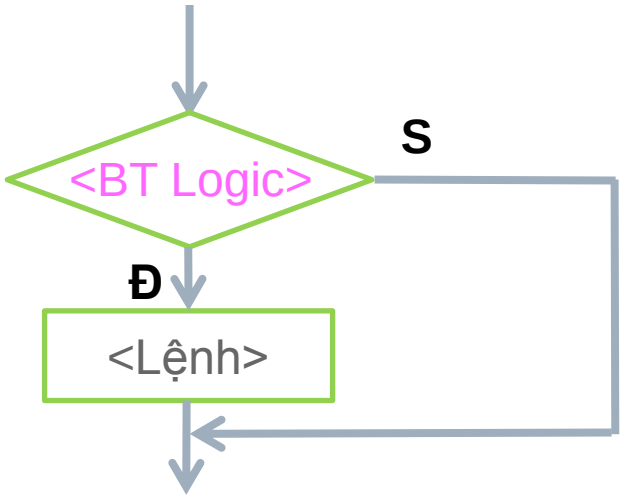
CƠ SỞ LẬP TRÌNH

CÁC CẤU TRÚC ĐIỀU KHIỂN TRONG NGÔN NGỮ C

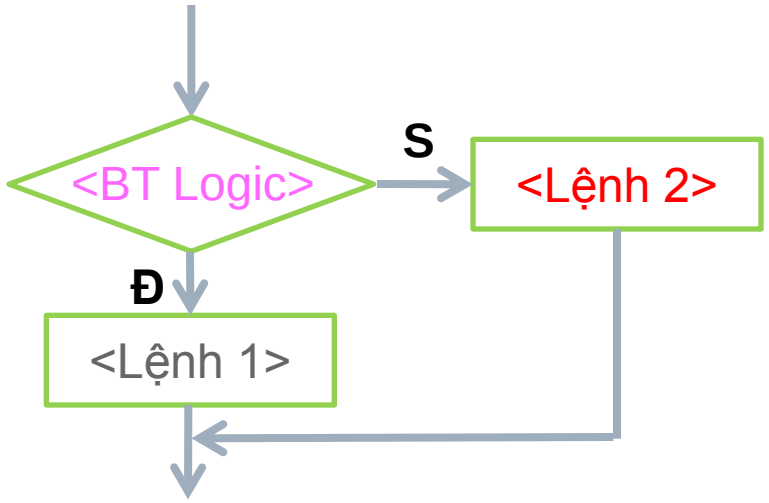
Nội dung

- Cấu trúc rẽ nhánh
  - Câu lệnh điều kiện if
  - Câu lệnh rẽ nhánh switch
  - Toán tử goto
- Cấu trúc lặp
  - Vòng lặp xác định for
  - Vòng lặp không xác định while
  - Vòng lặp không xác định do … while

Cấu trúc rẽ nhánh: Câu lệnh if … else




```
if (<BT Logic>)
        <Lệnh>;
```




```
if (<BT Logic>)
        <Lệnh 1>;
else
        <Lệnh 2>;
```

Cấu trúc rẽ nhánh: Câu lệnh if … else

```
void main()
{
        if (a == 0)
                printf("a bang 0");
        else
                printf("a khac 0");

        if (a == 0)
        {
                printf("a bang 0");
                a = 2912;
        }
        else
                printf("a khac 0");
}
```

Cấu trúc rẽ nhánh: Câu lệnh if … else

- Câu lệnh if và câu lệnh if… else là một câu lệnh đơn

```
{
        if (a == 0)
                printf("a bang 0");
}
{
        if (a == 0)
        {
                printf("a bang 0");
                a = 2912;
        }
        else
                printf("a khac 0");
}
```

Cấu trúc rẽ nhánh: Câu lệnh if … else

- Câu lệnh if có thể lồng vào nhau và else sẽ tương ứng với if gần nó nhất.

```
if (a != 0)
        if (b > 0)
                printf("a != 0 va b > 0");
        else
        printf("a != 0 va b <= 0");

if (a !=0)
{
        if (b > 0)
                printf("a != 0 va b > 0");
        else
                printf("a != 0 va b <= 0");
}
```

Cấu trúc rẽ nhánh: Câu lệnh if … else

- Nên dùng else để loại trừ trường hợp.

```
if (delta < 0)
        printf("PT vo nghiem");
if (delta == 0)
        printf("PT co nghiem kep");
if (delta > 0)
        printf("PT co 2 nghiem");

if (delta < 0)
        printf("PT vo nghiem");
else    // delta >= 0
        if (delta == 0)
                printf("PT co nghiem kep");
        else
                printf("PT co 2 nghiem");
```

