Bài 1: Giới Thiệu

1) Lập trình là gì?

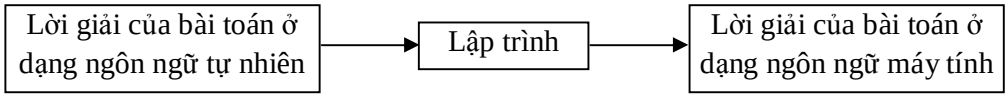
Lập trình là việc chuyển đổi lời giải bằng ngôn ngữ tự nhiên của 1 bài toán sang ngôn ngữ máy tính. Kết quả là ta sẽ có 1 chương trình chạy trên máy tính và giải quyết được bài toán đó.

2) Tại sao phải lập trình?

Lời giải của các bài toán thực tế được trình bày dưới dạng ngôn ngữ tự nhiên (ngôn ngữ giao tiếp trong đời sống hàng ngày như tiếng Việt, Anh, Pháp, …).

Máy tính cũng có ngôn ngữ riêng của nó. Nhưng ngôn ngữ trong máy tính thì rất đơn sơ. Ví dụ: cộng 2 số, so sánh 2 số, …

Do đó ta cần chuyển đổi lời giải của bài toán ở dạng ngôn ngữ tự nhiên sang ngôn ngữ máy tính. Quá trình chuyển đổi đó gọi là lập trình (hay mã hóa).



3) Các bước cần thiết khi lập trình

Để giải quyết 1 bài toán trên máy tính (lập trình), ta cần thực hiện những bước sau:

Bước 1: Xác định và hiểu rõ bài toán.

Bước 2: Tìm lời giải cho bài toán ở dạng ngôn ngữ tự nhiên.

Bước 3: Lập trình (**programming**).

Ví dụ: Lập trình giải bài toán phương trình bậc 2 như sau: $Ax^2 + Bx + C = 0$.

Bước 1: bài toán ở đây là phương trình bậc 2 có dạng $Ax^2 + Bx + C = 0$. Việc hiểu rõ bài toán ở đây là A, B, và C các hằng số và $A \neq 0$; x là ẩn số của phương trình.

Bước 2: như ta đã biết lời giải của phương trình bậc 2 được biểu diễn dưới dạng ngôn ngữ tự nhiên thông qua các bước như sau:

Tính $\Delta = B^2 - 4AC$

Nếu $\Delta \geq 0$ thì phương trình có 2 nghiệm là $x_{1,2} = \dfrac{-B \pm \sqrt{\Delta}}{2A}$

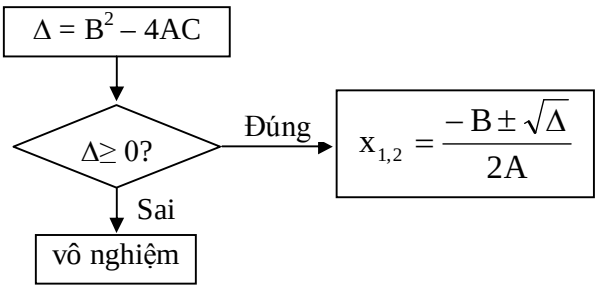
Nếu $\Delta < 0$ thì phương trình vô nghiệm

Bước 3: chọn ngôn ngữ để lập trình

4) Một số cách thức biểu diễn lời giải của bài toán

Lời giải của 1 bài toán thường được gọi là thuật giải (thuật toán, **algorithm**). Thông thường thuật giải của 1 bài toán thường được biểu diễn ở dạng lưu đồ (**scheme**) hay mã giả (**pseudo-code**).

Hình dưới là cách thức biểu diễn thuật toán giải phương trình bậc 2 ở dạng lưu đồ



Đoạn mã (**code**) dưới đây là cách thức biểu diễn thuật toán giải phương trình bậc 2 bằng mã giả

$\Delta = B^2 - 4AC$

If $\Delta \geq 0$ Then $x_{1,2} = \frac{-B \pm \sqrt{\Delta}}{2A}$

If $\Delta < 0$ Then phương trình vô nghiệm

5) Ngôn ngữ lập trình (Programming Languages)

Ngôn ngữ được dùng để cung cấp câu lệnh cho máy tính. Có rất nhiều loại ngôn ngữ, dưới đây liệt kê 1 số loại:

+ Ngôn ngữ cấp thấp: Assembly.

+ Ngôn ngữ cho mục đích tổng quát: PL/I, C, C++, Pascal, Modula-2, Ada, Java, C#, Visual Basic.

+ Ngôn ngữ cho toán, khoa học, kỹ thuật: Fortran, Maple, Matlab, Mathematica.

+ Ngôn ngữ cho tiến trình kinh doanh: Cobol, RPG.

+ Ngôn ngữ cho trí tuệ nhân tạo: Prolog, Lisp.

+ Ngôn ngữ xử lý chuỗi, ngôn ngữ script: Perl, Python, VBScript, JavaScript.

6) Visual Basic for Applications (VBA)

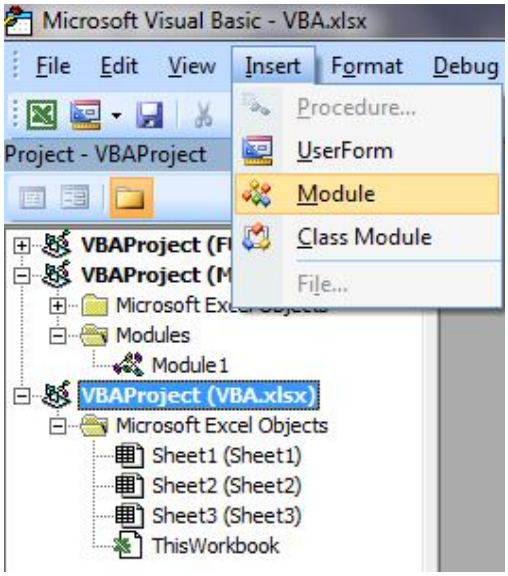
+ VBA là phần mở rộng của ngôn ngữ Visual Basic được dùng để nâng cao các tính năng trong Word, Excel, Access.

+ **Để mở cửa sổ VBA** trong Excel, nhấn phím Alt + F11.

+ **Thêm 1 module** vào trong Workbook.

Bước 1: Nếu chưa ở trong cửa sổ VBA thì nhấn Alt + F11

Bước 2: Click vào tên workbook. Click menu **Insert** rồi chọn **Module** (Module là 1 đơn vị chứa mã lệnh).

