

Chỉ dùng Visual C++6.0 hoặc Dev-C++

Cấu trúc điều khiển

Biểu thức luận lý (điều kiện):

- Là biểu thức cho một trong hai giá trị ĐÚNG hoặc SAI.
- Trong C biểu thức luận lý cho giá trị **1** nếu nó ĐÚNG và cho giá trị **0** nếu SAI.

Các biểu thức so sánh:

- Cho X, Y là hai giá trị số. Các biểu thức so sánh trong VB gồm:

$X < Y$ ($X < Y$),	$X >= Y$ ($X \geq Y$),
$X <= Y$ ($X \leq Y$),	$X == Y$ (so sánh bằng),
$X > Y$ ($X > Y$),	$X != Y$ ($X \neq Y$).

Chú ý: Nếu X và Y là biểu thức thì giá trị của hai biểu thức được dùng để so sánh.

Các phép toán luận lý:

Cho X, Y là hai biểu thức luận lý.

1. Phép toán $\&\&$ (và):

X	Y	$(X) \&\& (Y)$
1	1	1
1	0	0
0	1	0
0	0	0

2. Phép toán \parallel (Hay):

X	Y	$(X) \parallel (Y)$
1	1	1
1	0	1
0	1	1
0	0	0

3. Phép toán Not :

x	!(x)
1	0
0	1

2.5 Giá trị của biểu thức luận lý:

Ví dụ 1 :

```
int x, y, z, E;
```

```
x = 4; y=5 ; z = 1; E = (x > y) || z;
```

Hỏi : Giá trị của E.

Ví dụ 2 :

```
int x, y, z, E;
```

```
x = 4; y=5 ; z = 1; E = (x > y) && z;
```

Hỏi : Giá trị của E.

Ví dụ 3 :

```
int x, y, z, E;
```

```
x = 4; y=5 ; z = 1; E = (!(x > y)) && z;
```

Hỏi : Giá trị của E.

Cấu trúc điều khiển – Rẽ nhánh:

Dạng 1:

```
if (C) {  
    A;  
}
```

- C là một điều kiện (Biểu thức luận lý). A là các thao tác Nhập, Viết, gán, if, . . .

- Nếu *C là đúng* thì *thực hiện A*

Chú ý : Nếu A chỉ là 1 lệnh , có thể viết

if (C) A;

Ví dụ 3 : Viết chương trình giải phương trình bậc nhất

$$ax + b = 0.$$

- $a \neq 0$: phương trình có nghiệm duy nhất $x = -b/a$,
- $a=0$ và $b=0$: phương trình có vô số nghiệm,
- $a=0$ và $b \neq 0$: phương trình vô nghiệm.


```
int main( )
```

```
{
```

```
    double a, b ;
```

```
    1. scanf("%lf %lf", &a, &b);
```

```
    2. if (a != 0) printf("%.1f", -b/a) ;
```

```
    3. if ((a == 0) && (b==0)) printf("Vo so nghiem ") ;
```

```
    4. if ((a == 0) && (b!=0)) printf("Vo nghiem ") ;
```

```
    return 0;
```

```
}
```

Bài tập : Viết chương trình thực hiện : tìm giá trị lớn nhất (max) của hai số thực .



2.6.2 Dạng 2:

```
if (C) {  
    A;  
}  
else {  
    B;  
}
```

**Nếu C đúng thì thực hiện A,
ngược lại (C sai) , thực hiện B.**

Ví dụ 2: Hãy viết chương trình tìm giá trị lớn nhất của 2 số thực a và b.

Nhập a, b;

if (a > b) max = a ;

else max = b ;

Viết max;

Ví dụ 3 : Giải và biện luận $ax + b = 0$, a, b số thực.

Thuật toán:

B1. Nhập hai giá trị gán cho 2 biến a và b

B2. Nếu $a \neq 0$ là đúng thì viết ra màn hình $-b / a$

Ngược lại ($a = 0$) , **xét b ,**



Xét b ($a=0$):

Nếu $b = 0$ là đúng thì viết ra màn hình “Vo so nghiem”

Ngược lại ($b \neq 0$) , viết ra màn hình “Vo nghiem”

Giải Ví dụ 3 :

Nhập a, b ;

if (a != 0) {

printf("x = %lf \n", -b/a);

}

else {

Xét b (Bài tập tại lớp)

}

Bài giải ví dụ 3:

```
if (a != 0)
```

```
    printf("x = %lf \n", -b/a);
```

```
else {
```

```
    if (b == 0) printf(" Vo so nghiem \n");
```

```
    else  printf(" Vo nghiem \n");
```

```
}
```

