


Nguyễn Phúc Khải

CHƯƠNG 8: *CÁC LỆNH ĐIỀU KHIỂN VÀ VÒNG LẶP*

Các nội dung:

- Lệnh đơn và lệnh phức
- Lệnh IF
- Lệnh SWITCH-CASE
- Lệnh WHILE
- Lệnh DO-WHILE
- Lệnh FOR
- Lệnh BREAK và lệnh CONTINUE
- Lệnh RETURN
- Lệnh GOTO
- Lệnh RỖNG

LỆNH ĐƠN & LỆNH PHỨC

- Lệnh đơn là một biểu thức thuộc loại bất kỳ theo sau nó là một dấu chấm phẩy (;), do đó lệnh đơn còn được gọi là lệnh biểu thức.
- **Ví dụ:** Các lệnh sau đây là các lệnh đơn

```
a = a + 1;
b >>= 3;
printf (...);
```

LỆNH ĐƠN & LỆNH PHỨC

- Lệnh phức bao hàm một hay nhiều lệnh đơn được bao bên trong cặp dấu ngoặc nhọn ({ }) và được bộ dịch C xem như là một lệnh đơn.
- **Ví dụ:** Xét lệnh if sau

```
if (a > 0)
   {
   i += 2;
   a++
   }
```

LỆNH ĐƠN & LỆNH PHỨC

- Các lệnh điều khiển này có thể được chia ra làm hai nhóm:
  - Nhóm lệnh liên quan đến việc rẽ nhánh chương trình: **if-else**, **switch-case**, **goto**,...
  - Nhóm lệnh lặp: **while**, **for**, **do_while**

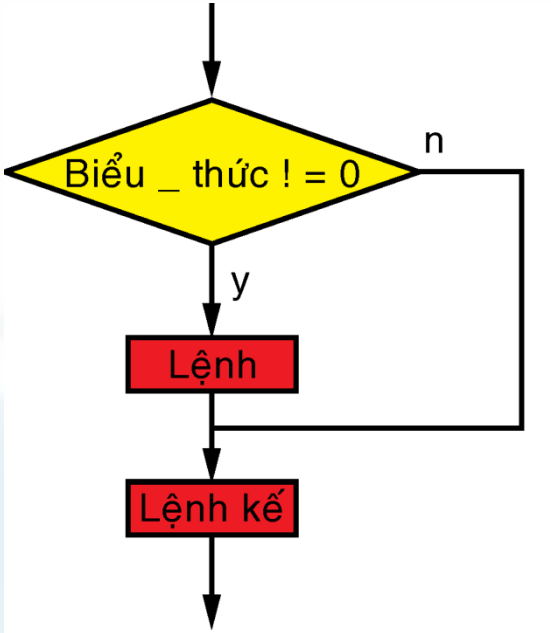
LỆNH IF

- Lệnh **if** cho phép lập trình viên thực hiện một lệnh đơn hay một lệnh phức tùy theo biểu thức điều kiện, nếu biểu thức có trị khác 0 thì lệnh được thực thi.
- ***Dạng 1:***

if (bieu_thuc)

lệnh;

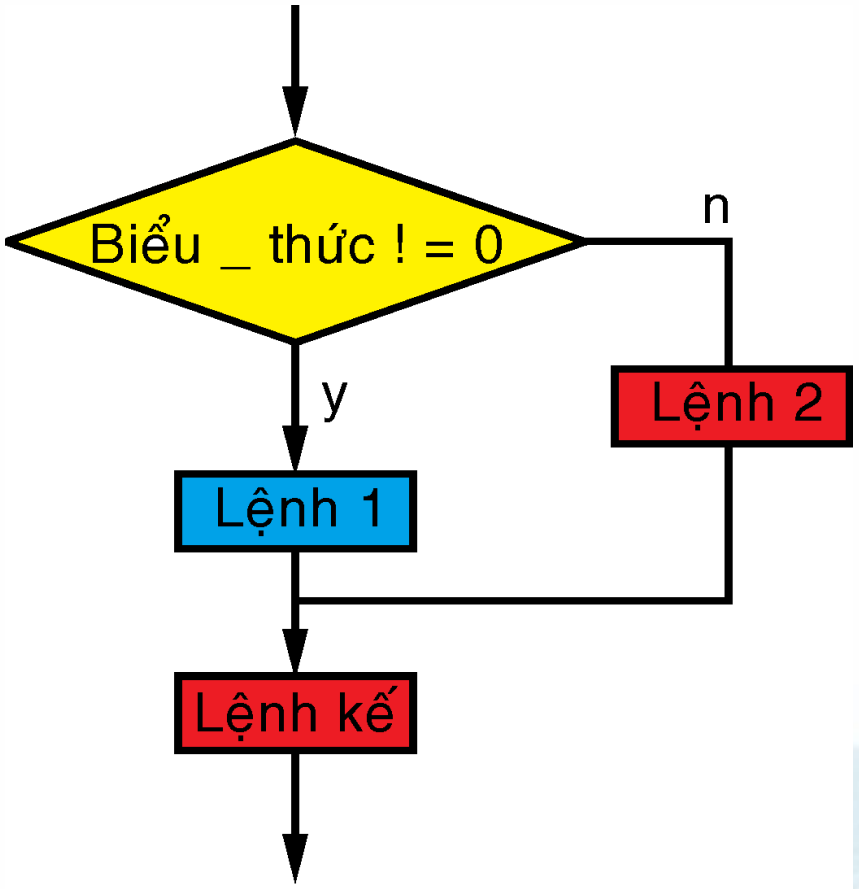
bieu_thuc là một biểu thức bất kỳ, có thể có hằng, biến hoặc gọi hàm trong đó và sau cùng là biểu thức này sẽ có trị 0 hoặc 1.