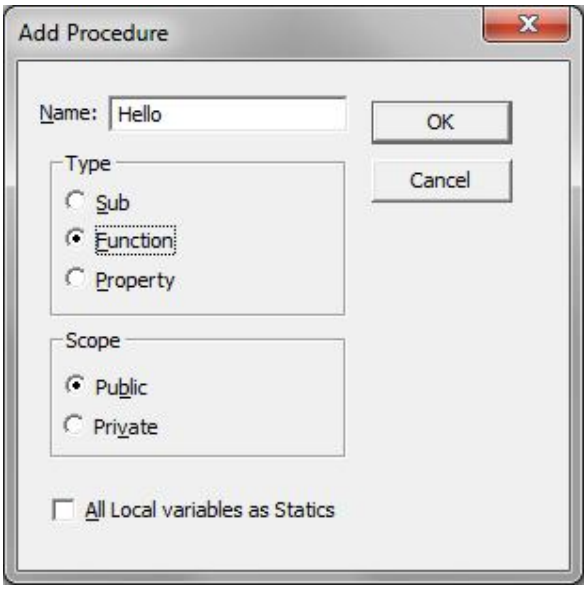


+ **Thêm 1 hàm** vào trong 1 module

Bước 1: Mở module

Bước 2: Click menu **Insert** rồi chọn **Procedure**

Bước 3: Đặt tên hàm tại ô **Name**; rồi chọn **Function**.



Khi đó trên cửa sổ VBA sẽ xuất hiện những dòng sau:

Public Function Hello()

End Function

Hàm là 1 tập hợp các câu lệnh để thực hiện 1 chức năng nào đó.

+ **Ví dụ**: Viết 1 hàm đơn giản trên VBA.

Bước 1: Mở cửa sổ VBA.

Bước 2: Thêm 1 hàm có tên là **Hello**.

Bước 3: Gõ vào dòng lệnh sau:

MsgBox ("Chao ban, day là chuong trinh VBA dau tien cua toi!!!")

Khi đó trên cửa sổ VBA, bạn sẽ thấy kết quả như sau:

```
Public Function Hello()
  MsgBox ("Chao ban, day là chuong trinh VBA dau tien cua toi!!!")
End Function
```

Bước 4: Kiểm tra (**test**) chức năng của hàm **Hello**.

Quay lại Excel, tại 1 ô bất kỳ, gõ vào công thức sau: **=Hello()**

Chú ý: để lưu workbook có chứa các hàm viết bằng VBA, ta phải dùng loại file **Excel Macro-Enabled Workbook** (*.**xlsm**)

Bài 2: Biến – Kiểu dữ liệu

Biến (**variable**) là 1 vùng bộ nhớ dùng để lưu trữ 1 giá trị có tính tạm thời. Trước khi sử dụng 1 biến, ta phải khai báo biến.

1) Cách khai báo biến

Dim <tên biến> **As** <kiểu> [,<tên biến> **As** <kiểu>]…

- tên biến : phải bắt đầu bằng 1 ký tự, dài tối đa 255 ký tự, không được có khoảng trắng và dấu chấm.

- kiểu (**data type**) : có những kiểu sau:

Loại	Tên kiểu	Số byte	Miền giá trị
Số nguyên	**Byte**	1	0 to 255
	Integer	2	[–32,768, 32,767]
	Long	4	[–2,147,483,648, 2,147,483,647]
Số thực	**Single**	4	Số âm: –3.402823**E**38 to –1.401298**E**–45 Số dương: 1.401298**E**–45 to 3.402823**E**38
	Double	8	Số âm: –1.79769313486232**E**308 .. –4.94065645841247**E**–324 Số dương: 4.94065645841247**E**–324 .. 1.79769313486232**E**308
Logic	**Boolean**	2	True, False
Ngày, giờ	**Date**	8	01/01/100 to 31/12/9999
Chuỗi	**String**		Từ 0 đến khoảng 2 tỷ ký tự
1 ký tự	**Char**	2	

Ví dụ 1:

Dim diem **As Single**, phai **As Boolean**, ngay **as Date**

Dim ten As String

Chú ý: Bật chế độ bắt buộc phải khai báo biến trước khi dùng: tại cửa sổ VBA, click menu **Tools** ⇒ **Options**, rồi chọn **Require Variable Declaration**.

2) Các phép toán

- Cộng, trừ, nhân, chia : **+, –, *, /**
- Lũy thừa : ^ VD: 6^2 → 36
- Trả về số dư của phép chia: **mod** VD: 14 **mod** 3 → 2
- Trả về thương số của phép chia: \ VD: 14 \ 3 → 4
- Nối nhiều chuỗi: **&** VD: "abc" **&** "456" **&** "xyz" → "abc456xyz"

3) Phép gán

Dùng để thay đổi giá trị của 1 biến. Cú pháp như sau: **<Tên biến> = <giá trị>**

Ví dụ 2:

```
ten = "Visual Basic"
phai = True
ngay = #10/18/2011#
diem = 6.5
diem = diem + 0.5
x = y^2 + (z mod y)
```

Ví dụ 3: viết hàm trả về thương số của 1 phép chia

```
Public Function ThuongSo(SoBiChia As Integer, SoChia As Integer)
   ThuongSo = SoBiChia \ SoChia
End Function
```

Chú ý:

- Khai báo tham số (**argument** / **parameter**): <tên tham số> **As** <kiểu>
- Tham số được dùng như biến, nghĩa là có thể áp dụng các phép toán, phép gán, …
- Hàm luôn trả về 1 giá trị, cú pháp: <tên hàm> = <giá trị trả về>

Bài tập

1) Viết hàm trả về số dư của 1 phép chia.

2) Viết hàm nhận vào 1 số nguyên có 3 chữ số, tính tổng và tích của 3 chữ số ấy.

3) Viết hàm nhận vào 1 số tiền nguyên, đổi số tiền ấy thành tờ 500, 20, và dư bao nhiêu.

4) Viết hàm nhận vào 1 thời gian tính theo giây, đổi sang dạng giờ:phút:giây.

(Gợi ý: các bài 2, 3, và 4 ta trả về 1 chuỗi)

Bài 3: Lệnh điều kiện (Conditional Statements)

Lệnh điều kiện cho phép CPU thực thi 1 tập hợp lệnh trong những điều kiện được định trước.

1) Điều kiện (Condition / Criterion)

- Điều kiện là những biểu thức so sánh và luôn trả về True hoặc False.

1.1) Đơn

<biểu thức> <phép toán so sánh> <biểu thức>

- phép toán so sánh: >, >=, <, <=, =, <>

Ví dụ 1: a > 10

(n mod 2) = 0

ngay <= #28/04/99#

ten = "VBA"

x+y <> z^2

1.2) Phức

- Tổ hợp của các điều kiện đơn và các phép toán logic (**And**, **Or**).