Chú ý :

```
int x=0;
```

```
if (x == 0) printf("Dung") ;
```

```
else printf("Sai") ;
```

➤ Trên màn hình : **Dung**

```
int x=0;
```

```
if (x = 0) printf("Dung") ;
```

```
else printf("Sai") ;
```

➤ Trên màn hình : **Sai**

Cấu trúc điều khiển – Chọn lựa:

```
switch (biểu_thức) {  
  case hằng_1 :  
    Lệnh_1;  
    break; /* optional */  
  case hằng_2 :  
    Lệnh_2;  
    break; /* optional */  
  default : /* optional */  
    Lệnh_default;  
}  
Lệnh ;
```

- **biểu thức** có giá trị là số nguyên,
- **hằng_1, hằng_2** là các hằng nguyên.

- Nếu **biểu_thức** = **hằng_i** thì thực hiện các lệnh trong **case hằng_i** cho đến khi gặp **break** của **case hằng_i**,
- Nếu trong **case hằng_i** không có **break** thì thực hiện các lệnh trong các **case hằng** *bên dưới* cho đến khi gặp **break**.

Ví dụ 1:

```
scanf("%d", &x);
```

```
switch(x) {
```

```
    case 0: printf("x = 0"); break;
```

```
    case 1: printf("x = 1"); break;
```

```
    case 2: printf("x = 2"); break;
```

```
    //-----
```

```
    case 3: printf("x = 3"); break;
```

```
    case 4: printf("x = 4"); break;
```

```
    case 5: printf("x = 5"); break;
```

```
    default : printf("x < 0 or x > 5.\n");
```

```
}
```

Ví dụ 2:

```
scanf("%d", &x);
switch(x) {
    case 0: printf("x belongs to {0,1,2}.\n"); break;
    case 1: printf("x belongs to {0,1,2}.\n"); break;
    case 2: printf("x belongs to {0,1,2}.\n"); break;
    //-----
    case 3: printf("x belongs to {3,4,5}.\n"); break;
    case 4: printf("x belongs to {3,4,5}.\n"); break;
    case 5: printf("x belongs to {3,4,5}.\n"); break;
    default : printf("x < 0 or x > 5.\n");
}
```

Ví dụ 3:

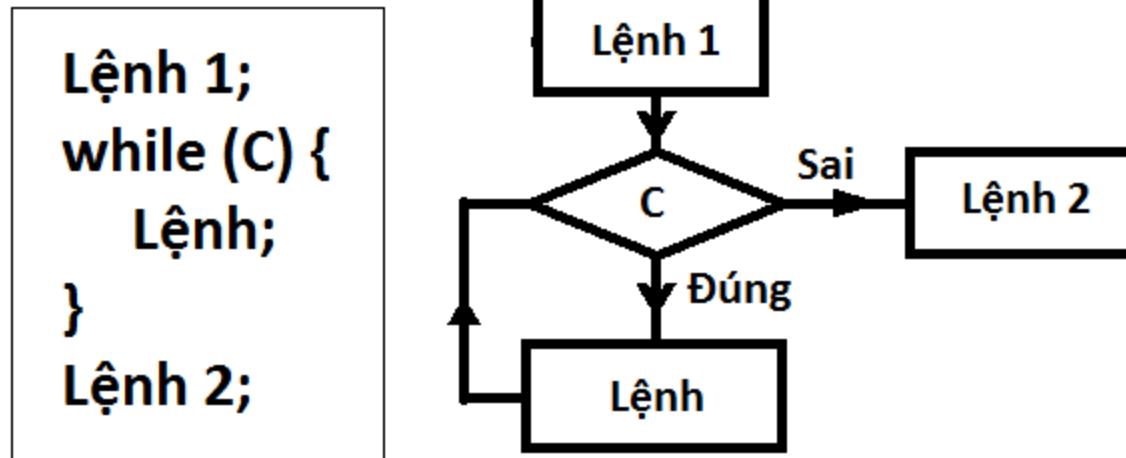
```
scanf("%d", &x);  
switch(x) {  
    case 0:  
    case 1:  
    case 2:  
        printf("x belongs to {0,1,2}.\n");  
        break;  
    case 3:  
    case 4:  
    case 5:  
        printf("x belongs to {3,4,5}.\n");  
        break;  
    default :  
        printf("x < 0 or x > 5.\n");  
}
```

Ví dụ 3 (viết cách khác) :

```
switch (x){  
    case 1:case 2: case 3:  
        printf("x belongs to {1,2,3}.\n"); break;  
    case 4:case 5: case 6:  
        printf("x belongs to {4,5,6}.\n");  
}
```