LỆNH IF

- ***Dạng 2:***

```
if (bieu_thuc)
    lệnh_1;
else
    lệnh_2;
```



LỆNH IF

```
#include <stdio.h>
#include <conio.h>
main()
   {   int n;
       clrscr();
       printf (Moi nhap mot so: );
       scanf (%d, &n);
       if (n % 2 == 0)
          printf ("So la so chan \n");
       printf ("Moi ban nhan mot phim de ket thuc
   \n");
             getch();          }
```

LỆNH IF

```
#include <stdio.h>
#include <conio.h>
main()
   {     int n;   clrscr();
         printf ("Moi nhap mot so: "); scanf ("%d", &n);
         if (n % 2 == 0)
           printf ("So la so chan \n");    ← vẫn có dấu chấm phẩy
         else
           printf ("So la so le \n");
         printf ("Moi ban nhan mot phim de ket thuc \n");
         getch();          }
```

LỆNH IF

Ví dụ: Xét chương trình sau đây:

```
if (a > 0)

        if (b > 0)

        c = b + a;
        else
         c = b – a;
```

```
 if (a > 0)
{
        if (b > 0)
         c = b – a;
}
else
        c = b – a;
```

LỆNH IF

```
if (biểu_thức_1)
    lenh_1;
else if (biểu_thức_2)
    lenh_2;
else if (biểu_thức_3)
    lenh_3;
.....
else
    lenh_n;
```

- Khi thực hiện lệnh **if_else** lồng nhau như thế này các biểu thức sẽ được tính lần lượt từ trên xuống dưới nếu có biểu thức nào khác 0, lệnh tương ứng với **if** đó sẽ được thi hành và toàn bộ phần còn lại của lệnh **if-else** được bỏ qua.

LỆNH IF

```
#include <stdio.h>
#include <conio.h>
main()
        {       char c;
                clrscr();
                printf ("Nhap mot ky tu: ");
                c = getchar();
```

