Chương 4

CÁC LỆNH CÓ CẤU TRÚC

Học xong chương này, sinh viên sẽ nắm được các vấn đề sau:

- *Khối lệnh trong C.*
- *Cấu trúc rẽ nhánh.*
- *Cấu trúc lựa chọn.*
- *Cấu trúc vòng lặp.*
- *Các câu lệnh "đặc biệt".*

I. KHỐI LỆNH

Một dãy các khai báo cùng với các câu lệnh nằm trong cặp dấu ngoặc móc { và } được gọi là một khối lệnh.

Ví dụ 1:

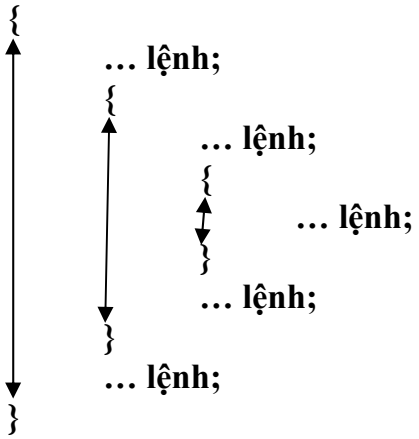
```
{
     char  ten[30];
     printf("\n Nhap vao ten cua ban:");
     scanf("%s", ten);
     printf("\n Chao Ban  %s",ten);
}
```

Ví dụ 2:

```
#include <stdio.h>
#include<conio.h>
int main ()
{  /*đây là đầu khối*/
     char ten[50];
     printf("Xin cho biet ten cua ban !");
     scanf("%s",ten);
     getch();
     return 0;
} /*đây là cuối khối*/
```

Một khối lệnh có thể chứa bên trong nó nhiều khối lệnh khác gọi là khối lệnh lồng nhau. Sự lồng nhau của các khối lệnh là không hạn chế.

Minh họa:



Lưu ý về phạm vi tác động của biến trong khối lệnh lồng nhau:

- Trong các khối lệnh khác nhau hay các khối lệnh lồng nhau có thể khai báo các biến cùng tên.

Ví dụ 1:

```
{
        … lệnh;
        {
                int a,b;        /*biến a, b trong khối lệnh thứ nhất*/
                … lệnh;
        }
        …lệnh;
        {
                int a,b;        /*biến a,b trong khối lệnh thứ hai*/
                … lệnh;
        }
}
```

Ví dụ 2:

```
{
        int a, b;               /*biến a,b trong khối lệnh "bên ngoài"*/
        … lệnh;
        {
                int a,b;        /*biến a,b bên trong khối lệnh con*/
        }
}
```

- Nếu một biến được khai báo bên ngoài khối lệnh và không trùng tên với biến bên trong khối lệnh thì nó cũng được sử dụng bên trong khối lệnh.

- Một khối lệnh con có thể sử dụng các biến bên ngoài, các lệnh bên ngoài không thể sử dụng các biến bên trong khối lệnh con.

Ví dụ:

```
{
        int    a, b, c;
        …lệnh;
        {
                int  c, d;
                …lệnh;
        }
}
```

II. CẤU TRÚC RẼ NHÁNH

Cấu trúc rẽ nhánh là một cấu trúc được dùng rất phổ biến trong các ngôn ngữ lập trình nói chung. Trong C, có hai dạng: dạng không đầy đủ và dạng đầy đủ.

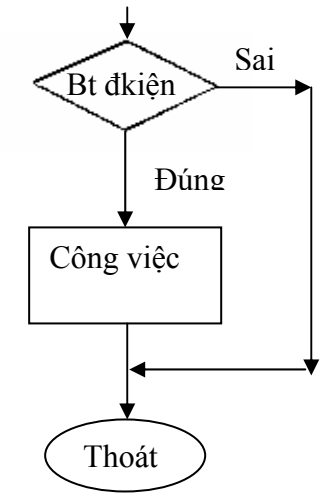
II.1. Dạng không đầy đủ

Cú pháp:

if (<*Biểu thức điều kiện***>)**

<Công việc>

Lưu đồ cú pháp:



Giải thích:

<Công việc> được thể hiện bằng 1 câu lệnh hay 1 khối lệnh.

Kiểm tra *Biểu thức điều kiện* trước.

Nếu điều kiện đúng (!= 0) thì thực hiện câu lệnh hoặc khối lệnh liền sau điều kiện.

Nếu điều kiện sai thì bỏ qua lệnh hoặc khối lệnh liền sau điều kiện (những lệnh và khối lệnh sau đó vẫn được thực hiện bình thường vì nó không phụ thuộc vào điều kiện sau if).

Ví dụ 1:

Yêu cầu người thực hiện chương trình nhập vào một số thực a. In ra màn hình kết quả nghịch đảo của a khi a $\neq 0$.

```
#include <stdio.h>
#include <conio.h>
int main ()
{
   float a;
   printf("Nhap a = "); scanf("%f",&a);
   if (a !=0 )
         printf("Nghich dao cua %f la %f",a,1/a);
   getch();
   return 0;
}
```

Giải thích:

- Nếu chúng ta nhập vào a $\neq 0$ thì câu lệnh `printf("Nghich dao cua %f la %f",a,1/a)` được thực hiện, ngược lại câu lệnh này không được thực hiện.

