

Cấu trúc điều khiển

GV. Nguyễn Minh Huy

Nội dung



- Biểu thức và toán tử trong C.
- Cấu trúc rẽ nhánh.
- Cấu trúc lặp.



- **Biểu thức và toán tử trong C.**
- Cấu trúc rẽ nhánh.
- Cấu trúc lặp.



■ Biểu thức trong C:

- Là một dãy hữu hạn các toán hạng và toán tử.

$a + b - d * c / e$

$(x >> (p + 1 - n)) \& \sim(\sim 0 << n)$

- Toán hạng: biến, hằng.

- Toán tử:

- Toán tử một ngôi: **<phép toán> a** $\rightarrow \sim a, !b, ++c$.
- Toán tử hai ngôi: **a <phép toán> b** $\rightarrow a + b, x / y$.
- Toán tử ba ngôi: toán tử điều kiện () ? :

- Kết quả biểu thức: một giá trị số.

Biểu thức và toán tử trong C



■ Toán tử số học:

■ Ký hiệu: +, -, *, /, %.

- % chỉ dùng với số nguyên.
- / kết quả phụ thuộc toán hạng.

```
int a = 5 % 3; // Đúng
```

```
float x = 5 % 3.0; // Sai
```

```
int b = 5 / 3; // Chia nguyên
```

```
float y = 5.0 / 3; // Chia thực
```

■ Toán tử so sánh:

■ Ký hiệu: >, <, >=, <=, ==, !=.

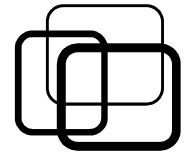
■ Kết quả: 1 (true), 0 (false).

```
int a = 5 > 3; // 1 (true)
```

```
int b = 5 == 3; // 0 (false)
```

```
int c = 5 != 3; // 1 (true)
```

Biểu thức và toán tử trong C



■ Toán tử logic:

■ Ký hiệu:

- ! (not), && (and), || (or).

■ Kết nối biểu thức so sánh.

■ Kết quả: 1 (true), 0 (false).

```
int a = (5 > 3) && (4 > 7); // 0 (false)
```

```
int b = (5 > 3) || (4 > 7); // 1 (true)
```

```
int c = !(5 == 3); // 1 (true)
```

■ Toán tử trên bit:

■ Ký hiệu:

- & (and), | (or), ^ (xor).
- ~ (bù).
- >> (dịch phải), << (dịch trái).

■ Thao tác trên bit dữ liệu.

```
short a = 5 & 6; // 0101 and 0110
```

```
short b = 5 | 6; // 0101 or 0110
```

```
unsigned short c = ~1; // not 0001
```

```
short d = a >> 1;
```

Biểu thức và toán tử trong C



■ Toán tử tăng, giảm:

- Ký hiệu: ++, --.
- Tăng/giảm 1 đơn vị trên biến.
 - Tiền tố: tính trước biểu thức.
 - Hậu tố: tính sau biểu thức.

```
int a = 5++;      // Sai
int a = 5;
int b = ++a * 4;   // b = 24
int c = a++ * 4;   // b = 20
```

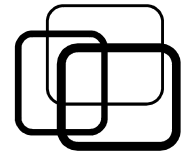
■ Toán tử gán:

- Ký hiệu: =, <phép toán>=
a <phép toán>= b;
→ a = a <phép toán> b;
- <phép toán>:
 - +, -, *, /, %, &, |, ^, >>, <<.

```
int a = 5;
int b, c, d;
d = c = b = a; // b = a
                // c = b
                // d = c

int e += a;     // e = e + a
int f *= a + 1; // f = f * (a + 1)
```

Biểu thức và toán tử trong C



■ Độ ưu tiên toán tử:

Operators	Associativity
() [] -> .	left to right
! ~ ++ -- + - * (type) sizeof	right to left
* / %	left to right
+ -	left to right
<< >>	left to right
< <= > >=	left to right
== !=	left to right
&	left to right
^	left to right
	left to right
&&	left to right
	left to right
? :	right to left
= += -= *= /= %= &= ^= = <<= >>=	right to left
,	left to right



- Biểu thức và toán tử trong C.
- **Cấu trúc rẽ nhánh.**
- Cấu trúc lặp.