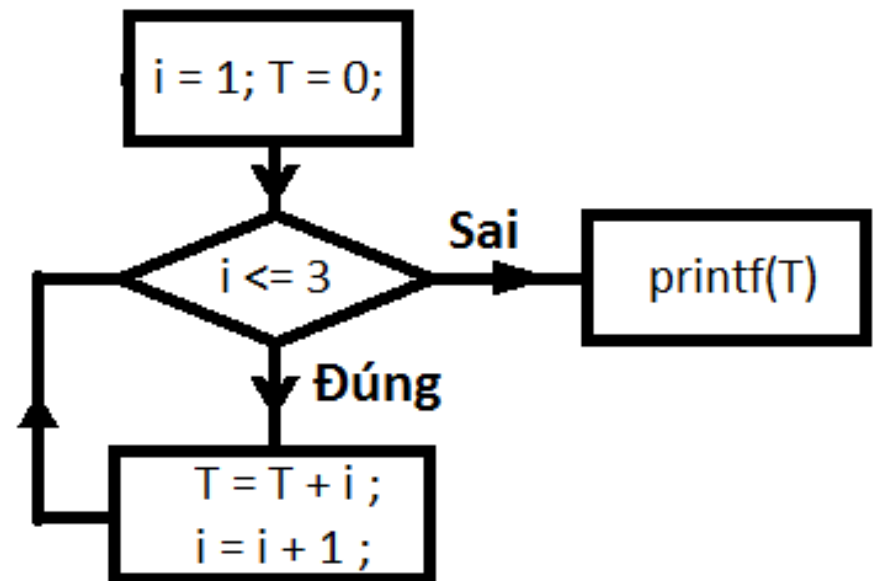
Cấu trúc điều khiển – Lặp:

Dạng 1: Gọi C là điều kiện.



Ví dụ 1 : Cho biết giá trị được viết ra màn hình của thuật toán sau. Viết chương trình theo thuật toán.

```
B1.  $i = 1 ; T = 0 ;$   
B2. while ( $i \leq 3$ ) {  
    B3.  $T = T + i ;$   
    B4.  $i = i + 1 ;$   
}  
B5.  $\text{printf}("%d", T);$ 
```



Ví dụ 1 :

Thuật toán :

B1. $i = 1, T = 0$

B2. Nếu $i \leq 3$ (là đúng) thì thực hiện

B21. $T = T + i$

B22. $i = i + 1$

B23. Quay lại B2.

B3. Viết T ra màn hình

B1. $i = 1 ; T = 0;$

B2. while ($i \leq 3$) {
 B3. $T = T + i ;$
 B4. $i = i + 1;$
}

B5. `printf("%d", T);`

Ví dụ 2 : (BT tại lớp)

Thuật toán :

B1. $i = 4, T = 0$

B2. Nếu $i > 0$ thì thực hiện

B21. $T = T + i$

B22. $i = i - 1$

B23. Quay lại B2.

B3. Viết T ra màn hình

Các bài toán :

Bài toán 1: Viết chương trình tính tổng , số nguyên.

$T = 1 + 2 + \dots + n$, $n \geq 0$. Nếu $n = 0$ thì $T = 0$.

Bước 1 : Ta viết cách tính tổng T dưới dạng *cộng dồn*:

B1. Nhập n

B2. $T = 0$

B3. Thực hiện các bước (cộng dồn vào T)

1. $T = T + 1$

2. $T = T + 2$

...

n. $T = T + n$

B4. Viết T ra màn hình.

Các bài toán :

Bài toán 1: Viết chương trình tính tổng , số nguyên.

$T = 1 + 2 + \dots + n$, $n \geq 0$. Nếu $n = 0$ thì $T = 0$.

Bước 1 : Ta viết cách tính tổng T dưới dạng *cộng dồn*:

B1. Nhập n

B2. $T = 0$

B3. Thực hiện các bước (cộng dồn vào T)

1. $T = T + 1$

2. $T = T + 2$

...

n. $T = T + n$

} Vòng lặp

B4. Viết T ra màn hình.

Chương trình C :

```
int n, T, i ;  
scanf("%d ", &n) ;  
T = 0 ; i = 1 ;  
while (i <= n) {  
    T = T + i ;  
    i = i + 1 ;  
}  
printf ("%d", T);
```

Bài toán 2 : Viết chương trình tính tổng

$$T = \frac{1}{1} + \frac{1}{1+2} + \frac{1}{1+2+3} + \dots + \frac{1}{1+\dots+n}, n \geq 0.$$

Nếu $n = 0$ thì $T = 0$.