Cấu trúc rẽ nhánh: Câu lệnh if … else

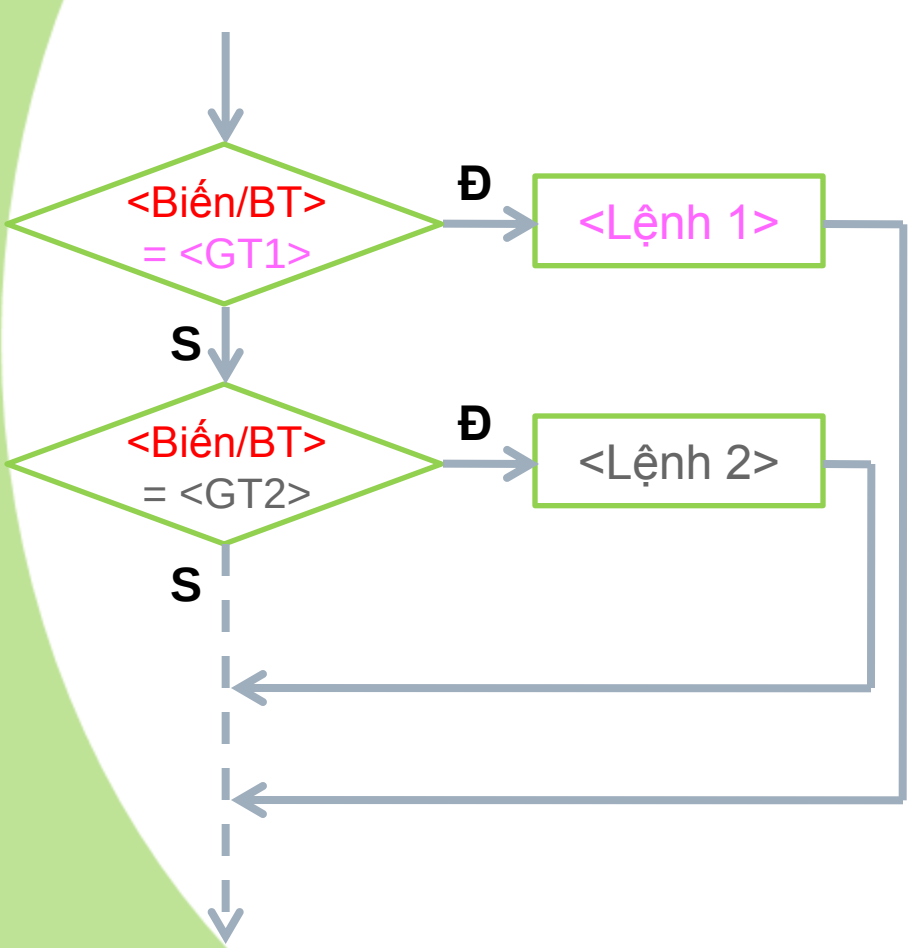
- Không được thêm ; sau điều kiện của if.