Cấu trúc rẽ nhánh



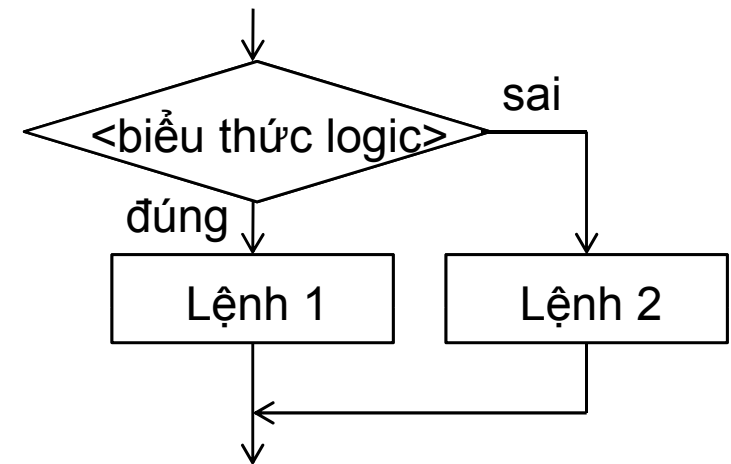
■ Câu lệnh if-else:

■ Cú pháp:

```
if (<biểu thức logic>
    <Lệnh 1>;
[else
    <Lệnh 2>;]
```

Mã giả:

Nếu <biểu thức logic>
Lệnh 1
[Ngược lại
Lệnh 2]

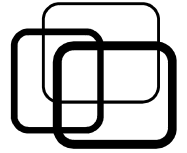


■ Ví dụ:

```
// Câu lệnh if-else đủ
if (n > 0)
    a = a * 2;
else
    a = a / 2;
```

```
// Bỏ mệnh đề else
if (n > 0)
    a = a * 2;
```

```
// Dùng khối lệnh
if (n > 0)
{
    a = a * 2;
    b = b + 1;
}
```



■ Câu lệnh if-else:

■ Lưu ý:

- Biểu thức logic phải đặt giữa ().
 - ➔ Giá trị 1: true.
 - ➔ Giá trị 0: false.
- **if-else** là câu lệnh phức.
 - ➔ Không có ; sau **if** hoặc **else**.
- **if-else** có thể lồng nhau.
 - ➔ **else** tương ứng **if** gần nhất.

```
if n > 0           // Sai
    a = a * 2;
```

```
if (1)             // Luôn đúng
    a = a * 2;
```

```
if (n > 0) ;       // Sai
    a = a * 2;
else ;
    a = a / 2;
```

```
if (n > 0)          // if-else lồng
    if (a > b)
        c = c + 1;
    else
        c = c - 1;
```



■ Câu lệnh if-else:

- if-else lồng nhau, kiểm tra điều kiện trên cùng 1 biến:

```
if (dtb >= 8)
    loai = "Gioi";
else
    if (dtb >= 6.5)
        loai = "Kha";
    else
        if (dtb >= 5)
            loai = "Trung binh";
        else
            loai = "Yeu";
```

```
if (dtb >= 8)
    loai = "Gioi";
else if (dtb >= 6.5)
    loai = "Kha";
else if (dtb >= 5)
    loai = "Trung binh";
else
    loai = "Yeu";
```



■ Câu lệnh switch-case:

■ Cú pháp:

```
switch (<biểu thức>
{
    [case <giá trị 1>:
        <Lệnh 1>;
        break;
    case <giá trị 2>:
        <Lệnh 2>;
        break;
    ....]
    [default:
        <Lệnh N>;]
}
```

// Câu lệnh if-else tương đương

```
if (<biểu thức> == <giá trị 1>)
    <Lệnh 1>;
else if (<biểu thức> == <giá trị 2>)
    <Lệnh 2>;
...
else
    <Lệnh N>;
```



■ Câu lệnh switch-case:

```
switch (thu)
{
    case 1:
        printf("Chu nhat"); break;
    case 2:
        printf("Thu hai"); break;
    case 3:
        printf("Thu ba"); break;
    case 4:
        printf("Thu tu"); break;
    case 5:
        printf("Thu nam"); break;
    case 6:
        printf("Thu sau"); break;
    case 7:
        printf("Thu bay"); break;
}
```



■ Câu lệnh switch-case:

■ Lưu ý:

- Biểu thức phải đặt giữa ().
- Giá trị ở mệnh đề **case**:
 - ➔ Giá trị đơn.
 - ➔ Không là miền giá trị.
- Câu lệnh **break**:
 - ➔ Ngắt giữa các **case**.
 - ➔ Có thể bỏ để ghép các **case**.

```
switch a + b // Sai
{
    ...
}

switch (a + b)
{
    case > 5: // Sai
        ...
}
```



■ Câu lệnh switch-case:

```
switch (thu)
{
    case 2:
    case 3:
    case 4:
    case 5:
    case 6:
        printf("Ngay lam viec"); break;
    case 1:
    case 7:
        printf("Ngay nghi"); break;
    default:
        printf("Ngay khong ton tai");
}
```




- Biểu thức và toán tử trong C.
- Cấu trúc rẽ nhánh.
- **Cấu trúc lặp.**



- Xét chương trình xuất số:
 - Xuất các số nguyên từ 1 đến 10.
 - Thực hiện 10 lần lệnh xuất.
 - Xuất các số nguyên từ 1 đến 100.
 - Thực hiện 100 lần lệnh xuất!!
 - ➔ Dừng lệnh lặp.