Ví dụ 2: (a > b+5) And (x = #5/20/88#)

(t = 1) Or (t = 3) Or (t = 5) Or (t = 7) Or (t = 8)

(phai = True) Or ((phai = False) And (ngay >= #06/14/89#))

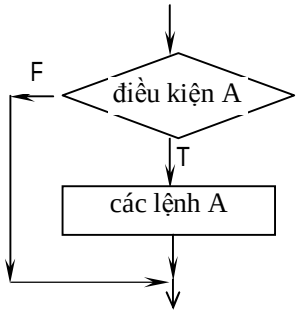
2) Lệnh If

2.1) Dạng 1

If < điều kiện A> **Then**

< các lệnh A>

End If



Ý nghĩa: thực thi **các lệnh A** khi **điều kiện A** đúng (True)

2.2) Dạng 2

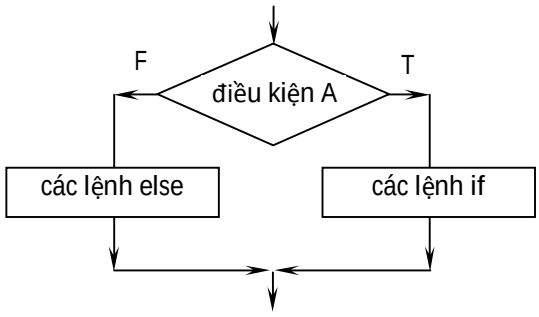
If < điều kiện A > **Then**

< các lệnh if >

Else

< các lệnh else >

End If



Ý nghĩa: thực thi **các lệnh if** khi **điều kiện A** đúng (True); ngược lại thực thi **các lệnh else**.

2.3) Dạng 3

If < đk1 > **Then**
    < các lệnh 1 >

Elseif < đk2 > **Then**
    < các lệnh 2 >

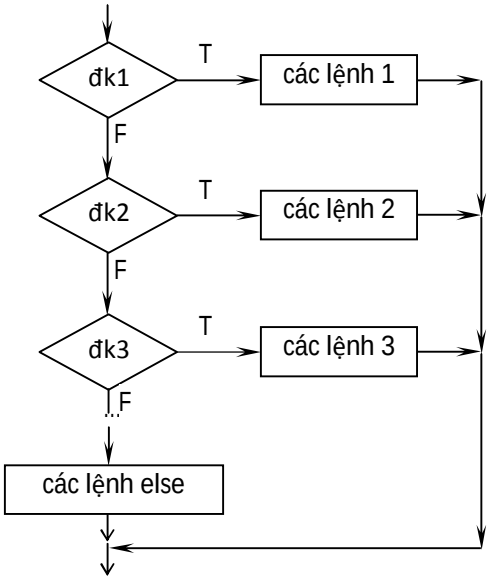
Elseif < đk3 > **Then**

    < các lệnh 3 >

    …

Else
    < các lệnh else>

End If



Ý nghĩa: thực thi **các lệnh 1** khi **đk1** đúng (True); ngược lại thực thi **các lệnh 2** khi **đk2** đúng; …; nếu tất cả các điều kiện đều sai (False) thì **các lệnh else** được thực thi.

Ví dụ 3: viết hàm nhận vào 1 số nguyên, cho biết số đó chẵn hay lẻ.

```
Public Function ChanLe(So As Integer)
    If (So mod 2) = 0 then
        ChanLe = "Số chẵn"
    Else
        ChanLe = "Số lẻ"
    End If
End Function
```

Ví dụ 4: viết hàm nhận vào 1 số, cho biết số đó âm, dương, hay bằng 0.

```
Public Function Dau(So As Double)
    If (So > 0) then
        Dau = "Số dương"
    Elseif (So < 0) then
        Dau = "Số âm"
    Else
        Dau = "Số 0"
    End If
End Function
```

3) Lệnh Select Case

Select Case <biểu thức B>

Case n_1

< các lệnh n_1 >

Case n_2

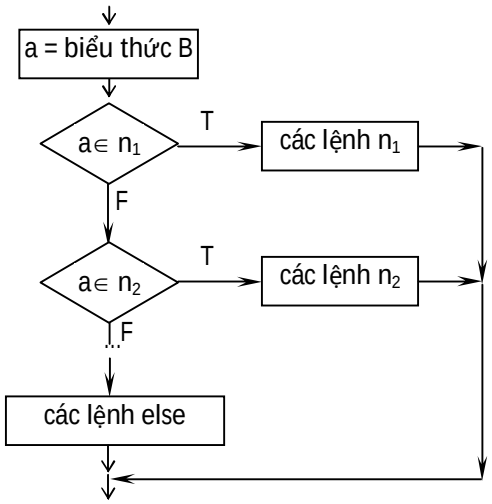
< các lệnh n_2 >

…

Case Else

< các lệnh else >

End Select



- n_1, n_2, … : là những tập hợp giá trị, có thể được viết dưới những dạng sau:

+ các giá trị rời rạc:

Cú pháp: **giá trị 1, giá trị 2, …**

Ví dụ 5: 1, 4, 7

"ab", "cde"

#03/15/89#, #04/16/99#

+ các giá trị liên tục:

Cú pháp: **giá trị 1 To giá trị 2**

Ví dụ 6: 1 to 10 [1, 10]

"A" to "Z"

Is > 8 (8, +∞)

Is <= 2 (-∞, 2]

Ý nghĩa: giả sử a là giá trị của **biểu thức B**. Nếu a thuộc tập **n_1** thì thực thi các lệnh **n_1**. Nếu a thuộc tập **n_2** thì thực thi **các lệnh n_2**. …Nếu a không thuộc tập hợp các giá trị nào thì **các lệnh else** được thực thi.

Ví dụ 7: viết hàm nhận vào tháng và năm, cho biết tháng đó có bao nhiêu ngày?

```
Public Function SoNgay(Thang As Byte, Nam As Integer)
    Dim sn As Byte, tmp As Integer
    Select Case Thang
        Case 1, 3, 5, 7, 8, 10, 12: sn = 31
        Case 4, 6, 9, 11: sn = 30
        Case 2
            tmp = Nam Mod 100
            If tmp = 0 Then tmp = Nam \ 100
            If (tmp Mod 4) = 0 Then sn = 29 Else sn = 28
    End Select
    SoNgay = sn
End Function
```

Bài tập

1) Viết lại ví dụ 3 và 4 bằng cách dùng SELECT.

2) Viết lại ví dụ 7 bằng cách dùng IF.

3) Viết hàm nhận vào 3 số. Cho biết số lớn nhất.

4) Viết hàm nhận vào 1 số nguyên có 3 chữ số, cho biết chữ số nhỏ nhất.

5) Viết hàm nhận vào 1 ngày. Cho biết thứ trong tuần.

6) Viết hàm nhận vào 1 năm. Cho biết năm âm lịch.

(*Hướng dẫn: năm âm lịch gồm can và chi.*

Can: Giáp-4, Ất-5, Bính-6, Đinh-7, Mậu-8, Kỷ-9, Canh-0, Tân-1, Nhâm-2, Quí-3.

Chi: Tí-4, Sửu-5, Dần-6, Mão-7, Thìn-8, Tỵ-9, Ngọ-10, Mùi-11, Thân-0, Dậu-1, Tuất-2, Hợi-3.)

7) Viết hàm nhận vào 2 số A và B. Giải phương trình $Ax + B = 0$.