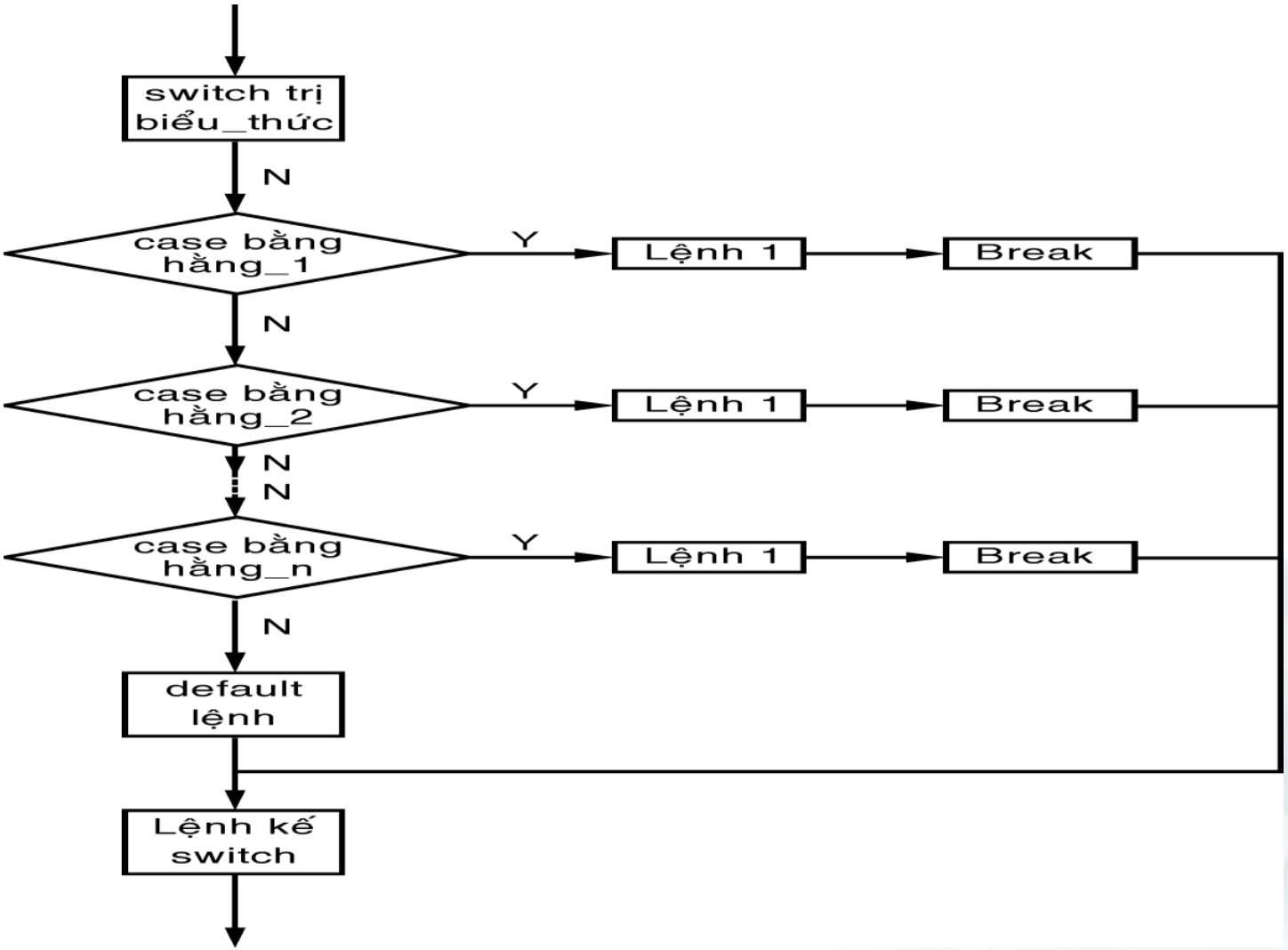
LỆNH IF

```
if (c == EOF)
        printf ("Da den cuoi file \n");
else if (c >= 'a' && c <= 'z')
        printf ("ky tu thuong\n");
else if (c >= 'A' && c <= 'Z')
        printf ("ky tu hoa\n");
else if (c >= '0' && c <= '9')
        printf ("ky tu so\n");
else
        printf ("ky tu khac\n");
getch();
}
```

LỆNH SWITCH-CASE

```
switch (biểu_thức)
    {
        case hằng_1:
                lệnh_1;
                break;
        case hằng_2:
                lệnh_2;
                break;
                :
                :
        case hằng_n:
                lệnh_n;
                break;
        default:
                lệnh;
                break;
    }
```

LỆNH SWITCH-CASE



LỆNH SWITCH-CASE

```
#include <stdio.h>
#include <conio.h>
main()
        {       int so;
                clrscr();
                printf ("Nhap mot so: ");
                scanf ("%d", &so);
                switch (so % 5)
                        {
                        case 0:
                                so += 5;
                                printf ("Tri la: %d\n", so);
                                break;
```

LỆNH SWITCH-CASE

```
            case 1:
                    so += 1;
                    printf ("Tri la: %d\n", so);
                    break;
            case 3:
                    so += 3;
                    printf ("Tri la: %d\n", so);
                    break;
            default:
                    printf ("Khong thoa\n");
                    break;
            }
    getch();
}
```

LỆNH SWITCH-CASE

- Lệnh **break** cuối mỗi **case** sẽ chuyển điều khiển chương trình ra khỏi lệnh **switch**. Nếu không có **break**, các lệnh tiếp ngay sau sẽ được thực thi dù các lệnh này có thể là của một **case** khác.