- Lệnh getch() luôn luôn được thực hiện vì nó không phải là "lệnh liền sau" điều kiện if.

Ví dụ 2: Yêu cầu người chạy chương trình nhập vào giá trị của 2 số a và b, nếu a lớn hơn b thì in ra thông báo "Gia trị của a lớn hơn giá trị của b", sau đó hiển thị giá trị cụ thể của 2 số lên màn hình.

```
#include <stdio.h>
#include<conio.h>
int main ()
{
   int a,b;
   printf("Nhap vao gia tri cua 2 so a, b!");
   scanf("%d%d",&a,&b);
   if (a>b)
```

```
    {
          printf("\n Gia tri cua a lon hon gia tri cua b");
          printf("\n a=%d,  b=%d",a,b);
    }
    getch();
    return 0;
}
```

Giải thích:

Nếu chúng ta nhập vào giá trị của a lớn hơn giá trị của b thì khối lệnh:

```
{
      printf("\n Gia tri cua a lon hon gia tri cua b");
      printf("\n a=%d,  b=%d",a,b);
}
```

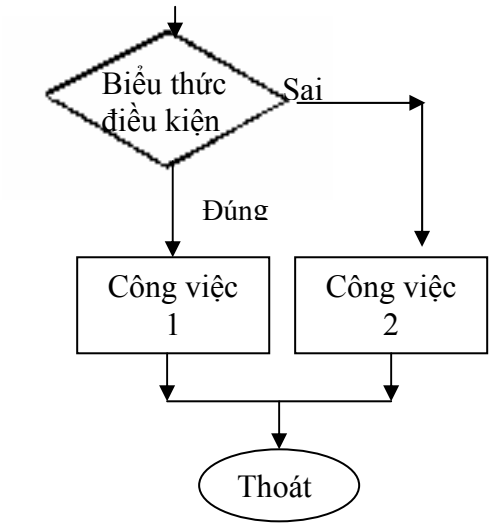
sẽ được thực hiện, ngược lại khối lệnh này không được thực hiện.

II.2. Dạng đầy đủ

Cú pháp:

if (<*Biểu thức điều kiện***>)**
    <Công việc 1>
else
    <Công việc 2>

Lưu đồ cú pháp:



Giải thích:
Công việc 1, công việc 2 được thể hiện là 1 câu lệnh hay 1 khối lệnh.
Đầu tiên *Biểu thức điều kiện* được kiểm tra trước.
Nếu điều kiện đúng thì thực hiện công việc 1.
Nếu điều kiện sai thì thực hiện công việc 2.
Các lệnh phía sau công việc 2 không phụ thuộc vào điều kiện.

Ví dụ 1: Yêu cầu người thực hiện chương trình nhập vào một số thực a. In ra màn hình kết quả nghịch đảo của a khi a $\neq 0$, khi a =0 in ra thông báo "Khong the tim duoc nghich dao cua a"

```
#include <stdio.h>
#include <conio.h>
int main ()
{
   float a;
   printf("Nhap a = "); scanf("%f",&a);
   if (a !=0 )
        printf("Nghich dao cua %f la %f",a,1/a);
   else
        printf("Khong the tim duoc nghich dao cua a");
```

```
    getch();
    return 0;
}
```

Giải thích:

- Nếu chúng ta nhập vào a ≠ 0 thì câu lệnh `printf("Nghich dao cua %f la %f",a,1/a)` được thực hiện, ngược lại câu lệnh `printf("Khong the tim duoc nghich dao cua a")` được thực hiện.

- Lệnh getch() luôn luôn được thực hiện.

Ví dụ 2: Yêu cầu người chạy chương trình nhập vào giá trị của 2 số a và b, nếu a lớn hơn b thì in ra thông báo "Gia trị của a lớn hơn giá trị của b, giá trị của 2 số", ngược lại thì in ra màn hình câu thông báo "Giá trị của a nhỏ hơn hoặc bằng giá trị của b, giá trị của 2 số".

```
#include <stdio.h>
#include<conio.h>
int main ()
{
   int a, b;
   printf("Nhap vao gia tri cua 2 so a va b !");
   scanf("%d%d",&a,&b);
   if (a>b)
        {
             printf("\n a lon hon b");
             printf("\n a=%d  b=%d ",a,b);
        }
   else
        {
             printf("\n a nho hon hoac bang b");
             printf("\n a=%d  b=%d",a,b);
        }
   printf("\n Thuc hien xong lenh if");
   getch();
   return 0;
}
```

Giải thích:

- Nếu chúng ta nhập vào 40 30 ↵ thì kết quả hiển ra trên màn hình là

  a lon hon b
  a=40 b=30
  Thuc hien xong lenh if

- Còn nếu chúng ta nhập 40 50 ↵ thì kết quả hiển ra trên màn hình là

  a nho hon hoac bang b
  a=40 b=50
  Thuc hien xong lenh if

Ví dụ 3: Yêu cầu người thực hiện chương trình nhập vào một số nguyên dương là tháng trong năm và in ra số ngày của tháng đó.

