biểu thức so sánh sẽ thuộc kiểu luận lý (True hay False). Khi cần kết nối các biểu thức so sánh lại với nhau chúng ta có thể dùng các toán tử luận lý And, Or.

4.2. Tập lệnh cơ bản của VB

4.2.1. Các kiểu dữ liệu

Việc khai báo các kiểu dữ liệu nhằm kiểm soát nội dung của dữ liệu, cho phép tối ưu hoá về tốc độ và kích cỡ chương trình. VisualBasic dùng kiểu Variant như là kiểu mặc định (kiểu bất kì), khi dùng Variant ta không phải chuyển đổi giữa các kiểu dữ liệu, VB tự làm việc đó.

- **Kiểu số**

Integer, Long, Double và Currency kiểu số chứa ít vùng nhớ hơn kiểu Variant. Tất cả biến kiểu số có thể được gán cho nhau và biến Variant VisualBasic làm tròn thay vì chặt bỏ phần thập phần trước khi gán nó cho số Integer. Kiểu Integer tốn ít nhất vùng dữ liệu, nó thường dùng làm biến đếm trong vòng lặp ForNext. Kiểu Singe, Double, Currrency cho các số thập phân. Currency hỗ trợ bốn chữ số thập phân và mười năm chữ số phần nguyên.

- **Kiểu String**

Mặc định biến hay tham số kiểu chuỗi có chiều dài thay đổi nó có thể tăng giảm tuỳ theo ta gán dữ liệu. Nhưng ta có thể khai báo chuỗi có chiều dài cố định.

- **Kiểu Boolean**

Là kiểu có hai giá trị Yes/ No, True/ False, Yes/ No, On/ Off.

- **Kiểu Date**

Khi các kiểu dữ liệu khác được chuyển sang Date, giá trị đứng trước là ngày, giá trị đứng sau dấu chấm là giờ. Nửa đêm là 0, nửa ngày là 0,5. Giựứ âm trước thể hiện ngày trước.

Ví dụ:

```
Click: cmd Exprole Date
Dim d Date As Date
If Is Date ( text Date text )
Then. Date = CV Date ( Text. date. Text )
Msg Box " Day = " & Day (d Date ) & - "Month= " Month  ( d Date)
    & - " Year = "Year ( d Date )
Else
  Msg Box " that isn't a valid date. Pleasetry again."
End if.
```

- **Kiểu Variant**

Có thể chứa nhiều loại dữ liệu chuỗi số thậm chí mảng. Ta không cần chuyển đổi dữ liệu, Visual Basic làm việc đó một cách tự động.

- **Kiểu mảng Array**

Mảng là một xâu các biến có cùng tên và cùng kiểu dữ liệu. Dùng Array làm chương trình đơn giản và rút gọn vì ta có thể dùng vòng lặp mảng có biến trên và biến dưới và các thành phần trong mảng là liên tục giữa hai biến .

Có 2 loại biến mảng: Mảng có chiều dài cố định và mảng động có chiều dài thay đổi lúc thi hành mảng có chiều dài cố định có thể được khai báo Public trong ứng dụng, Private trong Modul hoặc Private trong thủ tục.

4.2.2 Lệnh IF….THEN

CP1: Một dòng lệnh

If < Điều kiện > ***Then*** < Dòng lệnh >

CP2: Hai dòng lệnh:

If < Điều kiện > ***Then*** < khả thi lệnh 1 >

Else < Khả thi lệnh 2 >

End If

CP3: Nhiều dòng lệnh

If < Điều kiện 1 > Then < Khả thi lệnh 1>

Else If < điều kiện 2> Then < Khả thi lệnh 2 >

Else < khả thi lệnh 3 >

End if

Điều kiện là một so sánh hay một biểu thức mang giá trị số. VisualBasic thông dịch giá trị thành True / False (0 là False, giá trị khác 0 là True). Nếu điều kiện là True, Visual Basic thi hành tất cả dòng lệnh sau từ khoá Then.

Ví dụ:

If any Date < Now Then any Date = Now

If any Date < Now Then

Any Date = Now

End If

4.2.3. Lệnh SELECT CASE

Giải quyết trường hợp có quá nhiều Else If được dùng để chương trình dễ hiểu, dễ đọc. Biểu thức so sánh được tính toán một lần vào đầu cấu trúc. Sau đó, VisualBasic so sánh kết quả biểu thức với từng Case. Nếu bằng, nó thi hành khối lệnh trong Case đó.

Select Case < Tên biến >

Case < giá trị 1 hay phạm vi giá trị 1 >

< phát biểu 1 >

Case Else

< phát biểu n >

End Else

Mỗi danh sách biểu thức chứa một hoặc nhiều giá trị. Các giá trị cách nhau bằng dấu phẩy. Mỗi khối lệnh có thể chứa từ 0 đến nhiều dòng lệnh. Nếu có hơn một Case thoả điều kiện thì khối lệnh của Case đầu tiên được thi hành. Case Else không nhất thiết phải có, dùng trong trường hợp còn lại của các Case trước.