LỆNH SWITCH-CASE

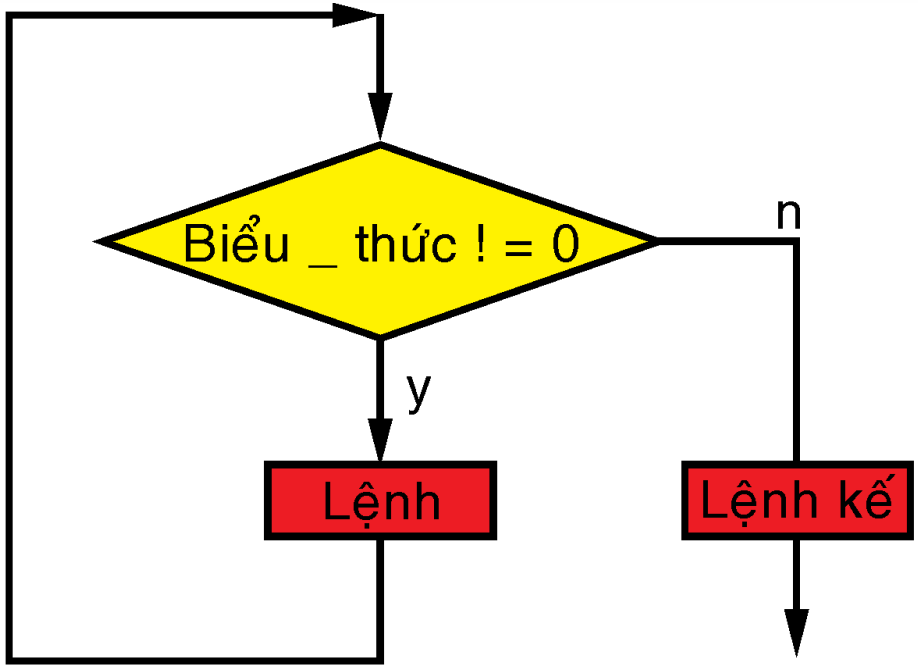
```
switch (thang)
        {
                case 4:
                case 6:
                case 9:
                case 11:
                        so_ngay = 30;
                        break;
```

LỆNH SWITCH-CASE

```
case 2:
        if (nam % 4 == 0)
                so_ngay = 29;
        else
                so_ngay = 28;
        break;
default:
        so_ngay = 31;
        break;          }
printf("Thang %d nam %d co %d ngay\n", thang, nam,
    so_ngay);
```

LỆNH WHILE

- Có thể nói **while** là lệnh lặp cơ bản của ngôn ngữ lập trình có cấu trúc, nó cho phép chúng ta lặp lại một lệnh hay một nhóm lệnh **trong khi** điều kiện còn đúng (true-tức khác 0). Cú pháp của lệnh **while**:




```
while (bieu-thuc)
      lenh
```

LỆNH WHILE

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>>
#include <time.h>
main()
        {       int i = 1;
                clrscr();
                randomize();
                printf ("So ngau nhien trong khoang 0-99 la:
   ");
```

LỆNH WHILE

```
while (i <= 10)
        {
            printf ("%d", random(100));
            i++;
        }
getch();
}
```

LỆNH WHILE

```
int i = 10;
clrscr();
randomize();
printf ("So ngau nhien trong khoang 0-99 la: ");
while (i)
        {
                printf ("%d", random(100));
                --i;
        }
```

LỆNH WHILE

```
#include <stdio.h>
#include <conio.h>
#define ESC 27
main()
        {
                char c;
                clrscr();
                printf ("Cac ky tu duoc nhap la: ");
                while (1)
                {
                        c = getche();
                        if (c == ESC)
                                break;
                }
        }
```

LỆNH WHILE

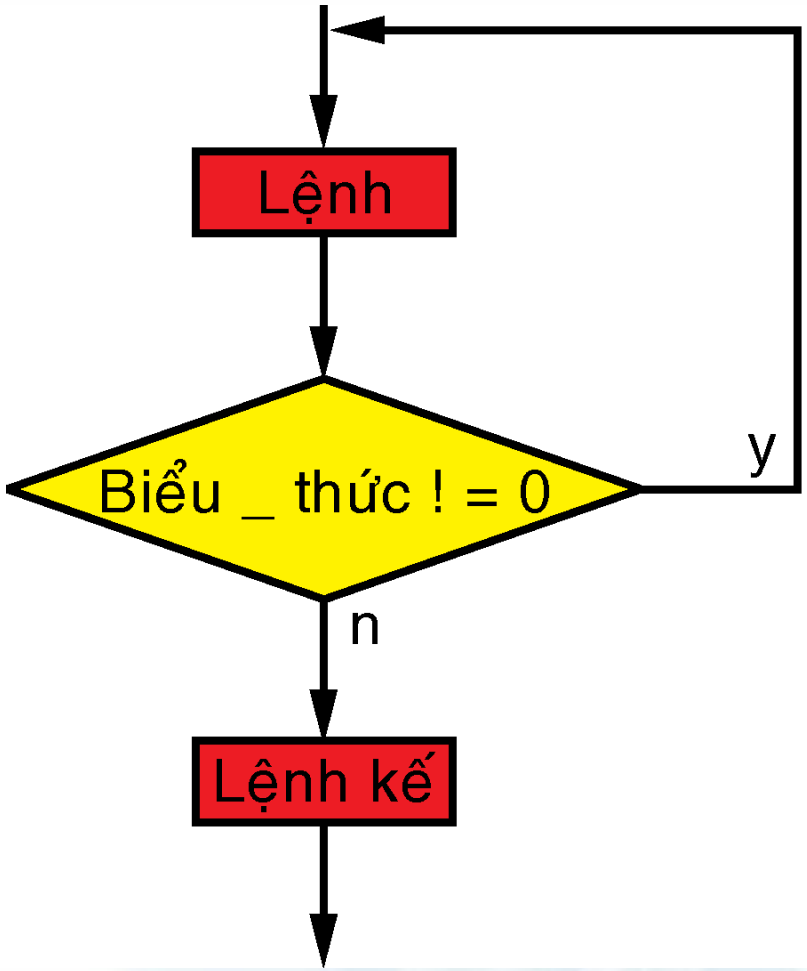
```
#include <stdio.h>
#include <conio.h>
#define ESC 27 main()
        {
                char c;
                clrscr();
                printf ("Cac ky tu duoc nhap la: ");
                while (getche() - ESC)
                        ;        ← lệnh thực thi rỗng