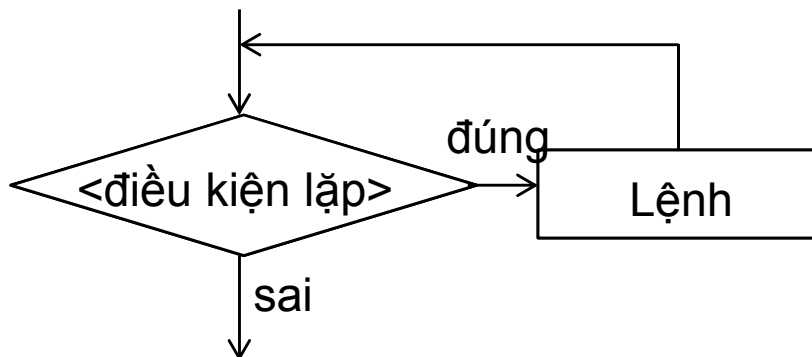


■ Câu lệnh while và do-while:

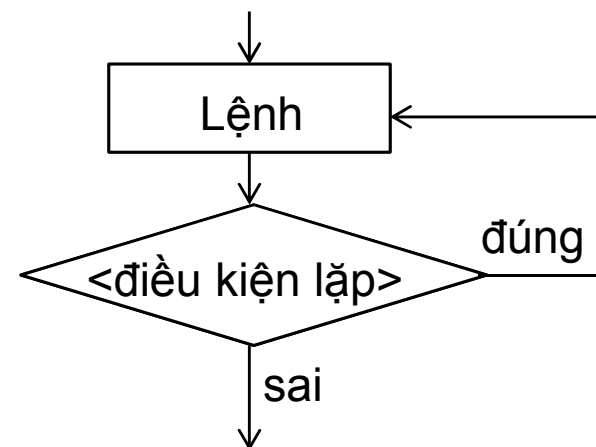
■ Cú pháp:

```
// Câu lệnh while  
while (<điều kiện lặp>)  
    <Lệnh>;
```

```
// Câu lệnh while tương đương  
    <Lệnh>;  
while (<điều kiện lặp>)  
    <Lệnh>;
```



```
// Câu lệnh do-while  
do  
{  
    <Lệnh>;  
} while (<điều kiện lặp>;)
```





■ Câu lệnh while và do-while:

■ Ví dụ:

// Câu lệnh while

```
printf("Nhap vao n = ");  
scanf("%d", &n);
```

```
i = 1;  
while (i <= n)  
{  
    printf("%d", i);  
    i++;  
}
```

// Câu lệnh do-while

```
printf("Nhap vao n = ");  
scanf("%d", &n);
```

```
i = 1;  
do  
{  
    printf("%d", i);  
    i++;  
} while (i <= n);
```



■ Câu lệnh while và do-while:

■ Lưu ý:

- Điều kiện lặp phải đặt giữa ().
- Một lệnh lặp thường có:
 - ➔ B1: Khởi tạo biến đếm.
 - ➔ B2: Kiểm tra điều kiện lặp.
 - ➔ B3: Thực hiện lệnh.
 - ➔ B4: Thay đổi biến đếm.

```
while n > 0    // Sai
{
    ...
}

k = 0;          // B1
while (k < n)   // B2
{
    S = S * k;   // B3
    k = k + 2;   // B4
}
```