4.2.4. Lệnh FOR….NEXT (vòng lặp)

Biết trước số lần lặp. Ta dùng biến đếm tăng dần hoặc giảm dần trong vòng lặp.

For < tên biến > = < giá trị đầu > To < giá trị cuối >

(Các lệnh khả thi)

4.2.5. Lệnh DO … WHILE(vòng lặp)

Thi hành một khối lệnh với số lần không định trước, trong đó một biểu thức điều kiện dùng so sánh để quyết định vòng lặp có tiếp tục hay không. Điều kiện phải quy về False (0) hoặc true (khác 0).

Kiểu1: Lặp trong khi điều kiện True

Do While <điều kiện>

(các phát biểu khả thi) hay (khối lệnh)

Kiểu 2: Lặp trong khi điều kiện là False

Do until <điều kiện>

Các phát biểu khả thi

Loop

4.2.6. Phát biểu gán

<Tên biến > = Giá trị

<Tên đối tượng>.<Tính chất giá trị> = giá trị

<Tên đối tượng>.<Tính chất giá trị>= <Tên đối tượng 2>

<Tính chất giá trị>

4.3. Các thủ tục và hàm cơ bản của VB

- **Thủ tục**

Khi lập trình ta thường gặp những đoạn chương trình lặp đi lặp lại nhiều lần ở những chỗ khác nhau. Để chương trình đỡ phức tạp, các đoạn này được thay thế bằng những chương trình con tương ứng, khi cần ta chỉ cần gọi nó ra mà không cần viết lại cả đoạn.

Mặc khác, đối với những chương trình lớn và phức tạp. Việc xem xét tổng quan cũng như việc gỡ rối, hiệu chỉnh sẽ rất khó khăn. Việc lập các chương trình con sẽ chia chương trình ra từng khối, từng Modul. Điều này giúp cho chúng ta kiểm tra, gỡ rối và điều khiển dễ dàng.

Cấu trúc của một thủ tục có dạng như sau:

Sub < tên thủ tục >

' Các lệnh.. '

End sub

Nếu bạn muốn dùng thủ tục này trong toàn bộ chương trình thì bạn dùng thêm, từ khoá Public trước từ khoá Sub, còn nếu ta chỉ muốn dùng trong Form chứa nó thì bạn thêm từ khoá Private trước từ khoá Sub.

- **Hàm:**

Hàm cũng tương tự như thủ tục, chỉ khác là ta dùng hàm khi muốn nhận lại một kết quả trả lại của hàm, cấu trúc hàm như sau:

Function <tên hàm> (tham số As kiểu) **As** <Kiểu trả về>

' các lệnh...'

Tên hàm = giá trị trả về

End Function

4.3.1 HÀM ABS (Number)

Hàm này trả về một số giá trị tuyệt đối của Number.

4.3.2 HÀM SIN (Number as Double)

Tính sin của một góc.

4.3.3 HÀM COS (Number As Double)

Tính Cos của một góc.

4.3.4 HÀM TAN (Number As Double)

Tính hàm tang của một góc.

4. 3.5 HÀM ATN (Number As Double)

Tính hàm Arctang của một góc.

4. 3.6 HÀM INT (Number)

Trả về phần nguyên của số nếu là số dương. Còn nếu là số âm thì trả về phần nguyên có giá trị nhỏ hơn phần nguyên một đơn vị.

4. 3.7 HÀM FIX (Number)

Hàm trả về phần nguyên của số.

4.3.8 HÀM SQN (Number)

Hàm trả về một số nguyên

Nếu số > 0 sẽ trả về giá trị là 1

Nếu số < 0 sẽ trả về giá trị là - 1

Nếu số = 0 sẽ trả về giá trị là 0

4.3.9 HÀM SQR (Number)

Tính căn bậc hai của một số.

4.3.10 HÀM EXP (Number)

Tính e mũ của số

4.3.11 HÀM RND (Number)

Hàm này trả lại một số thực ngẫu nhiên.

4.3.12 HÀM DAY (Biến)

Trả về số ghi ngày của biến bạn nhập vào.

4.3.13HÀM MONTH (Biến)

Trả về số ghi tháng bạn nhập vào.

4.3.14 HÀM YEAR (Biến)

Trả về số ghi năm của biến bạn cần nhập vào.

4.3.15 HÀM NOW

Trả về ngày tháng năm và thời gian hiện tại.

4.3.16 HÀM WEEKDAY (biến)

Cho biết thứ mấy trong tuần.