        }
```

LỆNH DO-WHILE

- Lệnh lặp **do-while** thực thi lệnh trước rồi mới kiểm tra điều kiện sau.
- Cú pháp của lệnh **do-while** như sau:

```
do
   lenh
while (bieu_thuc);
```



LỆNH DO-WHILE

```
#include <stdio.h>
#include <conio.h>
#define ESC 27
main()
   {
        char c;
        clrscr();
        printf ("\n Moi an cac phim mui ten \n");
```

LỆNH DO-WHILE

```
do
{
        c = getch();
        if (c == 0)
        {
                c = getch();
                switch(c)
                {
                case 'H':
                 printf ("Ban da an mui ten len\n");
                 break;
```

LỆNH DO-WHILE

```
                    case 'P':
                     printf ("Ban da an mui ten xuong\n");
                     break;
                    case 'K':
                     printf ("Ban da an mui ten qua trai\n");
                     break;
                    case 'M':
                    printf ("Ban da an mui ten qua phai\n");
break;
                    } /* end switch */
                }
        }while (c != 27);
    }
```

LỆNH DO-WHILE

- Chú ý rằng mỗi phím mũi tên khi được ấn đều sinh ra hai ký tự: ký tự đầu luôn là ký tự có mã ASCII là 0 (tức ký tự NUL), ký tự thứ hai là các mã ASCII tương ứng với phím, trong ví dụ trên thì
  - Phím mũi tên lên có mã là 0 và 'H‘
  - Phím mũi tên xuống có mã là 0 và 'P‘
  - Phím mũi tên qua trái có mã là 0 và 'K‘
  - Phím mũi tên có mã là 0 và 'M'.

LỆNH FOR

- Vòng lặp **for** là một lệnh lặp cho phép kiểm tra điều kiện trước, giống như **while**. Cú pháp của lệnh **for** như sau:

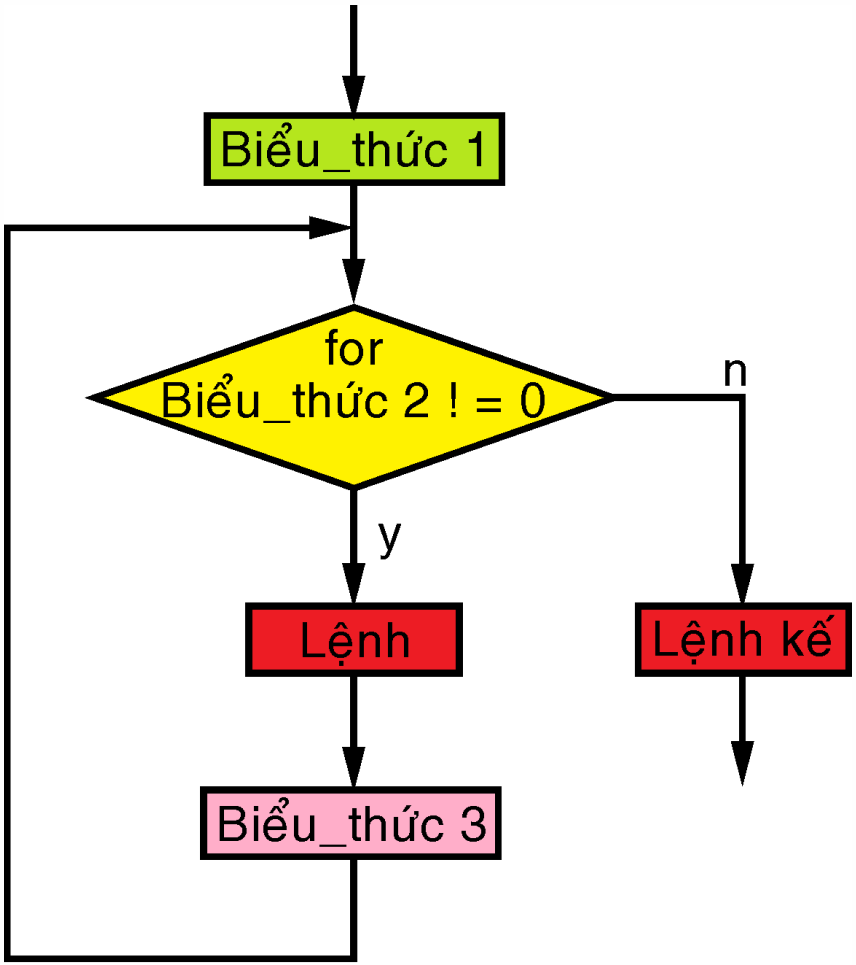
for (biểu_thức1; biểu_thức2; biểu_thức3)

lệnh

LỆNH FOR

- **biểu_thức1** là biểu thức để khởi động trị đầu cho biến điều khiển vòng for, nó có thể là biểu thức gán hay biểu thức phẩy, có thể không có.
- **biểu_thức2** là biểu thức cho phép kiểm tra xem vòng lặp có được tiếp tục lặp nữa hay không.
- **biểu_thức3** là biểu thức có ý nghĩa cho phép thay đổi biến điều khiển vòng lặp để vòng lặp tiến dần đến kết thúc. Biểu thức này được tính sau khi các lệnh thực thi trong thân vòng for được thực hiện xong.

LỆNH FOR



- **Ví dụ:** vòng lặp for để tính tổng từ 1 tới n như sau