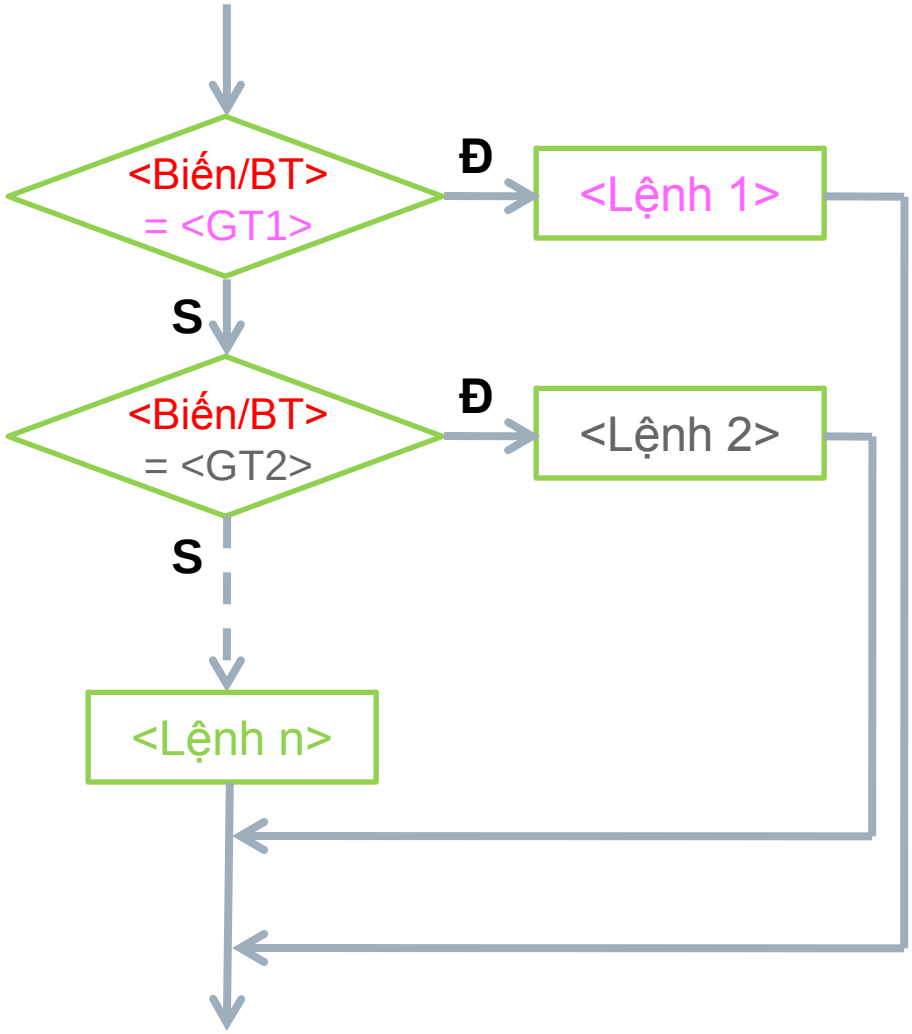
```
void main()
{
        int a = 0;
        if (a != 0)
                printf("a khac 0.");

        if (a != 0);
                printf("a khac 0.");

        if (a != 0)
        {
        };
        printf("a khac 0.");
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch





Cấu trúc rẽ nhánh: Câu lệnh switch

```
void main()
{
        int a;
        printf("Nhap a: ");
        scanf("%d", &a);

        switch (a)
        {
                case 1 : printf("Mot"); break;
                case 2 : printf("Hai"); break;
                case 3 : printf("Ba"); break;
        }
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch

```
void main()
{
        int a;
        printf("Nhap a: ");
        scanf("%d", &a);

        switch (a)
        {
                case 1 : printf("Mot"); break;
                case 2 : printf("Hai"); break;
                case 3 : printf("Ba"); break;
                default : printf("Ko biet doc");
        }
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch

- Câu lệnh switch là một câu lệnh đơn và có thể lồng nhau.

```
{
        switch (a)
        {
                case 1 : printf("Mot"); break;
                case 2 : switch (b)
                         {
                                 case 1 : printf("A"); break;
                                 case 2 : printf("B"); break;
                         } break;
                case 3 : printf("Ba"); break;
                default : printf("Khong biet doc");
        }
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch

- Các giá trị trong mỗi trường hợp phải khác nhau.

```
switch (a)
{
        case 1 : printf(“Mot”); break;
        case 1 : printf(“MOT”); break;
        case 2 : printf(“Hai”); break;
        case 3 : printf(“Ba”); break;
        case 1 : printf(“1”); break;
        case 1 : printf(“mot”); break;
        default : printf(“Khong biet doc”);
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch

- switch sẽ nhảy đến case tương ứng và thực hiện đến khi nào gặp break hoặc cuối switch sẽ kết thúc.

```
switch (a)
{
        case 1 : printf("Mot"); break;
        case 2 : printf("Hai"); break;
        case 3 : printf("Ba"); break;
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch

- switch nhảy đến case tương ứng và thực hiện đến khi nào gặp break hoặc cuối switch sẽ kết thúc.

```
switch (a)
{
        case 1 : printf(“Mot”); break;
        case 2 : printf(“Hai”); break;
        case 3 : printf(“Ba”); break;
}
switch (a)
{
        case 1 : printf(“Mot”); break;
        case 2 : printf(“Hai”); break;
        case 3 : printf(“Ba”); break;
}
```