(*Hướng dẫn:*

nếu (A = 0) và (B = 0) thì phương trình có vô số nghiệm.

Nếu (A = 0) và (B ≠ 0) thì phương trình vô nghiệm.

Nếu (A ≠ 0) thì phương trình có nghiệm x = -B/A.)

8) Viết hàm nhận vào 3 số A, B và C. Giải phương trình $Ax^2 + Bx + C = 0$.

(*Hướng dẫn:*

nếu (A = 0) thì giải phương trình $Bx + C = 0$.

Nếu (A ≠ 0)

+ Tính $\Delta = B^2 - 4AC$

+ Nếu $\Delta < 0$ thì vô nghiệm

+ Nếu $\Delta \geq 0$ thì có 2 nghiệm xác định bởi công thức $\frac{-B \pm \sqrt{\Delta}}{2A}$)

Bài 4: Lệnh vòng lặp (Loop Statements)

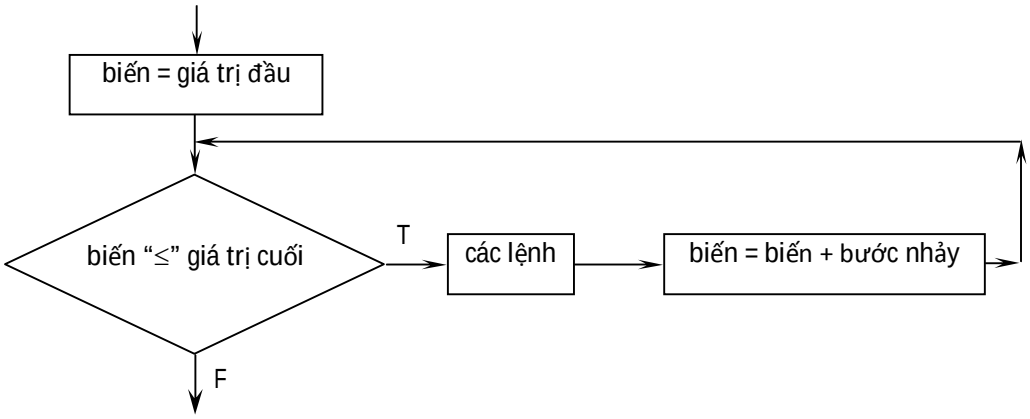
Lệnh vòng lặp cho phép thực thi các lệnh nhiều lần.

1) FOR

For <biến> = <giá trị đầu> **To** <giá trị cuối> **Step** <bước nhảy>

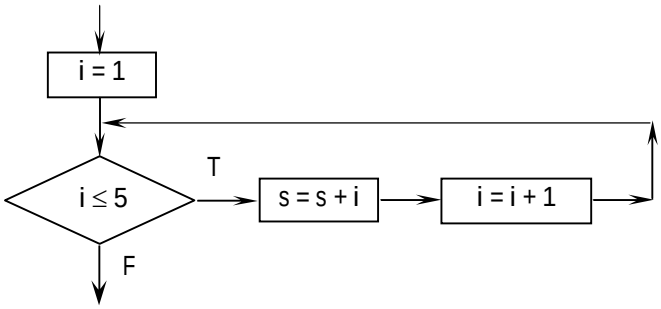
< các lệnh>

Next <biến>

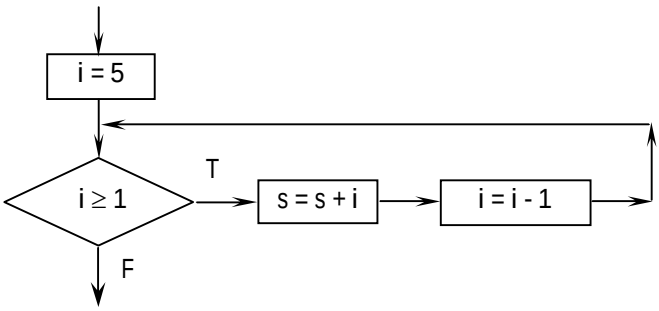


Ví dụ 1:

```
For i = 1 to 5 step 1
    s = s + i
next i
```




```
For i = 5 to 1 step –1
    s = s + i
next i
```

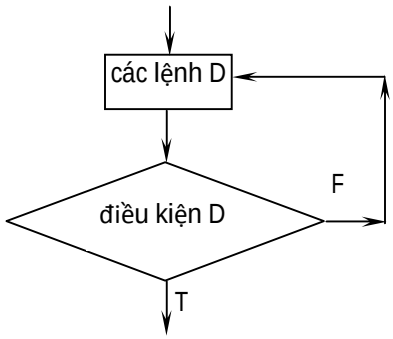


2) DO

Do

<các lệnh D>

Loop Until <điều kiện D>



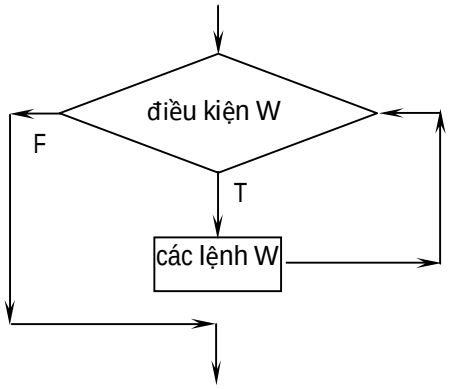
Ý nghĩa: thực thi **các lệnh D** cho đến khi **điều kiện D** đúng (True).

3) WHILE

While <điều kiện W>

<các lệnh W>

Wend



Ý nghĩa: khi **điều kiện W** còn đúng (True) thì thực thi **các lệnh W**.

Ví dụ 2: viết hàm nhận 1 số nguyên dương N, tính tổng và tích các số từ 1 đến N.

Tổng = 1 + 2 + 3 + … + N

Tích = 1x2x3x…xN = N!

```
Public Function TongTich(N As Integer)
    Dim Tong As Integer, Tich As Double
    Tong = 0
    Tich = 1
    For i = 1 to N
        Tong = Tong + i
        Tich = Tich*i
    Next i
    TongTich = "Tong = " & Tong & "; Tich = " & Tich
End Function
```

Ví dụ 3: viết hàm nhận vào 1 số nguyên dương N, tính tổng các chữ số trong N.

```
Public Function TongChuSo(N As Integer)
    Dim Tong As Integer, cs as Byte
    Tong = 0
    While N > 0
        cs = N mod 10
        N = N \ 10
        Tong = Tong + cs
    Wend
    TongChuSo = Tong
End Function
```

Bài tập

1) Viết lại ví dụ 2 bằng cách sử dụng vòng lặp **Do**.

2) Viết hàm nhận vào 1 số nguyên dương N, tìm chữ số nhỏ nhất trong N.

3) Viết hàm nhận vào 1 số nguyên dương N, tính tổng các số lẻ nhỏ hơn hoặc bằng N.