Bước 1a: Ta viết cách tính tổng T dưới dạng *cộng dồn*:

B1. Nhập n (giả sử giá trị nhập là 3)

B2. $T = 0$

B3. 1. $T = T + \frac{1}{1}$

 2. $T = T + \frac{1}{1+2}$

 3. $T = T + \frac{1}{1+2+3}$

B4. Viết T ra màn hình.

Bước 1b : Ta viết cách tính tổng T dưới dạng *cộng dồn*:

B1. $S = 0, T = 0$

B2. 1. $S = S + 1, \quad T = T + \frac{1}{S}$

 2. $S = S + 2, \quad T = T + \frac{1}{S}$

 3. $S = S + 3, \quad T = T + \frac{1}{S}$

B3. Viết T ra màn hình.

Bước 2 : Tổng quát cho n.

B1. Nhập n

B2. $T = 0, S = 0, i = 1$

B3. Nếu $i \leq n$ thì thực hiện

B21. $S = S + i, T = T + \frac{1}{S}, i = i + 1$

B22. Quay lại B3.

B4. Viết T ra màn hình.



Bài toán 3 : Viết chương trình kiểm tra n nguyên dương có là số nguyên tố không.

Ta có :

- 1) $n = 1$: không là số nguyên tố.
- 2) $n = 2$: là số nguyên tố.
- 3) $n > 2$: là số nguyên tố nếu n chỉ chia hết cho 1 và chính nó

Bài giải :

B1. Nhập n

B2. Nếu $n = 1$ thì viết ra màn hình “ n không là NT”

B3. Nếu $n = 2$ thì viết ra màn hình “ n là NT”

B4. Nếu $n > 2$ thì

B41. Đếm số lần n chia hết cho i , $i = 1, \dots, n$, gán kết quả vào d

B42. Nếu $d=2$ thì Viết “ n là NT”

Ngược lại Viết “ n không là NT”

Bài toán 4 : Viết chương trình tính tổng các chữ số của một số nguyên dương.

B1. Nhập n , $T = 0$

B2. Nếu $n \neq 0$ thì

$T = T + \text{dư của } n \text{ chia nguyên cho } 10$

Thay n bởi n chia nguyên cho 10

Quay lại B2.

B3. Viết T

Chương trình :

Nhập n

```
while (n != 0) {  
    du = n % 10;  
    T = T + du;  
    n = n / 10;  
}
```

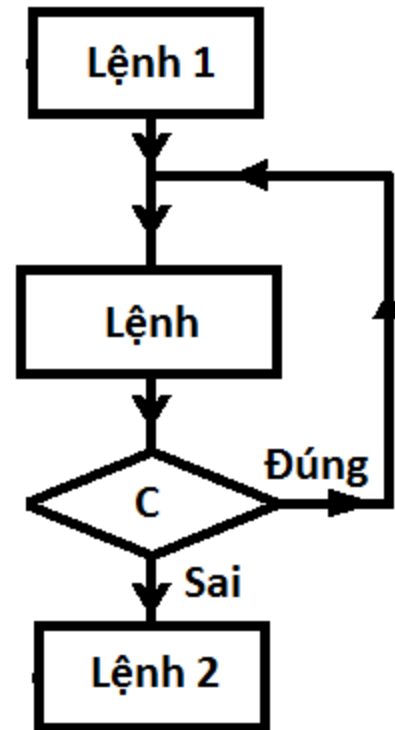
Viết T

Thực hiện CT (từng lệnh) với $n = 15$.

Cấu trúc điều khiển – Lặp:

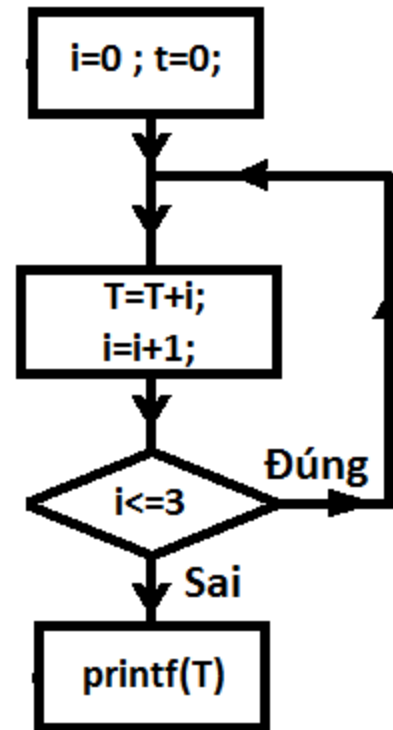
Dạng 2:

```
Lệnh 1;  
Do {  
    Lệnh ;  
} while (C) ;  
Lệnh 2;
```