- Tháng có 31 ngày: 1, 3, 5, 7, 8, 10, 12
- Tháng có 30 ngày: 4, 6, 9, 10
- Tháng có 28 hoặc 29 ngày : 2

```
#include <stdio.h>
#include<conio.h>
```

```
int main ()
{
      int thg;
      printf("Nhap vao thang trong nam !");
      scanf("%d",&thg);
      if (thg==1||thg==3||thg==5||thg==7||thg==8||thg==10||thg==12)
            printf("\n Thang %d co 31 ngay ",thg);
      else if (thg==4||thg==6||thg==9||thg==11)
            printf("\n Thang %d co 30 ngay",thg);
            else if (thg==2)
                  printf("\n Thang %d co 28 hoac 29 ngay",thg);
                 else printf("Khong co thang %d",thg);
      printf("\n Thuc hien xong lenh if");
      getch();
      return  0;
}
```

Giải thích:

- Nếu chúng ta nhập vào một trong các số 1, 3, 5, 7, 8, 10, 12 thì kết quả xuất hiện trên màn hình sẽ là

Thang <số> co 31 ngay
Thuc hien xong lenh if

- Nếu chúng ta nhập vào một trong các số 4, 6, 9, 11 thì kết quả xuất hiện trên màn hình sẽ là

Thang <số> co 30 ngay
Thuc hien xong lenh if

- Nếu chúng ta nhập vào số 2 thì kết quả xuất hiện trên màn hình sẽ là

Thang 2 co 28 hoac 29 ngay
Thuc hien xong lenh if

- Nếu chúng ta nhập vào số nhỏ hơn 0 hoặc lớn hơn 12 thì kết quả xuất hiện trên màn hình sẽ là

Khong co thang <số>
Thuc hien xong lenh if

Trong đó <số> là con số mà chúng ta đã nhập vào.

Lưu ý:

- Ta có thể sử dụng các câu lệnh if…else lồng nhau. Trong trường hợp if…else lồng nhau thì *else sẽ kết hợp với if gần nhất chưa có else*.

- Trong trường hợp câu lệnh if "bên trong" không có else thì phải viết nó trong cặp dấu {} (coi như là khối lệnh) để tránh sự kết hợp else if sai.

Ví dụ 1:

```
if ( so1>0)
      if (so2 > so3)
            a=so2;
      else   /*else của if (so2>so3) */
            a=so3;
```

Ví dụ 2:

```
if (so1>0)
{
      if (so2>so3)   /*lệnh if này không có else*/
      a=so2;
}
```

```
        else  /*else của if (so1>0)*/
              a=so3;
```

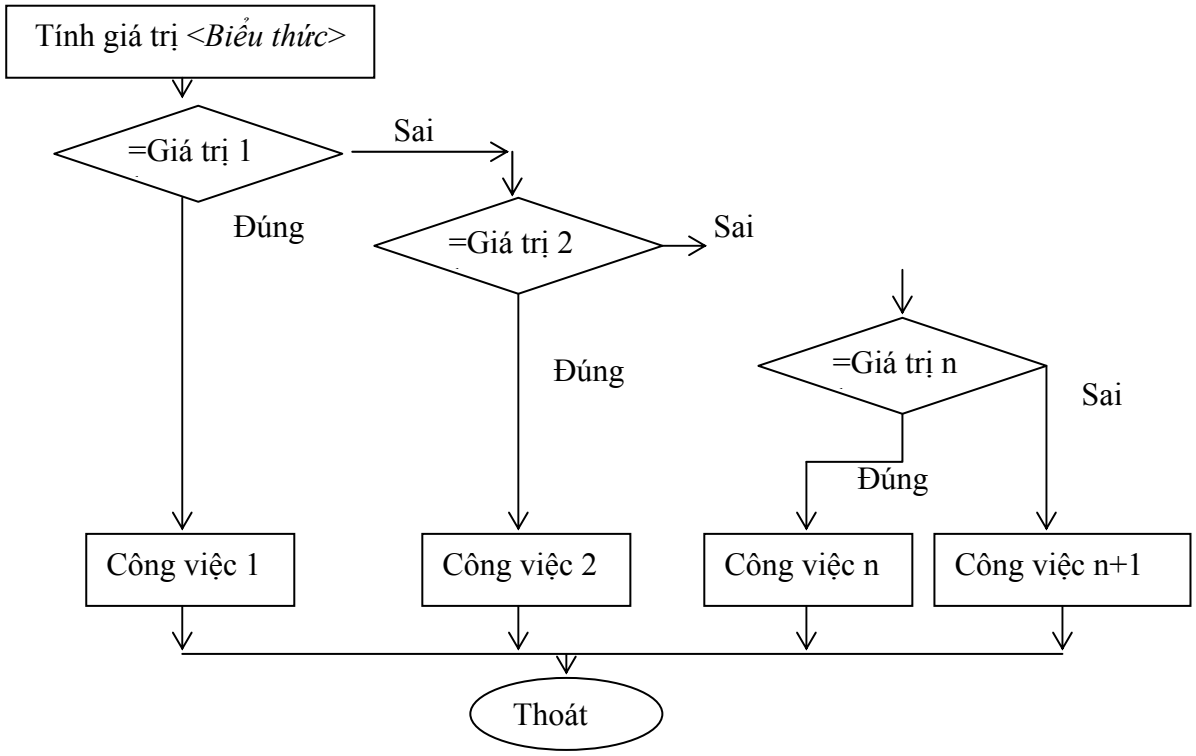
III CẤU TRÚC LỰA CHỌN

Cấu trúc lựa chọn cho phép lựa chọn một trong nhiều trường hợp. Trong C, đó là câu lệnh switch.