4) Viết hàm nhận vào 1 số nguyên dương N, nếu N lẻ thì tính tích các số lẻ ≤ n, nếu N chẵn thì tính tích các số chẵn ≤ n.

5) Viết hàm nhận vào 1 số nguyên dương N. Tính S = 1 + 1/2! + 1/3! + … + 1/n!

6) Viết hàm nhận vào 1 số nguyên dương n, và số thực x. Tính $S = x - x^2/2! + x^3/3! - x^4/4! + \ldots + (-1)^{n+1}x^n/n!$

7) $\begin{cases} U_0 = 0 \\ U_1 = 1 \\ U_i = U_{i-1} + U_{i-2} \end{cases}$ nhập số nguyên n, tính U_n?

8) Viết hàm nhận 1 số hệ 2 ở dạng chuỗi, đổi số đó sang hệ 10.

9) Viết hàm nhận 1 số hệ 16 ở dạng chuỗi, đổi số đó sang hệ 10.

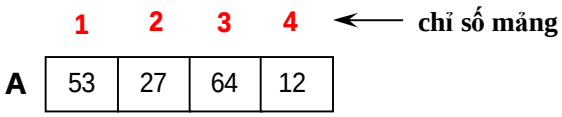
Bài 5: Mảng (Array)

- Mảng là 1 dãy các biến có cùng kiểu dữ liệu. Các biến trong mảng có cùng 1 tên gọi là tên mảng. Các biến trong mảng được phân biệt với nhau thông qua 1 chỉ số (số nguyên). Mỗi biến trong mảng được gọi là phần tử.

1) Khai báo và sử dụng

- Khai báo 1 mảng: **Tên mảng(n_1 to n_2) As kiểu**

Ví dụ 1:
```
Dim A(1 To 4) As Byte
Dim N(0 To 6) As Integer
Dim X(–2 To 4) As Double
```



- Các số 53, 27, 64, 12 là những giá trị của các phần tử có chỉ số là 1, 2, 3 và 4.
- Để lấy giá trị 1 phần tử, ta dùng cú pháp: **Tên mảng(chỉ số)**
- Để gán giá trị cho 1 phần tử, ta dùng cú pháp: **Tên mảng(chỉ số) = giá trị**

Ví dụ 2: A(2) = A(1) + A(4)*2

2) Mảng 2 chiều (bảng / ma trận)

- Mảng 2 chiều được cấu thành từ nhiều dòng và nhiều cột.
- Khai báo: **Tên mảng(d_1 to d_2, c_1 to c_2) As kiểu**
- Sử dụng: **Tên mảng(chỉ số dòng, chỉ số cột)**

Ví dụ 3:
```
Dim M(1 To 3, 1 To 4) As Double
M(1, 3) = –2.5
```

3) Mảng có số phần tử thay đổi (mảng động/dynamic array)

- Khai báo: **Tên mảng() As kiểu**
- Xác định số phần tử của mảng: **Redim Tên mảng(n_1 to n_2)**

  Redim Tên mảng(d_1 to d_2, c_1 to c_2)

Ví dụ 4:
```
Dim A() As Byte
Dim M() As Double
ReDim A(1 To 5)
ReDim M(1 To 2, 1 To 3)
```

Chú: từ khóa **Preserve** dùng để giữ lại các giá trị của mảng 1 chiều khi **Redim**.

- Hàm **LBound** và **UBound**

Ví dụ 5:

```
Dim M() As Byte, d As Integer, c As Integer
```

```
ReDim M(1 To 3, –2 To 5)
c = LBound(M)        // c = 1
d = UBound(M)        // d = 3
c = LBound(M, 2)     // c = –2
d = UBound(M, 2)     // d = 5
```

4) Một số bài toán trên mảng

a) Tính tổng các phần tử của mảng

```
Public Function Tong(Ran As Range)
Dim d As Integer, c As Integer, sum As Double
sum = 0
For d = 1 To Ran.Rows.Count
   For c = 1 To Ran.Columns.Count
      sum = sum + Ran.Cells(d, c)
   Next c
Next d
Tong = sum
End Function
```

b) Tìm phần tử trong mảng

```
Public Function Tim(Val As Double, Ran As Range)
Dim num As Integer, oCell As Range
num = 0
For Each oCell In Ran
   If oCell.Value = Val Then
      num = num + 1
      oCell.Font.Color = vbRed
   End If
Next
Tim = num
End Function
```

c) Tìm phần tử lớn nhất trong mảng

```
Public Function LonNhat(Ran As Range)
Dim max As Double, v As Integer, d As Integer, c As Integer
max = Ran.Cells(1, 1)
For d = 1 To Ran.Rows.Count
   For c = 1 To Ran.Columns.Count
      If Ran.Cells(d, c) > max Then max = Ran.Cells(d, c)
   Next c
Next d
v = Tim(max, Ran)
LonNhat = max
End Function
```

Bài tập

1) Viết hàm nhận 1 số hệ 10, đổi số đó sang hệ 2.

2) Viết hàm nhận 1 số hệ 10, đổi số đó sang hệ 16.

3) Tính dãy Fibonacci bằng cách dùng mảng.

$$\begin{cases} U_0 = 0 \\ U_1 = 1 \\ U_i = U_{i-1} + U_{i-2} \end{cases}$$

Bài 6: Ứng dụng VBA trong Excel

Ví dụ 1: viết hàm nhận vào 2 mảng x_i và W_i. Trả về $\sum_{i=1}^{n} x_i W_i$.

```
Public Function EV(X As Range, W As Range)
Dim i As Integer, sum As Double

sum = 0
For i = 1 To X.Columns.Count
   sum = sum + X.Cells(1, i) * W.Cells(1, i)
Next i

EV = sum
End Function
```

Ví dụ 2: Đổi chữ thành chữ hoa của vùng chọn.

```
Public Sub Hoa()
Dim SelRan As Range, d As Long, c As Long, s As String

Set SelRan = Selection

For d = 1 To SelRan.Rows.Count
   For c = 1 To SelRan.Columns.Count
      s = SelRan.Cells(d, c)
      SelRan.Cells(d, c) = UCase(s)
   Next c
Next d
End Sub
```

Cách đưa thủ tục Hoa thành 1 nút trên thanh Menu:

Bước 1: **Right click** trên thanh **menu** chọn **Customize Quick Access Toolbar…**

Bước 2: Tại ô **Choose commands from**, chọn Macros. Click mục **Hoa** trong ô phía dưới. Rồi click nút **Add**. Click **Ok**.

Bước 3: Trên thanh menu sẽ xuất hiện 1 nút **Hoa**. Khi click vào nút này thì thủ tục **Hoa** sẽ thực thi.

Chú ý:

- Thủ tục (**Sub**) không trả về giá trị và không gọi thực hiện trong công thức.