Cấu trúc lặp



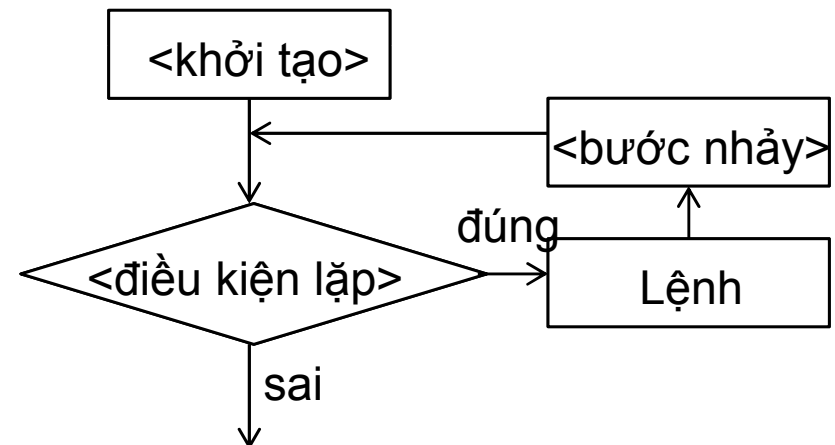
■ Câu lệnh for:

■ Cú pháp:

```
for ([<khởi tạo>] ; [<điều kiện lặp>] ; [<bước nhảy>])  
    <Lệnh>;
```

// Câu lệnh while tương đương

```
<khởi tạo>;  
while (<điều kiện lặp>)  
{  
    <Lệnh>;  
    <bước nhảy>;  
}
```





■ Câu lệnh for:

■ Ví dụ:

```
printf("Nhap vao n = ");  
scanf("%d", &n);
```

// Đầy đủ.

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

```
printf("Nhap vao n = ");  
scanf("%d", &n);
```

// Bỏ khởi tạo.

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

```
printf("Nhap vao n = ");  
scanf("%d", &n);
```

// Bỏ luôn bước nhảy.

```
i = 1;  
for ( ; i <= n; )  
{  
    printf("%d", i);  
    i++;  
}
```



■ Lệnh break và continue:

■ Lệnh break:

- Thoát khỏi vòng lặp.
- Dừng kết hợp với **if-else**.

■ Lệnh continue:

- Bỏ qua một lần lặp.
- Dừng kết hợp với **if-else**.

```
printf("Nhap vao n = ");  
scanf("%d", &n);
```

```
for (i = 1; ; i++)  
{  
    if (i > n)  
        break;  
    if (i % 2 == 0)  
        continue;  
  
    printf("%d", i);  
}
```




■ Biểu thức và toán tử trong C:

- Biểu thức: một dãy hữu hạn toán tử và toán hạng.
- Toán tử số học: $+$, $-$, $*$, $/$, $\%$.
- Toán tử so sánh: $>$, $<$, $>=$, $<=$, $==$, $!=$.
- Toán tử logic: $!$, $\&\&$, $||$.
- Toán tử trên bit: \sim , $\&$, $|$, $^$, $>>$, $<<$.
- Toán tử tăng, giảm: $++$, $--$.
- Toán tử gán: $=$, $<\text{phép toán}>=$.





■ Cấu trúc rẽ nhánh:

- Câu lệnh if-else.
- Câu lệnh switch-case.

■ Cấu trúc lặp:

- Câu lệnh while: kiểm tra điều kiện lặp trước.
- Câu lệnh do-while: kiểm tra điều kiện lặp sau.
- Câu lệnh for:
 - Khởi tạo biến đếm.
 - Kiểm tra điều kiện lặp.
 - Thực hiện lệnh.
 - Thay đổi biến đếm.





■ Bài tập 4.1:

Viết chương trình C mô phỏng máy tính tay như sau:

- Nhập vào 2 số nguyên a, b .
- Nhập vào phép tính $(+, -, *, /, \%)$.
- Thực hiện phép tính vừa nhập trên 2 số nguyên và xuất kết quả.

Định dạng nhập:

Nhap $a, b = 7 \ 5$

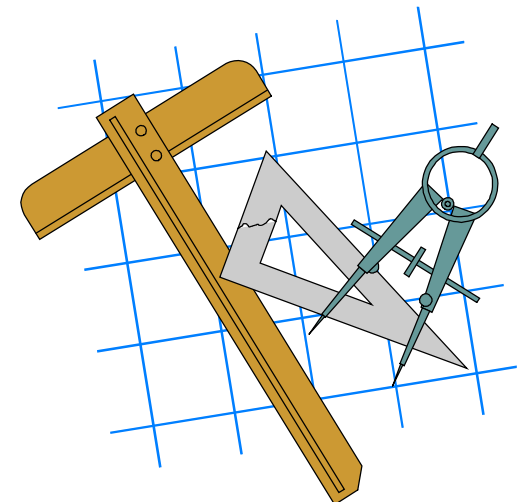
*Nhap phép tính $(+, -, *, /, \%) = +$*

Định dạng xuất:

Ket qua = 12

Định dạng xuất lỗi chia 0:

Loi chia 0!