```
s = 0;
for (i = 1; i <= n; i++)
    s += i;
```

- Có thể viết ngắn gọn hơn như sau

```
for (i = 1, s = 0; i <= n; i++)
    s += i;
```

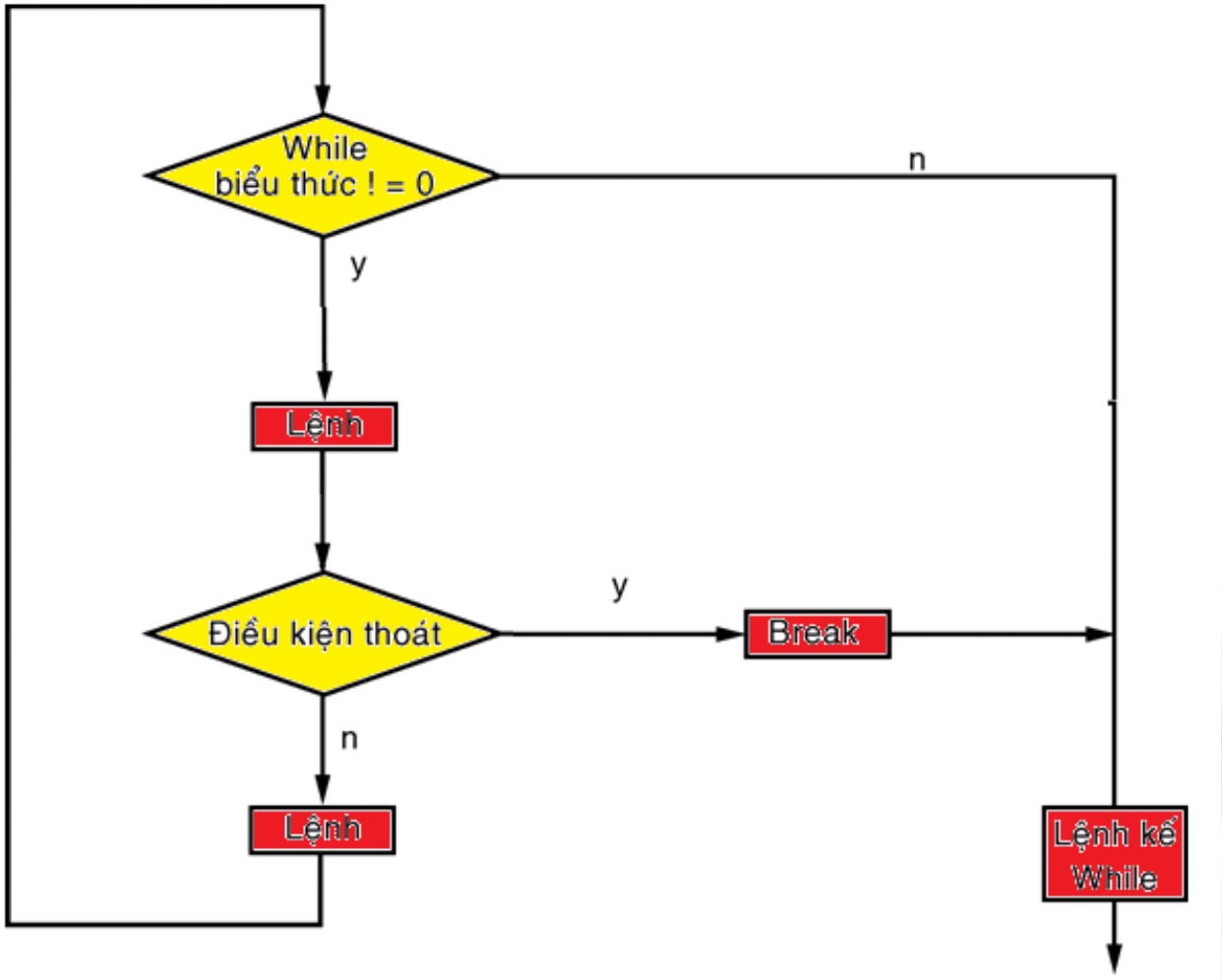
LỆNH FOR