Ví dụ 3: Dùng hàm có sẵn trong Excel

```
Public Function Ex(Ran As Range)
Ex = Application.WorksheetFunction.Sum(Ran)
End Function
```

Ví dụ 4: Thay đổi kiểu chữ (**Font**)

```
Public Sub KieuChu(Ran As Range)
Dim SelRan As Range

Set SelRan = Selection
With SelRan
   .Font.Bold = True
   .Font.Color = RGB(125, 236, 123)
   .Font.Name = "Times New Roman"
   .Font.Size = 12
End With
End Sub
```

Ví dụ 5: Thay đổi đường viền (**Borders**)

```
Public Sub KeKhung(Ran As Range)
Dim SelRan As Range

Set SelRan = Selection
With SelRan
   .Borders.Item(xlEdgeBottom).LineStyle = 1
   .Borders.Item(xlEdgeTop).LineStyle = 1
   .Borders.Item(xlEdgeLeft).LineStyle = 1
   .Borders.Item(xlEdgeRight).LineStyle = 1
   .Borders.Item(xlInsideHorizontal).LineStyle = 1
   .Borders.Item(xlInsideVertical).LineStyle = 1
   .Borders.Item(xlDiagonalDown).LineStyle = 1
   .Borders.Item(xlDiagonalUp).LineStyle = 1
End With
End Sub
```

Bài 7: Thủ tục – Hàm

Thủ tục / hàm là 1 đoạn lệnh để thực hiện 1 chức năng nào đó. Sau khi thực hiện xong đoạn lệnh đó thì hàm luôn trả về 1 giá trị còn thủ tục thì không.

1) Cấu trúc của 1 thủ tục/hàm

Public Sub < **tên thủ tục** >**([phần định nghĩa tham số])**

< Các lệnh của thủ tục >

End Sub

Public Function **< tên hàm >([phần định nghĩa tham số])** **As** **< kiểu giá trị trả về >**

< Các lệnh của hàm >

< tên hàm > = < giá trị trả về >

End Function

2) Cách gọi thủ tục / hàm thực hiện

Call < tên thủ tục >[(< danh sách tham số truyền >)]

< tên biến > = < tên hàm >[(< danh sách tham số truyền >)]

3) Cách định nghĩa tham số trong thủ tục / hàm

ByVal < tên tham số > **As** < kiểu >, …

hoặc **ByRef** < tên tham số > **As** < kiểu >, …

- **ByVal** : tham số có kiểu giá trị (tham trị), có 2 đặc điểm sau:

+ Khi truyền cho phép truyền giá trị hoặc biến.

+ Khi tham số bị thay đổi giá trị bởi những câu lệnh bên trong thủ tục định nghĩa nó, thì biến truyền tương ứng sẽ không bị thay đổi.

- **ByRef** : tham số có kiểu biến (tham biến), có 2 đặc điểm sau:

+ Khi truyền chỉ nên truyền biến.

+ Khi tham số bị thay đổi giá trị bởi những câu lệnh bên trong thủ tục định nghĩa nó thì biến truyền tương ứng sẽ bị thay đổi theo.

Ví dụ 1:

```
Public Function ThamTri()
Dim y As Integer, s As String

y = 10
s = "y truoc khi goi Test la " & y
Call Test(y)
s = s & "; y sau khi goi Test la " & y
ThamTri = s
End Function

Public Sub Test(ByVal x As Integer)
x = x + 5
End Sub
```

Ví dụ 2:

```
Public Function ThamBien()
Dim y As Integer, s As String

y = 10
s = "y truoc khi goi Test la " & y
Call Test(y)
s = s & "; y sau khi goi Test la " & y

ThamBien = s
End Function

Public Sub Test(ByRef x As Integer)
x = x + 5
End Sub
```

Chú ý: mảng luôn được truyền ở dạng tham biến.

Ví dụ 3:

```
Public Function ThamBien()
Dim y(1 To 5) As Integer, i As Integer

For i = 1 To 5
   y(i) = i
Next i
Call Test(y)
End Function

Public Sub Test(x() As Integer)
Dim i As Integer

For i = 1 To 5
   x(i) = x(i) + 1
Next i
End Sub
```

4) Tầm ảnh hưởng của các biến

- Biến cục bộ / tham số được khai báo trong thủ tục, hàm.

+ Biến cục bộ chỉ được sử dụng đối với các câu lệnh trong thủ tục định nghĩa ra nó.
+ Biến cục bộ chỉ tồn tại trong bộ nhớ khi thủ tục được thực hiện.

Ví dụ 4:

```
Public Function CucBo()
Dim x As Integer

x = x + 5

CucBo = x
End Function
```

- Biến toàn cục Module

a) Cách khai báo:

- Tại phần trên cùng của cửa sổ **Module**, điền vào các câu lệnh khai báo biến.

b) Đặc điểm:

+ Biến toàn cục Module được sử dụng đối với tất cả các câu lệnh trong Module.
+ Biến toàn cục Module tồn tại trong bộ nhớ khi workbook được mở.

Ví dụ 5:

```
Option Explicit
Dim TC As Integer          ' TC là biến toàn cục Module
```

```
Public Function Test1()
TC = TC + 1
Test1 = TC
End Function
```

```
Public Function Test2()
TC = TC + 2
Test2 = TC
End Function
```

5) Sự trùng tên

Khi có sự trùng tên giữa biến cục bộ và biến toàn cục Module. Thì biến đó sẽ được hiểu là:

+ Biến cục bộ: khi thực hiện các câu lệnh của thủ tục định nghĩa ra biến trùng tên đó.
+ Biến toàn cục Module: khi thực hiện các câu lệnh ở phần còn lại.

Ví dụ 6:

```
Option Explicit
Dim TC As Integer          ‘ TC là biến toàn cục Module
```

```
Public Function Test1()
TC = TC + 1
Test1 = TC
End Function
```

```
Public Function Test2()
Dim TC as Byte
TC = TC + 2
Test2 = TC
End Function
```

Bài 8: Hộp đối thoại

1) Phương pháp lập trình

Sau khi có lời giải của bài toán ở dạng ngôn ngữ tự nhiên, ta sẽ tiến hành tổ chức chương trình sao cho dễ phát triển, kiểm tra, nâng cấp, và làm theo nhóm. Dưới đây ta trình bày sơ lược 3 phương pháp.

a) Lập trình theo hướng thủ tục/hàm (Procedural Programming)

+ Chương trình được chia thành nhiều thủ tục và hàm.

+ Mỗi thủ tục/hàm thực hiện 1 chức năng nào đó.

+ Mỗi thủ tục/hàm bao gồm 1 số câu lệnh nào đó để thực thi chức năng của nó.

+ Khi tên của thủ tục/hàm được gọi thì chức năng của thủ tục/hàm đó được thực hiện (các câu lệnh của thủ tục/hàm được thực thi).

+ Mỗi thủ tục/hàm nên có tham số để chúng trở nên linh động và tổng quát.

b) Lập trình theo hướng đối tượng (Object Oriented Programming–OOP)

+ Chương trình được chia thành nhiều đối tượng.