Bài tập



■ Bài tập 4.2:

Viết chương trình C giải phương trình bậc hai: $ax^2 + bx + c = 0$.

Định dạng nhập:

Nhap he so a, b, c = 2 -5 3

Định dạng xuất:

Nghiem 1 = 1

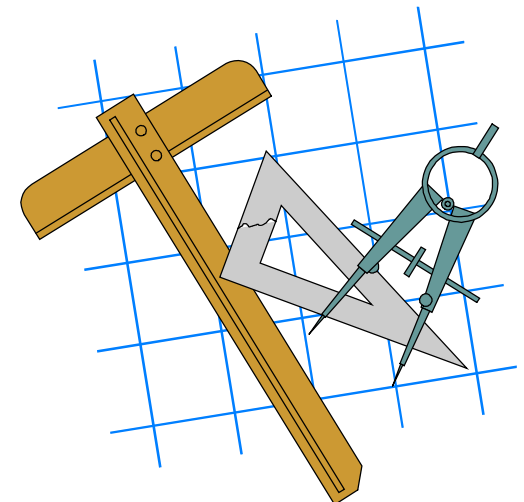
Nghiem 2 = 1.5

Định dạng xuất nghiệm kép:

Nghiem kep = <ngiem kep>

Định dạng xuất vô nghiệm:

Phuong trinh vo nghiem!





■ Bài tập 4.3:

Viết chương trình C tính số ngày trong tháng như sau:

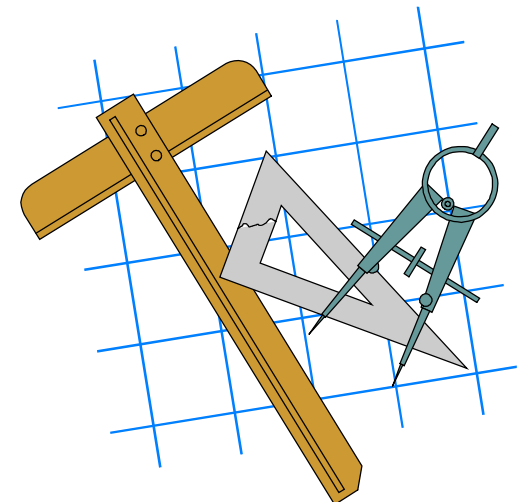
- Nhập vào tháng và năm.
- Tính số ngày trong tháng và xuất kết quả.

Định dạng nhập:

Nhap thang va nam = 6 2012

Định dạng xuất:

*Thang **6** nam **2012** co **30** ngay.*





■ Bài tập 4.4:

Viết chương trình C như sau:

- Nhập vào số nguyên dương N.

- Tính và xuất kết quả:

a) $N! = 1 * 2 * \dots * N.$

b) $\ln(2) = 1 - 1/2 + 1/3 - \dots +/- 1/N.$

c) $PI = 4 (1 - 1/3 + 1/5 - \dots +/- 1/(2*N + 1)).$

d) $S = a_1 + a_2 + \dots a_k (\{ a_i \} \text{ là tập số chính phương } \leq N).$

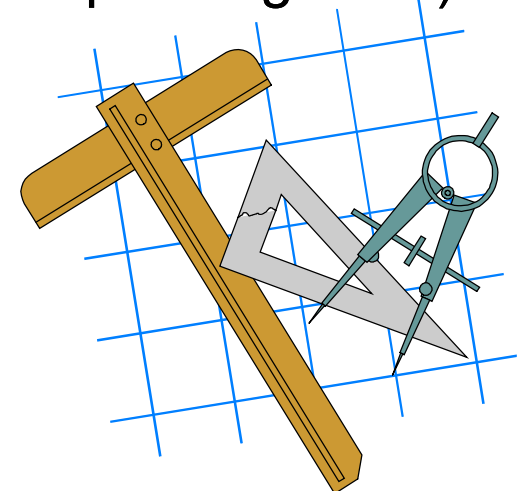
Định dạng xuất:

$N! = \text{<ket qua cau a>}$

$\ln(2) = \text{<ket qua cau b>}$

$PI = \text{<ket qua cau c>}$

$S = \text{<ket qua cau d>}$





■ Bài tập 4.5:

Viết chương trình C tìm và đếm những số có tính chất sau:

- Số nguyên dương có 3 chữ số.
- Chữ số hàng chục = chữ số hàng trăm + chữ số hàng đơn vị.

Định dạng xuất:

1: <số thứ 1 thỏa tính chất>

2: <số thứ 2 thỏa tính chất>

...

N: <số thứ N thỏa tính chất>

Co tat ca N so thoa tinh chat.