Cú pháp:

switch (<*Biểu thức*>)
{
case giá trị 1:
Khối lệnh thực hiện công việc 1;
break;
…
case giá trị n:
Khối lệnh thực hiện công việc n;
break;
[default :
Khối lệnh thực hiện công việc mặc định;
break;]
}

Lưu đồ:



Giải thích:

- Tính giá trị của biểu thức trước.
- Nếu giá trị của biểu thức bằng giá trị 1 thì thực hiện công việc 1 rồi thoát.

- Nếu giá trị của biểu thức khác giá trị 1 thì so sánh với giá trị 2, nếu bằng giá trị 2 thì thực hiện công việc 2 rồi thoát.

- Cứ như thế, so sánh tới giá trị n.

- Nếu tất cả các phép so sánh trên đều sai thì thực hiện công việc mặc định của trường hợp *default*.

Lưu ý:

- Biểu thức trong switch() phải có kết quả là giá trị kiểu số nguyên (int, char, long, …).

- Các giá trị sau case cũng phải là kiểu số nguyên.

- Không bắt buộc phải có default.

Ví dụ 1: Nhập vào một số nguyên, chia số nguyên này cho 2 lấy phần dư. Kiểm tra nếu phần dư bằng 0 thì in ra thông báo "số chẵn", nếu số dư bằng 1 thì in thông báo "số lẻ".

```
#include <stdio.h>
#include<conio.h>
int main ()
{   int songuyen, phandu;
   clrscr();
   printf("\n Nhap vao so nguyen ");
   scanf("%d",&songuyen);
   phandu=(songuyen % 2);
   switch(phandu)
   {
     case 0: printf("%d la so chan ",songuyen);
          break;
     case 1: printf("%d la so le ",songuyen);
          break;
   }
   getch();
   return 0;
}
```

Ví dụ 2: Nhập vào 2 số nguyên và 1 phép toán.

- Nếu phép toán là '+', '-', '*' thì in ra kết qua là tổng, hiệu, tích của 2 số.

- Nếu phép toán là '/' thì kiểm tra xem số thứ 2 có khác không hay không? Nếu khác không thì in ra thương của chúng, ngược lại thì in ra thông báo "khong chia cho 0".

```
#include <stdio.h>
#include<conio.h>
int main ()
{   int so1, so2;
    float  thuong;
    char pheptoan;
     clrscr();
     printf("\n Nhap vao 2 so nguyen ");
     scanf("%d%d",&so1,&so2);
     fflush(stdin);
     /*Xóa ký tự enter trong vùng đệm trước khi nhập phép toán */
     printf("\n Nhap vao phep toan ");
     scanf("%c",&pheptoan);
     switch(pheptoan)
     {
```

```
        case '+':
             printf("\n %d + %d =%d",so1, so2, so1+so2);
             break;
        case '-':
             printf("\n %d - %d =%d",so1, so2, so1-so2);
             break;
        case '*':
             printf("\n %d * %d =%d",so1, so2, so1*so2);
             break;
        case '/':
             if (so2!=0)
             {    thuong=float(so1)/float(so2);
                  printf("\n %d / %d =%f", so1, so2, thuong);
             }
             else printf("Khong chia duoc cho 0");
             break;
        default :
             printf("\n  Chua  ho  tro  phep  toan  %c",  pheptoan);
             break;
      }
      getch();
      return 0;
}
```

Trong ví dụ trên, tại sao phải xóa ký tự trong vùng đệm trước khi nhập phép toán?

Ví dụ 3: Yêu cầu người thực hiện chương trình nhập vào một số nguyên dương là tháng trong năm và in ra số ngày của tháng đó.

- Tháng có 31 ngày: 1, 3, 5, 7, 8, 10, 12
- Tháng có 30 ngày: 4, 6, 9, 10
- Tháng có 28 hoặc 29 ngày : 2
- Nếu nhập vào số <1 hoặc >12 thì in ra câu thông báo “không có tháng này “.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   int thang;
   clrscr();
   printf("\n Nhap vao thangs trong nam ");
   scanf("%d",&thang);
   switch(thang)
   {
     case 1:
     case 3:
     case 5:
     case 7:
     case 8:
     case 10:
     case 12:
          printf("\n Thang %d co 31 ngay ",thang);
          break;
     case 4:
     case 6:
     case 9:
     case 11:
          printf("\n Thang %d co 30 ngay ",thang);
          break;
```