+ Đối tượng là 1 thực thể có thật trong cuộc sống như: sinh viên, toán, cái bàn, đường tròn, …

+ Một đối tượng được cấu tạo bởi thuộc tính (**property**/**attribute**) và phương thức (**method**).

+ Thuộc tính là những tính chất mô tả đối tượng.

+ Phương thức (**cũng là thủ tục/hàm**) là những chức năng của đối tượng.

+ Ví dụ: Xét đối tượng hình tròn thì

Thuộc tính: bán kính, màu đường viền, màu tô, …

Phương thức: lấy bán kính hiện tại, thay đổi bán kính, tô màu nền, tô màu đường viền, …

+ Khi muốn sử dụng thuộc tính/phương thức của 1 đối tượng, ta dùng cú pháp: **đối tượng.thuộc tính**, **đối tượng.phương thức**

c) Lập trình theo hướng sự kiện (Event Driven Programming)

+ Đây là sự mở rộng của OOP nhưng thêm khái niệm sự kiện/biến cố (**Event**).

+ Sự kiện (**cũng là thủ tục/hàm**) là những tác động từ môi trường bên ngoài lên đối tượng mà đối tượng có thể hiểu.

+ Ví dụ: xét đối tượng nút lệnh (command button) thì các sự kiện là: click, right click, mouse move

+ Khi lập trình theo hướng này, thì ta chủ yếu viết các câu lệnh cho sự kiện cần thiết. Nghĩa là dạy đối tượng cách ứng xử với tác động từ bên ngoài.

+ Các câu lệnh của 1 sự kiện chỉ được thi hành khi sự kiện đó được kích hoạt.

2) Form

+ Form là 1 cửa sổ. Để thêm 1 Form, click menu Insert ⇨ UserForm.

+ Một vài thuộc tính quan trọng của Form:

Name: tên của Form (dùng để viết code)

Caption: chuỗi ký tự xuất hiện trên thanh tiêu đề của Form

+ Một vài sự kiện quan trọng của Form:

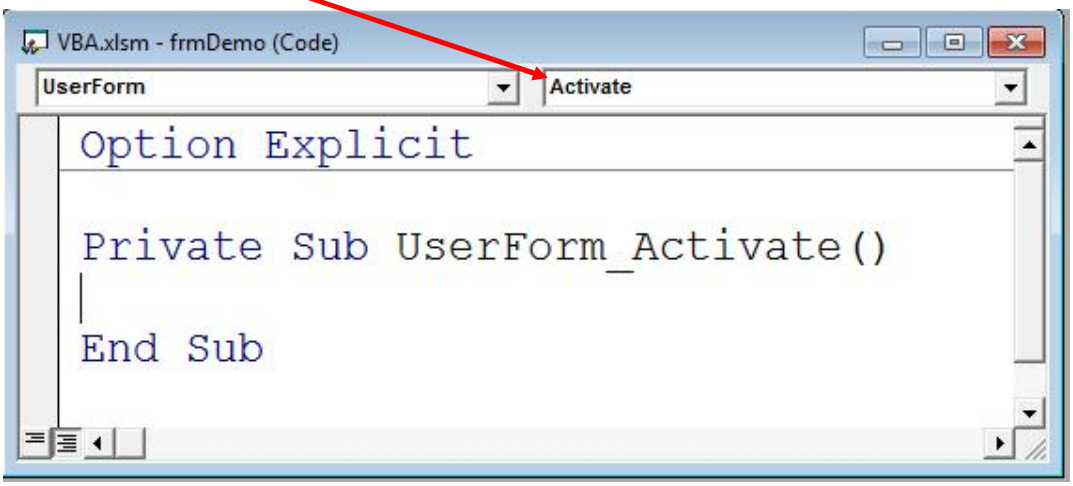
Activate: khi Form được kích hoạt

Terminate: khi đóng Form.

Ví dụ 1: Tạo 1 Form với các sự kiện sau:
Activate: hiển thị Hello
Terminate: hiển thị Bye Bye
MouseMove: thay đổi màu nền của Form.

Bước 1: Insert ⇨ UserForm. Sau đó đặt thuộc tinh **Name** là **frmDemo**.

Bước 2: Double click trên Form để mở cửa sổ viết sự kiện cho Form. Rồi sau đó chọn sự kiện cần viết code tại ô này



Bước 3: viết code cho 3 sự kiện như sau

```
Private Sub UserForm_Activate()
MsgBox ("Hello")
End Sub

Private Sub UserForm_MouseMove(…)
Me.BackColor = RGB(Rnd() * 255, Rnd() * 255, Rnd() * 255)
End Sub

Private Sub UserForm_Terminate()
MsgBox ("Bye Bye")
End Sub
```

Bước 4: viết hàm sau trong 1 Module.

```
Public Function Test()
frmDemo.Show
End Function
```

3) Command Button: nút lệnh.

+ Một vài thuộc tính quan trọng:

Name: tên

Caption: chuỗi ký tự xuất hiện trên nút lệnh

Enabled = True / False ⇔ các sự kiện được / không được kích hoạt

Visible = True / False ⇔ hiển thị / ẩn

+ Một vài sự kiện quan trọng của Form:

Click: sự kiện này xảy ra khi người dùng click chuột trên nút lệnh.

4) Label: hiển thị chuỗi ký tự tĩnh (người dùng không thể thay đổi). Thuộc tính **Caption** chứa chuỗi ký tự hiển thị.

5) TextBox: hiển thị chuỗi ký tự mà người dùng có thể thay đổi.

+ Một vài thuộc tính quan trọng:

Text: chuỗi ký tự hiển thị.

Font: font chữ.

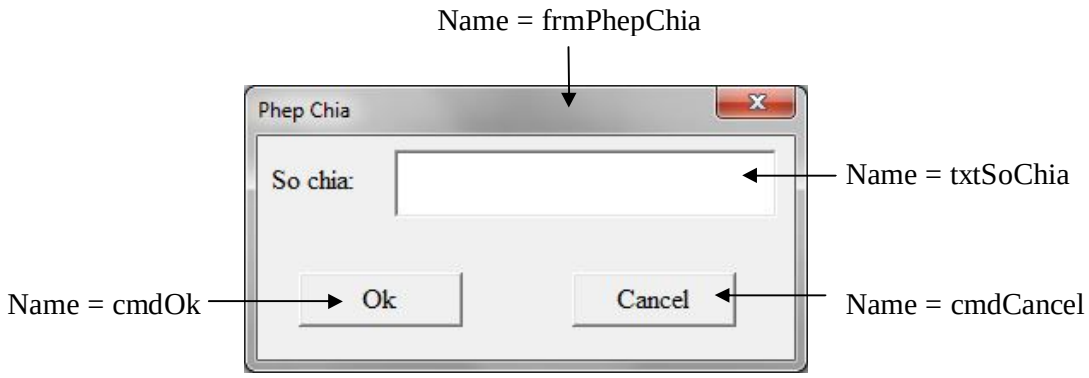
ForeColor: màu chữ.