```
#include <stdio.h>
#include <conio.h>
#define ESC 27
main()
        {       char c;
                clrscr();
                printf ("Cac ky tu duoc nhap la: ");
                for ( ; (c = getch()) != ESC;) ;
        }
```

LỆNH BREAK & LỆNH CONTINUE

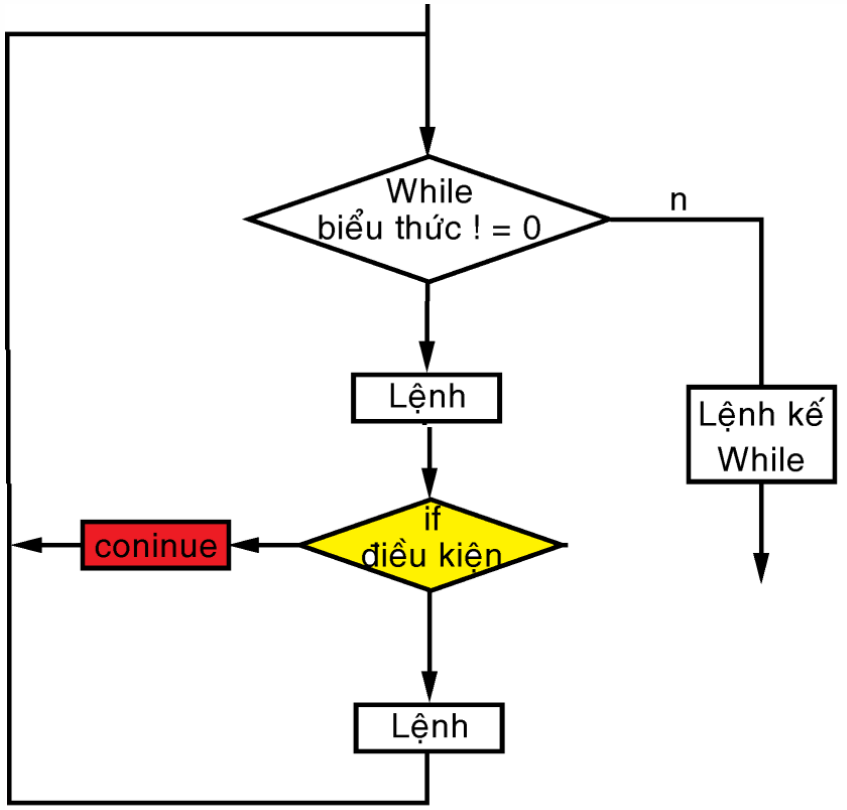
- Đây là hai lệnh nhảy không điều kiện của C, chúng cho phép lập trình viên có thể thay đổi tiến trình lặp của các cấu trúc lặp mà ta đã biết: **for**, **while**, **do-while**.
- **Lệnh break:**
  - Trong cấu trúc **switch-case**, lệnh **break** sẽ kết thúc lệnh **switch-case**;
  - Trong các cấu trúc lặp thì lệnh **break** cho phép thoát sớm ra khỏi vòng lặp (**while**, **for** hoặc **do-while**) chứa nó mà không cần xét điều kiện của lệnh kế tiếp sau vòng lặp.

LỆNH BREAK & LỆNH CONTINUE



LỆNH BREAK & LỆNH CONTINUE

- **Lệnh continue:** có tác dụng chuyển điều khiển chương trình về đầu vòng lặp chuẩn bị cho chu kỳ lặp mới, bỏ qua các lệnh còn lại nằm ngay sau lệnh nó trong chu kỳ lặp hiện hành. Lệnh này chỉ được dùng trong các vòng lặp, để bỏ qua các lệnh không cần thực thi trong vòng lặp khi cần thiết.



LỆNH BREAK & LỆNH CONTINUE