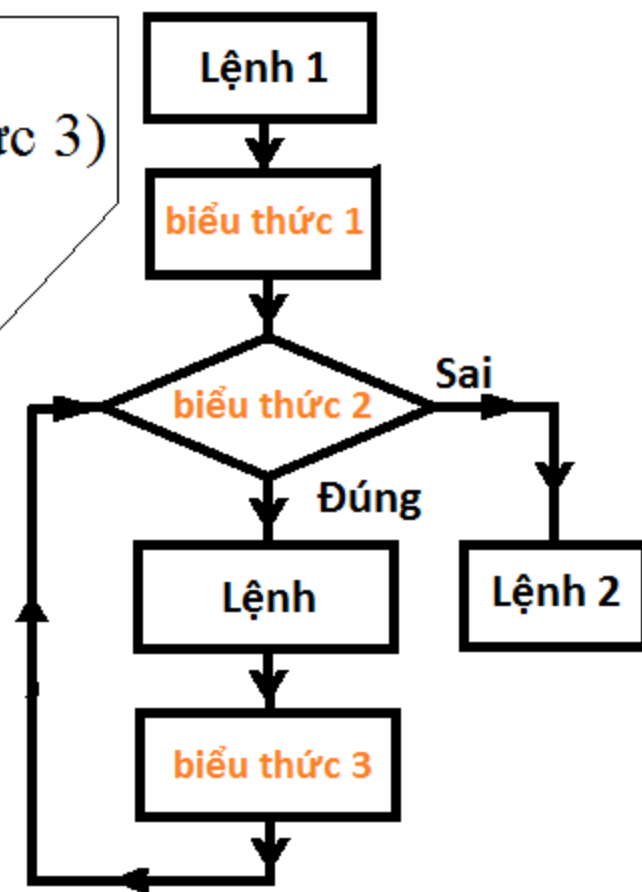
Ví dụ :

```
int i = 0 , T = 0 ;  
do {  
    T=T+i;  
    i=i+1;  
}while (i<=3);  
printf("%d\n", T);
```



Dạng 3:

```
Lệnh 1 ;  
for (biểu thức 1 ; biểu thức 2 ; biểu thức 3)  
{  
    Lệnh ;  
}  
Lệnh 2 ;
```



Ví dụ 1 :

```
int i , T;  
T = 0;  
for (i=1; i<=3; i=i+1)  
{  
    T=T+i;  
}  
printf("%d\n", T);
```

Ví dụ 2 :

```
int i, T ;  
i = 1 , T = 0;  
for (   ; i<=3; i=i+1) T=T+i;  
printf("%d\n", T);
```

Ví dụ 3 :

```
int i = 0 , T = 0;  
for ( ; i<=3; )  
{  
    T = T+i;  
    i = i+1;  
}  
printf("%d\n", T);
```


Bài tập : Dùng vòng lặp for.

B1. $T = 0$

B2. $i = 3$

B3. Nếu $i \leq 8$ thì thực hiện

$$T = T + i$$

$$i = i + 2$$

Quay lại B3.

B4. Viết T ra màn hình.

Lệnh break:

Dùng để kết thúc các vòng lặp gần nó nhất.

```
while (C1) {
```

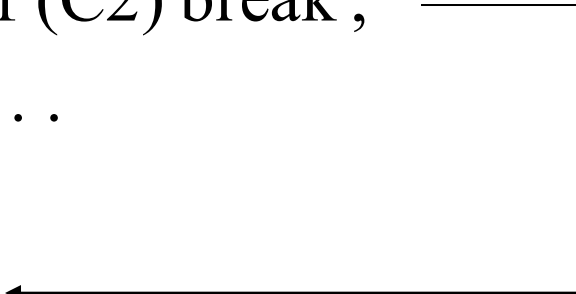
```
    . . .
```

```
    if (C2) break ;
```

```
    . . .
```

```
}
```

```
Lệnh ;
```



2.9 Lệnh break:

Dùng để kết thúc các vòng lặp gần nó nhất.

```
while (C1) {
```

```
    ...
```

```
        while (C2) {
```

```
            ...
```

```
            if (C3) break ;
```

```
            ...
```

```
        }
```

```
        Lệnh ;
```

```
    }
```

```
Lệnh ;
```