Cấu trúc rẽ nhánh: Câu lệnh switch

- Tận dụng tính chất khi bỏ break;

```
switch (a)
{
        case 1 : printf("So le"); break;
        case 2 : printf("So chan"); break;
        case 3 : printf("So le"); break;
        case 4 : printf("So chan"); break;
}
switch (a)
{
        case 1 :
        case 3 : printf("So le"); break;
        case 2 :
        case 4 : printf("So chan"); break;
}
```

Cấu trúc rẽ nhánh: Toán tử goto

- Nhãn được viết như tên biến và có thêm dấu: (hai chấm) đứng sau, nhãn có thể được gán cho bất kì câu lệnh nào trong chương trình
- Lệnh nhảy goto có dạng:

  goto nhan;

  - Khi gặp lệnh này, máy nhảy đến nhãn viết sau từ khoá goto
- Ví dụ:

```
main()
{       int i;
        vaosl: printf("Nhap i: "); scanf("%d",&i);
        if (n<10) goto vaosl;
}
```

Cấu trúc rẽ nhánh: Bài tập thực hành

1. Nhập một số bất kỳ. Hãy đọc giá trị của số nguyên đó nếu nó có giá trị từ 1 đến 9, ngược lại thông báo không đọc được.
2. Nhập một chữ cái. Nếu là chữ thường thì đổi sang chữ hoa, ngược lại đổi sang chữ thường.
3. Giải phương trình bậc nhất $ax + b = 0$.
4. Giải phương trình bậc hai $ax^2 + bx + c = 0$.