```
        case 2:
              printf ("\ Thang 2 co 28 hoac 29 ngay");
              break;
        default :
              printf("\n Khong co thang %d", thang);
              break;
    }
    getch();
    return 0;
}
```

Trong ví dụ trên, tại sao phải sử dụng case 1:, case 3:, …case 12: ?

IV. CẤU TRÚC VÒNG LẶP

Cấu trúc vòng lặp cho phép lặp lại nhiều lần 1 công việc (được thể hiện bằng 1 câu lệnh hay 1 khối lệnh) nào đó cho đến khi thỏa mãn 1 điều kiện cụ thể.

IV.1. Vòng lặp for

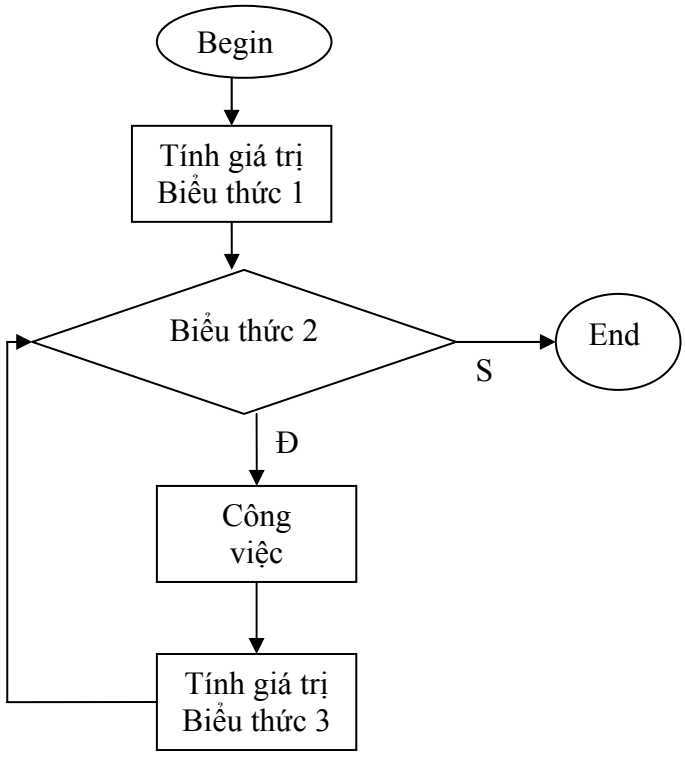
Lệnh for cho phép lặp lại công việc cho đến khi điều kiện sai.

Cú pháp:

for (Biểu thức 1; biểu thức 2; biểu thức 3)

<Công việc>

Lưu đồ:



Giải thích:

<*Công việc*>: được thể hiện là 1 câu lệnh hay 1 khối lệnh. Thứ tự thực hiện của câu lệnh for như sau:

B1: Tính giá trị của biểu thức 1.

B2: Tính giá trị của biểu thức 2.

- Nếu giá trị của biểu thức 2 là sai (=0): *thoát khỏi câu lệnh for*.

- Nếu giá trị của biểu thức 2 là đúng (!=0): <Công việc> được thực hiện.

B3: Tính giá trị của biểu thức 3 và quay lại B2.

Một số lưu ý khi sử dụng câu lệnh for:

- Khi biểu thức 2 vắng mặt thì nó được coi là luôn luôn đúng

- Biểu thức 1: thông thường là một phép gán để khởi tạo giá trị ban đầu cho biến điều kiện.

- Biểu thức 2: là một biểu thức kiểm tra điều kiện đúng sai để dừng vòng lặp.

- Biểu thức 3: thông thường là một phép gán để thay đổi giá trị của biến điều kiện.

- Trong mỗi biểu thức có thể có nhiều biểu thức con. Các biểu thức con được phân biệt bởi dấu phẩy.

Ví dụ 1: Viết đoạn chương trình in dãy số nguyên từ 1 đến 10.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   int i;
   clrscr();
   printf("\n Day so tu 1 den 10 :");
   for (i=1; i<=10; i++)
        printf("%d ",i);
   getch();
   return 0;
}
```

Kết quả chương trình như sau:

```
Day so tu 1 den 10 :1 2 3 4 5 6 7 8 9 10
```

Ví dụ 2: Viết chương trình nhập vào một số nguyên n. Tính tổng của các số nguyên từ 1 đến n.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   unsigned int n,i,tong;
   clrscr();
   printf("\n Nhap vao so nguyen duong n:"); scanf("%d",&n);
   tong=0;
   for (i=1; i<=n; i++)
        tong+=i;
   printf("\n Tong tu 1 den %d =%d ",n,tong);
   getch();
   return 0;
}
```

Nếu chúng ta nhập vào số 9 thì kết quả như sau:

```
Nhap vao so nguyen duong n:9
Tong tu 1 den 9 =45 _
```

Ví dụ 3: Viết chương trình in ra trên màn hình một ma trận có n dòng m cột như sau:

```
1    2    3    4    5    6    7
2    3    4    5    6    7    8
3    4    5    6    7    8    9
…
```

```
#include <stdio.h>
#include<conio.h>
int main ()
{   unsigned int dong, cot, n, m;
   clrscr();
   printf("\n Nhap vao so dong va so cot :");
   scanf("%d%d",&n,&m);
   for (dong=0;dong<n;dong++)
   {
      printf("\n");
      for (cot=1;cot<=m;cot++)
        printf("%d\t",dong+cot);