MaxLength: số ký tự tối đa (=0 nghĩa là không giới hạn).

MultiLine = True / False ⇔ nhiều dòng / 1 dòng.

Ví dụ 2: chia các ô trong vùng chọn bởi 1 số.

Bước 1: Tạo Form như hình sau.



Bước 2: Viết các sự kiện sau.

```
Private Sub cmdCancel_Click()
Unload frmPhepChia
End Sub
```

```
Private Sub cmdOk_Click()
Dim SoChia As Double, cell As Range
SoChia = CDbl(txtSoChia.Text)
For Each cell In Selection
    cell.Value = cell.Value / SoChia
Next cell
End Sub
```

Bước 3: Viết thủ tục sau trong 1 Module.

```
Public Sub PhepChia()
frmPhepChia.Show
End Sub
```

Bước 4: Đưa thủ tục **PhepChia** ở bước 3 lên thanh công cụ.

+ **Right click** trên thanh **menu** chọn **Customize Quick Access Toolbar…**
+ Tại ô **Choose commands from**, chọn Macros. Click mục **PhepChia** trong ô phía dưới. Rồi click nút **Add**. Click **Ok**.
+ Trên thanh menu sẽ xuất hiện 1 nút **PhepChia**. Khi click vào nút này thì thủ tục **PhepChia** sẽ thực thi.

6) OptionButton / CheckBox: chọn 1 mục duy nhất / chọn nhiều mục cùng lúc.

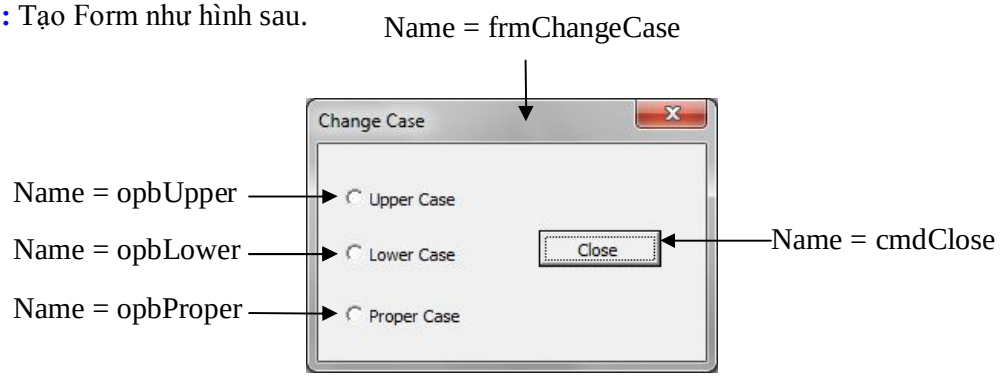
+ Một vài thuộc tính quan trọng:

Value = True / False ⇔ chọn / không chọn.

Caption: chữ hiển thị.

Ví dụ 3: đổi chữ hoa/thường cho các ô trong vùng chọn.

Bước 1: Tạo Form như hình sau.



Bước 2: Viết các sự kiện sau.

```
Private Sub cmdClose_Click()
Unload frmChangeCase
End Sub

Private Sub opbLower_Click()
Dim cell As Range

For Each cell In Selection
   cell.Value = LCase(cell.Value)
Next cell

End Sub
```

```
Private Sub opbProper_Click()
Dim cell As Range

For Each cell In Selection
   cell.Value = Application.WorksheetFunction.Proper(cell.Value)
Next cell

End Sub

Private Sub opbUpper_Click()
Dim cell As Range

For Each cell In Selection
   cell.Value = UCase(cell.Value)
Next cell

End Sub
```

7) ListBox: hiển thị 1 danh sách các mục (**item**).

+ Một vài thuộc tính quan trọng:

MultiSelect:

0 : chỉ được chọn 1 mục.
1 : cho phép chọn nhiều mục bằng cách click vào các mục cần chọn.
2 : cho phép chọn nhiều mục bằng cách nhấn CTRL hoặc SHIFT và click vào các mục cần chọn.

ListCount : tổng số mục có trong ListBox.

List : là mảng kiểu chuỗi, các phần tử được đánh chỉ số từ 0 đến (**ListCount** –1). Mỗi phần tử giữ chuỗi ký tự của 1 mục.

Selected : đây là 1 mảng kiểu Boolean với số phần tử giống như thuộc tính **List**.

Selected(i) = true ⇔ **List**(i) được chọn
Selected(i) = false ⇔ **List**(i) không được chọn

+ Một vài phương thức :

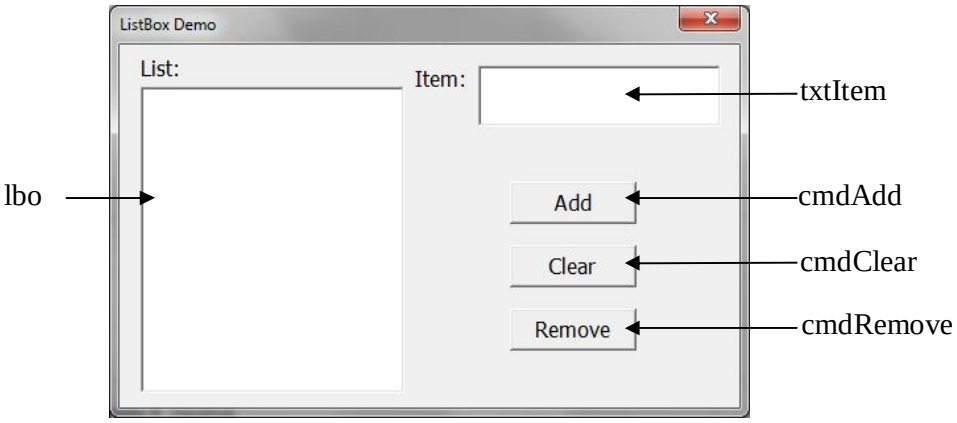
Xóa tất cả các mục: <Tên ListBox>.**Clear**

Thêm 1 mục vào ListBox: <Tên ListBox>.**AddItem** < chuỗi >

Xóa 1 mục: <Tên ListBox>.**RemoveItem** < index >

Ví dụ 4:

Bước 1: Tạo Form có Name là **frmListBox** như hình sau.



Bước 2: Viết các sự kiện sau.

```
Private Sub cmdAdd_Click()
lbo.AddItem txtItem.Text
End Sub

Private Sub cmdClear_Click()
lbo.Clear
End Sub

Private Sub cmdRemove_Click()
Dim i As Long

i = 0
While i <= (lbo.ListCount - 1)
   If lbo.Selected(i) = True Then
      lbo.RemoveItem i
   Else
      i = i + 1
   End If
Wend
End Sub
```

Bước 3: viết hàm sau trong 1 Module.

```
Public Function Test()
frmListBox.Show
End Function
```