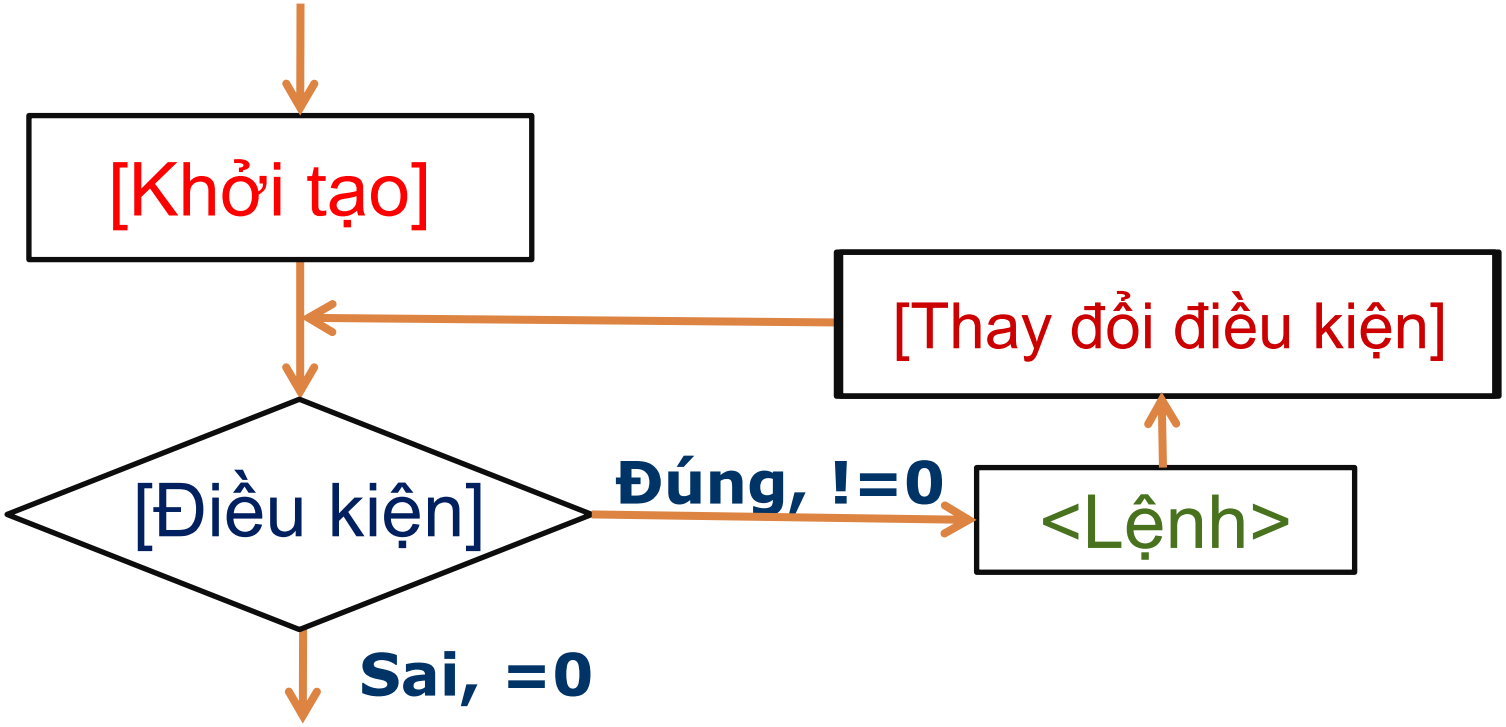
Cấu trúc rẽ nhánh: Bài tập thực hành

5. Nhập 4 số nguyên a, b, c và d. Tìm số có giá trị nhỏ nhất (min).
6. Nhập 4 số nguyên a, b, c và d. Hãy sắp xếp giá trị của 4 số nguyên này theo thứ tự tăng dần.
7. Tính tiền đi taxi từ số km nhập vào. Biết:
   a. 1 km đầu giá 15000đ
   b. Từ km thứ 2 đến km thứ 5 giá 13500đ
   c. Từ km thứ 6 trở đi giá 11000đ
   d. Nếu trên 120km được giảm 10% tổng tiền.

Cấu trúc rẽ nhánh: Bài tập thực hành

8. Nhập vào tháng và năm. Cho biết tháng đó có bao nhiêu ngày.
9. Nhập độ dài 3 cạnh 1 tam giác. Kiểm tra đó có phải là tam giác không và là tam giác gì?

Cấu trúc lặp: Câu lệnh for



for ([Khởi tạo]**;** [Điều kiện]**;** [Thay đổi điều kiện]**)**

<Lệnh>**;**

Cấu trúc lặp: Câu lệnh for

BÀI TOÁN: Tính tổng n số tự nhiên đầu tiên
INPUT: Số tự nhiên n nhập từ bàn phím
OUTPUT: Tổng n số tự nhiên đầu tiên S=1+2+…+n

THUẬT TOÁN:

c 1. Nhập số tự nhiên *n*
Bước 2. Gán *Tong* = 0; *Dem* = *1*;
c 3. Nếu *Dem* <= *n* là sai,đưa ra *Tong* và kết thúc
c 4. Gán *Tong* = *Tong* + *Dem*
Bước 5. Tăng số đếm: *Dem* = *Dem* + 1
Bước 6. Quay lại Bước 3

Cấu trúc lặp: Câu lệnh for

BÀI TOÁN: Tính tổng n số tự nhiên đầu tiên
INPUT: Số tự nhiên n nhập từ bàn phím
OUTPUT: Tổng n số tự nhiên đầu tiên S=1+2+…+n

```
#include <stdio.h>
#include <conio.h>
int main()
{
  int n, tong, dem;
  printf("Nhap so tu nhien n= "); scanf("%d",&n);
  tong = 0;
  for (dem = 0; dem <=n; dem++)
    {
      tong = tong + dem;
    }
  printf("Tong %d so tu nhien dau tien la: %d",n , tong);
  getch();
}
```

Cấu trúc lặp: Câu lệnh for

- Trong câu lệnh for, có thể sẽ không có phần [Khởi tạo]
- Ví dụ: Viết ra màn hình các số từ 1 đến 10

Cách viết 1:

```
int i;
for (i = 1; i <= 10; i++)
      printf("%d ", i);
```

Cách viết 2:

```
int i = 1;
for (; i <= 10; i++)
      printf("%d", i);
```

Cấu trúc lặp: Câu lệnh for

- Trong câu lệnh for, có thể sẽ không có phần [Điều kiện]
  - Khi đó [Điều kiện] là luôn luôn đúng
- Ví dụ: Viết ra màn hình các số từ 1 đến 10