   }
   getch();
   return 0;
}
```

Kết quả khi nhập 3 dòng 6 cột như sau

```
Nhap vao so dong va so cot :3 6

1       2       3       4       5       6
2       3       4       5       6       7
3       4       5       6       7       8
```

III.2. Vòng lặp while

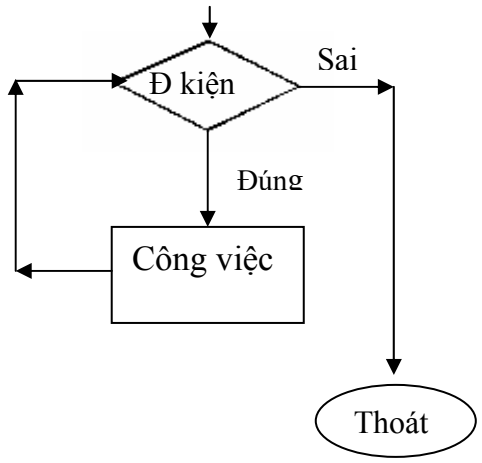
Vòng lặp while giống như vòng lặp for, dùng để lặp lại một công việc nào đó cho đến khi điều kiện sai.

Cú pháp:

while (*Biểu thức điều kiện***)**

<Công việc>

Lưu đồ:



Giải thích:

- <Công việc>: được thể hiện bằng 1 câu lệnh hay 1 khối lệnh.
- Kiểm tra *Biểu thức điều kiện* trước.
- Nếu điều kiện sai (=0) thì thoát khỏi lệnh while.
- Nếu điều kiện đúng (!=0) thì thực hiện công việc rồi quay lại kiểm tra điều kiện tiếp.

Lưu ý:

- Lệnh while gồm có biểu thức điều kiện và thân vòng lặp (khối lệnh thực hiện công việc)
- Vòng lặp dừng lại khi nào điều kiện sai.
- Khối lệnh thực hiện công việc có thể rỗng, có thể làm thay đổi điều kiện.

Ví dụ 1: Viết đoạn chương trình in dãy số nguyên từ 1 đến 10.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   int i;
   clrscr();
   printf("\n Day so tu 1 den 10 :");
   i=1;
   while (i<=10)
        printf("%d ",i++);
   getch();
   return 0;
}
```

Kết quả chương trình như sau:

```
Day so tu 1 den 10 :1 2 3 4 5 6 7 8 9 10
```

Ví dụ 2: Viết chương trình nhập vào một số nguyên n. Tính tổng của các số nguyên từ 1 đến n.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   unsigned int n,i,tong;
   clrscr();
   printf("\n Nhap vao so nguyen duong n:");
   scanf("%d",&n);
   tong=0;
   i=1;
   while (i<=n)
   {
        tong+=i;
        i++;
   }
   printf("\n Tong tu 1 den %d =%d ",n,tong);
   getch();
   return 0;
}
```

Nếu chúng ta nhập vào số 9 thì kết quả như sau:

```
Nhap vao so nguyen duong n:9

Tong tu 1 den 9 =45 _
```

Ví dụ 3: Viết chương trình in ra trên màn hình một ma trận có n dòng m cột như sau:

1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9

…

```
#include <stdio.h>
#include<conio.h>
int main ()
{   unsigned int dong, cot, n, m;
   clrscr();
   printf("\n Nhap vao so dong va so cot :");
   scanf("%d%d",&n,&m);
   dong=0;
   while (dong<n)
   {
      printf("\n");
      cot=1;
      while (cot<=m)
         {
               printf("%d\t",dong+cot);
               cot++;
         }
      dong++;
   }
   getch();
   return 0;
}
```

Kết quả khi nhập 3 dòng 6 cột như sau

```
Nhap vao so dong va so cot :3 6

1       2       3       4       5       6
2       3       4       5       6       7
3       4       5       6       7       8
```

IV.3. Vòng lặp do… while

Vòng lặp do … while giống như vòng lặp for, while, dùng để lặp lại một công việc nào đó khi điều kiện còn đúng.

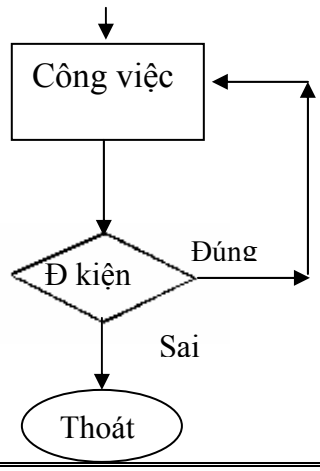
Cú pháp:

do

<Công việc>

while (<*Biểu thức điều kiện***>)**

Lưu đồ:



Giải thích:

- <Công việc>: được thể hiện bằng 1 câu lệnh hay 1 khối lệnh.
- Trước tiên công việc được thực hiện trước, sau đó mới kiểm tra *Biểu thức điều kiện*.
- Nếu điều kiện sai thì thoát khỏi lệnh do …while.
- Nếu điều kiện còn đúng thì thực hiện công việc rồi quay lại kiểm tra điều kiện tiếp.

Lưu ý:

- Lệnh do…while thực hiện công việc ít nhất 1 lần.
- Vòng lặp dừng lại khi điều kiện sai.
- Khối lệnh thực hiện công việc có thể rỗng, có thể làm thay đổi điều kiện.

Ví dụ 1: Viết đoạn chương trình in dãy số nguyên từ 1 đến 10.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   int i;
   clrscr();
   printf("\n Day so tu 1 den 10 :");
   i=1;
   do
        printf("%d ",i++);
   while (i<=10);
   getch();
   return 0;
}
```