```
i = 0;
while (i <= 10)
        {
                i ++;
                if (i >= 6 && i <= 8)
                        continue;
                printf (" Tri hien thoi của i là %d\n", i);
        }
```

LỆNH BREAK & LỆNH CONTINUE

```
#include <stdio.h>
#include <conio.h>
main()
   {     double a[100];
         double tong;
         int i, n;
         clrscr();
         printf ("Co bao nhieu so can tinh: ");
         scanf ("%d", &n);
         printf ("Nhap cac so can tinh tong: ");
         for (i = 0; i < n; i++)
                  scanf ("%lf", &a[i]);
```

LỆNH BREAK & LỆNH CONTINUE

```
        for (i = 0, tong = 0; i < n; i++)
                {       if (a[i] <= 0)
                                continue;
                        tong += a[i];           }
        printf ("Tong cua cac so duong la %.2lf\n", tong);
                for (i = 0; i < n; i++)
                {       if (a[i] <= 0)
                                continue;
                        printf("Thuong cua tong voi so thu %d
la %5.2lf\n",i,tong/a[i]);                      }
        getch();
}
```

LỆNH RETURN

- Lệnh này dùng để thoát ra khỏi hàm hiện thời trở về hàm đã gọi nó, có thể trả về cho hàm gọi một trị. Lệnh này sẽ kết thúc hàm dù nó nằm ở đâu trong thân hàm. Khi gặp lệnh này C sẽ không thực hiện bất cứ lệnh nào sau lệnh return nữa. Các cú pháp của lệnh return như sau:

return;

return (biểu-thức);

return biểu-thức;

LỆNH RETURN

```
#include <stdio.h>
#include <conio.h>
#define ESC '\x1b'
void nhan_ky_tu (void);                    ← prototype của hàm
main()
        {
                char c;
                clrscr();
                printf ("Moi ban nhap cac ky tu: ");
                nhan_ky_tu ();             ← gọi hàm
        }
```

LỆNH RETURN

```
void nhan_ky_tu (void)
        {
                while (1)
                        if (getche() == ESC)
                                return;
        }
```

← *định nghĩa hàm*

LỆNH RETURN

- Ví dụ:
- Thiết kế hàm trả về kết quả so sánh hai số theo quy tắc sau đây:
  - số đầu > số sau: hàm trả về trị 1
  - số đầu = số sau: hàm trả về trị 0
  - số đầu < số sau: hàm trả về trị -1

LỆNH GOTO

- Mặc dù không ủng hộ nhưng C vẫn có lệnh rẽ nhánh không điều kiện **goto**, lệnh này cho phép chuyển điều khiển chương trình cho một lệnh nào đó.
- Cú pháp của lệnh **goto:**

goto nhãn;

- Với **nhãn** là một danh hiệu không chuẩn, danh hiệu này sẽ được đặt ở trước lệnh mà ta muốn nhảy đến theo cú pháp sau:

nhan: lệnh

LỆNH GOTO

- **nhãn** mà lệnh **goto** muốn nhảy đến phải nằm trong cùng một hàm với lệnh goto đó, do đó trong những hàm khác nhau có thể có các tên nhãn giống nhau, nhưng trong cùng một hàm các tên nhãn này phải khác nhau.

```
main()
        {
lap_lai:                clrscr();                                   ...
                if ((c = getch()) != ESC)
                        goto lap_lai;
        }
```

LỆNH RỖNG

- Trong C có khái niệm lệnh rỗng, lệnh này chỉ có một dấu chấm phẩy (;), nó rất cần thiết trong nhiều trường hợp, như đối với các vòng lặp, khi ta đặt các lệnh biểu thức thực thi vào trong các biểu thức của lệnh thì ta không cần có thêm lệnh thực thi làm thân cho chúng nữa, khi đó nếu để trống, C sẽ hiểu nhầm rằng lệnh kế tiếp sẽ là thân của vòng lặp, do đó chỉ còn cách cho một lệnh rỗng làm thân của chúng.

LỆNH RỖNG

- **Ví dụ 1:**

```
for (i = gt = 1; i <= n; gt *= i++)
                    ;
printf ("Giai thua %d! = %d\n", n, gt);
```

- **Ví dụ 2:**
- for (i =1,s = 0; i < 10; i++) ; s += i;
- printf("Tong la %d \n",s);

Kết thúc chương 8