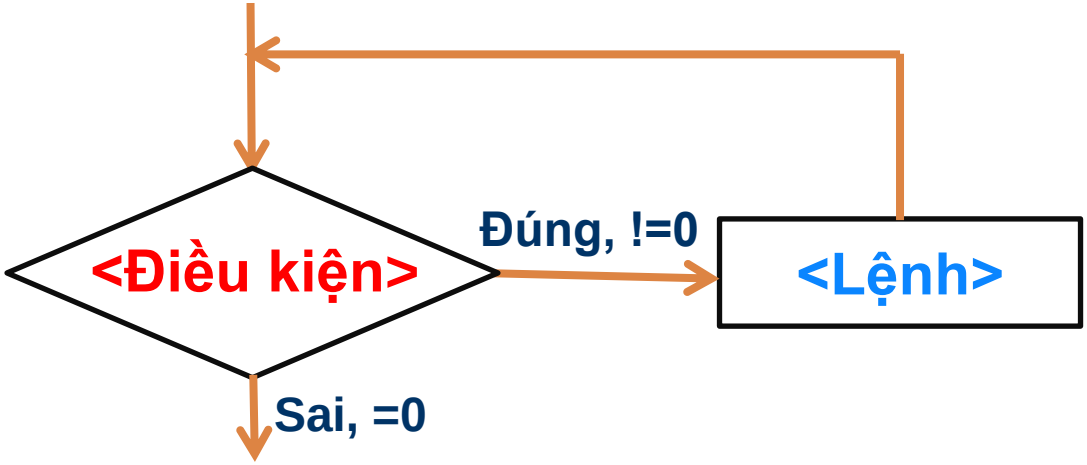
Cách viết 1:

```
int i;
for (i = 1; i <= 10; i++)
      printf("%d ", i);
```

Cách viết 2:

```
for (i = 1; ; i++)
{
      if (i > 10) break;
      printf("%d", i);
}
```

Cấu trúc lặp: Câu lệnh while



while (<Điều kiện>**)**

<Lệnh>**;**

Cấu trúc lặp: Câu lệnh while

BÀI TOÁN: Nhập một dãy số tự nhiên từ bàn phím cho đến khi số 0 được nhập và đếm số lượng số vừa nhập
INPUT: Nhập các số nguyên từ bàn phím
OUTPUT: Số lượng số vừa nhập

THUẬT TOÁN:

Bước 1. Gán *Dem = 0*;
c 2. Nhập một số nguyên *k*;
c 3. Nếu *k = 0* là đúng,đưa ra *Dem* và kết thúc
Bước 4. Tăng số đếm: *Dem = Dem + 1*
c 5. Nhập một số nguyên *k*;
Bước 6. Quay lại Bước 3

Cấu trúc lặp: Câu lệnh while

BÀI TOÁN: Nhập một dãy số tự nhiên từ bàn phím cho đến khi số 0 được nhập và đếm số lượng số vừa nhập
INPUT: Nhập các số nguyên từ bàn phím
OUTPUT: Số lượng số vừa nhập

```c
#include <stdio.h>
#include <conio.h>
int main()
{
  int k, dem = 0;
  scanf("%d",&k);
  while (k!=0)
  {
    dem++;
    scanf("%d",&k);
  }
  printf("So luong cac so nguyen da nhap la: %d",dem);
  getch();
}
```

Cấu trúc lặp: Câu lệnh while

- Câu lệnh while có thể bị lặp vô tận (loop)

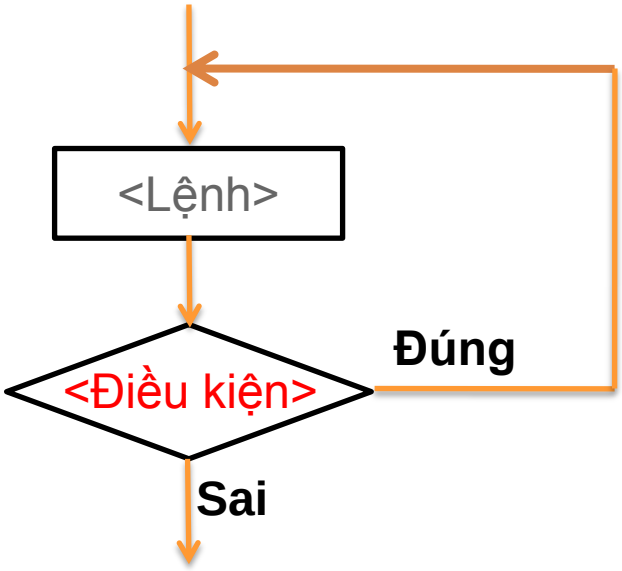
```
void main()
{
        int n;
        n = 1;
        while (n < 10) printf("%d", n);
}
```

```
void main()
{
        int n = 1;
        while (1)
                printf("%d", n);
}
```