Kết quả chương trình như sau:

```
Day so tu 1 den 10 :1 2 3 4 5 6 7 8 9 10
```

Ví dụ 2: Viết chương trình nhập vào một số nguyên n. Tính tổng của các số nguyên từ 1 đến n.

```
#include <stdio.h>
#include<conio.h>
int main ()
{   unsigned int n,i,tong;
   clrscr();
   printf("\n Nhap vao so nguyen duong n:");
   scanf("%d",&n);
   tong=0;
   i=1;
   do
   {
        tong+=i;
        i++;
   } while (i<=n);
   printf("\n Tong tu 1 den %d =%d ",n,tong);
   getch();
   return 0;
}
```

Nếu chúng ta nhập vào số 9 thì kết quả như sau:

```
Nhap vao so nguyen duong n:9
Tong tu 1 den 9 =45 _
```

Ví dụ 3: Viết chương trình in ra trên màn hình một ma trận có n dòng m cột như sau (n, m>=1):

1 2 3 4 5 6 7
2 3 4 5 6 7 8
3 4 5 6 7 8 9
…

```
#include <stdio.h>
#include<conio.h>
int main ()
{   unsigned int dong, cot, n, m;
   clrscr();
   printf("\n Nhap vao so dong va so cot :");
   scanf("%d%d",&n,&m);
   dong=0;
   do
   {
      printf("\n");
      cot=1;
         do
         {
               printf("%d\t",dong+cot);
               cot++;
         } while (cot<=m);
      dong++;
   } while (dong<n);
   getch();
   return 0;
}
```

Kết quả khi nhập 3 dòng 6 cột như sau

```
Nhap vao so dong va so cot :3 6

1       2       3       4       5       6
2       3       4       5       6       7
3       4       5       6       7       8
```

IV.4. So sánh các vòng lặp

Vòng lặp for, while:

- Kiểm tra điều kiện trước thực hiện công việc sau nên đoạn lệnh thực hiện công việc có thể không được thực hiện .

- Vòng lặp kết thúc khi nào điều kiện sai.

Vòng lặp do…while:

- Thực hiện công việc trước kiểm tra điều kiện sau nên đoạn lệnh thực hiện công việc được thực hiện ít nhất 1 lần.

- Vòng lặp kết thúc khi nào điều kiện sai.

V. CÁC CÂU LỆNH ĐẶC BIỆT

V.1. Lệnh break

Cú pháp: **break**

Dùng để thoát khỏi vòng lặp. Khi gặp câu lệnh này trong vòng lặp, chương trình sẽ thoát ra khỏi vòng lặp và chỉ đến câu lệnh liền sau nó. Nếu nhiều vòng lặp --> break sẽ thoát ra khỏi vòng lặp gần nhất. Ngoài ra, break còn được dùng trong cấu trúc lựa chọn switch.

IV.2. Lệnh continue

Cú pháp: **continue**

- Khi gặp lệnh này trong các vòng lặp, chương trình sẽ bỏ qua phần còn lại trong vòng lặp và tiếp tục thực hiện lần lặp tiếp theo.

- Đối với lệnh for, *biểu thức 3* sẽ được tính trị và quay lại bước 2.

- Đối với lệnh while, do while; *biểu thức điều kiện* sẽ được tính và xét xem có thể tiếp tục thực hiện <*Công việc*> nữa hay không? (dựa vào kết quả của *biểu thức điều kiện*).

VI. BÀI TẬP

VI.1 Mục đích yêu cầu

Làm quen và nắm vững các lệnh có cấu trúc của C, biết cách chọn lựa trong trường hợp nào sẽ sử dụng cấu trúc nào. Thực hiện các các chương trình trong phần nội dung bằng cách kết hợp các lệnh lặp, các lệnh rẽ nhánh và các lệnh đơn.

VI.2 Nội dung

1. Viết chương trình nhập 3 số từ bàn phím, tìm số lớn nhất trong 3 số đó, in kết quả lên màn hình.

2. Viết chương trình tính chu vi, diện tích của tam giác với yêu cầu sau khi nhập 3 số a, b, c phải kiểm tra lại xem a, b, c có tạo thành một tam giác không? Nếu có thì tính chu vi và diện tích. Nếu không thì in ra câu " Không tạo thành tam giác".

3. Viết chương trình giải phương trình bậc nhất ax+b=0 với a, b nhập từ bàn phím.

4. Viết chương trình giải phương trình bậc hai $ax^2+bx + c = 0$ với a, b, c nhập từ bàn phím.

5. Viết chương trình nhập từ bàn phím 2 số a, b và một ký tự ch.

Nếu: ch là "+" thì thực hiện phép tính a + b và in kết quả lên màn hình.

ch là "–" thì thực hiện phép tính a - b và in kết quả lên màn hình.

ch là "*" thì thực hiện phép tính a * b và in kết quả lên màn hình.

ch là "/" thì thực hiện phép tính a / b và in kết quả lên màn hình.