- Lặp vô hạn chỉ kết thúc được nhờ câu lệnh break

➔ Trong thân vòng lặp, cần có lệnh thay đổi giá trị <Điều kiện>

Cấu trúc lặp: Câu lệnh do … while



do

<Lệnh>**;**

while (<Điều kiện>**)**

Cấu trúc lặp: Câu lệnh do … while

BÀI TOÁN: Nhập một dãy số tự nhiên từ bàn phím cho đến khi số 0 được nhập và đếm số lượng số vừa nhập
INPUT: Nhập các số nguyên từ bàn phím
OUTPUT: Số lượng số vừa nhập

```
#include <stdio.h>
#include <conio.h>
int main()
{
  int k, dem = 0;
  printf("Nhap day cac so nguyen cho den khi nhap so 0\n");
  do
  {
    scanf("%d",&k);
    dem++;
  } while (k!=0);
  printf("So luong cac so nguyen da nhap la: %d",dem-1);
  getch();
}
```

Cấu trúc lặp: Câu lệnh do … while

- Câu lệnh do… while có thể bị lặp vô tận (loop)

```
        int n = 1;
        do
                printf("%d", n);
        while (n < 10);
```

```
void main()
{
        int n = 1;
        do
                printf("%d", n);
        while (1)
}
```

Một số lưu ý

- Cho phép ra khỏi for, while, do…while và switch
- Khi có nhiều chu trình lồng nhau, break đưa máy ra khỏi chu trình bên trong nhất chứa nó.
- Ví dụ: Thuật toán kiểm tra tính nguyên tố của n

```
int i,n, ng_to = 1;
for (i=2;i<=sqrt(n);i++)
   if (n%i==0)
      { ng_to=0;
        break;
      }
if (ng_to)
      printf("\n%d la so nguyen to",n);
else printf("\n%d la hop so",n);
```

Một số lưu ý (tt)

- continue dùng để bắt đầu một vòng mới của chu trình bên trong nhất chứa nó:
  - Trong thân toán tử for: máy sẽ chuyển tới bước khởi đầu lại
  - Trong thân của while hoặc do…while: máy chuyển tới xác định giá trị biểu thức (viết sau while), sau đó tiến hành kiểm tra điều kiện kết thúc chu trình
- Lưu ý: continue không áp dụng cho switch

Ví dụ câu lệnh continue

```
#include <stdio.h>
void main()
{ int i;
  for (i=1;i<=5;i++)
      { printf("\nBat dau %d",i);
        if (i<4) continue;
       printf("\nChao ban");
      }
}
```

- Kết quả:

```
Bat dau 1
Bat dau 2
Bat dau 3
Bat dau 4
Chao ban
Bat dau 5
Chao ban
```

Cấu trúc lặp: Bài tập thực hành

1. Nhập một số nguyên dương n (n > 0).

   Hãy cho biết:

   a. Có phải là số đối xứng? Ví dụ: 121, 12321, …
   b. Có phải là số chính phương? Ví dụ: 4, 9, 16, …
   c. Có phải là số nguyên tố? Ví dụ: 2, 3, 5, 7, …
   d. Chữ số lớn nhất và nhỏ nhất?
   e. Các chữ số có tăng dần hay giảm dần không?

Cấu trúc lặp: Bài tập thực hành

2. Nhập một số nguyên dương n. Tính:
   a. $S = 1 + 2 + \ldots + n$
   b. $S = 1^2 + 2^2 + \ldots + n^2$
   c. $S = 1 + 1/2 + \ldots + 1/n$
   d. $S = 1*2*\ldots*n = n!$
   e. $S = 1! + 2! + \ldots + n!$
3. Nhập 3 số nguyên a, b và n với a, b < n. Tính tổng các số nguyên dương nhỏ hơn n chia hết cho a nhưng không chia hết cho b.
4. Tính tổng các số nguyên tố nhỏ hơn n
   (0 < n < 50000)

Cấu trúc lặp: Bài tập thực hành

5. Nhập một số nguyên dương n. Xuất ra số ngược lại. Ví dụ: Nhập 1706 ➔ Xuất 6071.
6. Tìm và in lên màn hình tất cả các số nguyên trong phạm vi từ 10 đến 99 sao cho tích của 2 chữ số bằng 2 lần tổng của 2 chữ số đó.
7. Tìm bội số chung lớn nhất của 2 số nguyên dương a và b nhập từ bàn phím.
8. Nhập n. In n số đầu tiên trong dãy Fibonacy biết

$$F_0=F_1=1$$

$$F_n = F_{n-1} + F_{n-2}$$