6. Viết chương trình nhập vào 2 số là tháng và năm của một năm. Xét xem tháng đó có bao nhiêu ngày? Biết rằng:

Nếu tháng là 4, 6, 9, 11 thì số ngày là 30.

Nếu tháng là 1, 3, 5, 7, 8, 10, 12 thì số ngày là 31.

Nếu tháng là 2 và năm nhuận thì số ngày 29, ngược lại thì số ngày là 28.

7. Có hai phương thức gửi tiền tiết kiệm: gửi không kỳ hạn lãi suất 2.4%/tháng, mỗi tháng tính lãi một lần, gửi có kỳ hạn 3 tháng lãi suất 4%/tháng, 3 tháng tính lãi một lần.

Viết chương trình tính tổng cộng số tiền cả vốn lẫn lời sau một thời gian gửi nhập từ bàn phím.

8. Một số nguyên dương chia hết cho 3 nếu tổng các chữ số của nó chia hết cho 3. Viết chương trình nhập vào một số có 3 chữ số, kiểm tra số đó có chia hết cho 3 dùng tính chất trên.(if)

9. Trò chơi "Oẳn tù tì": trò chơi có 2 người chơi mỗi người sẽ dùng tay để biểu thị một trong 3 công cụ sau: Kéo, Bao và Búa.

Nguyên tắc: Kéo thắng bao.
Bao thắng búa.
Búa thắng kéo.

Viết chương trình mô phỏng trò chơi này cho hai người chơi và người chơi với máy. (switch)

10. Viết chương trình tính tiền điện gồm các khoản sau:

Tiền thuê bao điện kế : 1000 đồng / tháng.

Định mức sử dụng điện cho mỗi hộ là 50 Kw

Phần định mức tính giá 450 đồng /Kwh

Nếu phần vượt định mức <= 50 Kw tính giá phạt cho phần này là 700 đồng/Kwh .

Nếu phần vượt định mức lớn 50 Kw và nhỏ hơn 100Kw tính giá phạt cho phần này là 910 đồng/Kwh

Nếu phần vượt định mức lớn hơn hay bằng 100 Kw tính giá phạt cho phần này là 1200 đồng/Kwh .

Với : chỉ số điện kế cũ và chỉ số điện kế mới nhập vào từ bàn phím. In ra màn hình số tiền trả trong định mức, vượt định mức và tổng của chúng. (if)

11. Viết chương trình nhận vào giờ, phút, giây dạng (hh:mm:ss), từ bàn phím. Cộng thêm một số giây vào và in ra kết quả dưới dạng (hh:mm:ss).

12.Viết chương trình nhập vào ngày tháng năm của một ngày, kiểm tra nó có hợp lệ không.

13. Kiểm tra một ký tự nhập vào thuộc tập hợp nào trong các tập ký tự sau:

Các ký tự chữ hoa: 'A' ...'Z'

Các ký tự chữ thường: 'a' ... 'z'

Các ký tự chữ số : '0' ... '9'

Các ký tự khác.

14. Hệ thập lục phân dùng 16 ký số bao gồm các ký tự 0 .. 9 và A, B, C, D, E ,F.

Các ký số A, B, C, D, E, F có giá trị tương ứng trong hệ thập phân như sau:

A	10
B	11
C	12
D	13
E	14
F	15

Hãy viết chương trình cho nhập vào ký tự biểu diễn một ký số của hệ thập lục phân và cho biết giá trị thập phân tương ứng. Trường hợp ký tự nhập vào không thuộc các ký số trên, đưa ra thông báo lỗi :

"Hệ thập lục phân không dùng ký số này"

15. Viết chương trình nhập vào ngày tháng năm của ngày hôm nay, in ra ngày tháng năm của ngày mai.

16. Viết chương trình tính các tổng sau:

a) $S=1 + 2 + ... + n$

b) $S=1/2 + 2/3 + ... +n/(n+1)$

c) $S= - 1 +2 - 3 +4 - ... + (-1)^n n$

17. Viết chương trình nhập vào một dãy n số, tìm số lớn nhất của dãy và xác định vị trí của số lớn nhất trong dãy.

18. Fibonacci là một dãy số được định nghĩa như sau:

$$F_n = \begin{cases} 1, nÕu\, n = 1 \\ 2, nÕu\, n = 2 \\ F_{n-1} + F_{n-2}, nÕu\, n > 2 \end{cases}$$

Viết chương trình in ra màn hình dãy Fibonacci có n số hạng, n nhập từ bàn phím khi cho chạy chương trình.

19. Viết chương trình đếm số chữ số của một số nguyên n.

20. Tìm số nguyên dương k nhỏ nhất sao cho $2^k > n$ với n là một số nguyên dương nhập từ bàn phím.

21. Viết chương trình in ra số đảo ngược của một số nguyên n, với n nhập từ bàn phím.

22. Tính giá trị trung bình của một dãy số thực, kết thúc dãy với -1.

23. Viết chương trình mô phỏng phép chia nguyên DIV 2 số nguyên a và b như sau: để chia nguyên a và b ta tính trị a-b, sau đó lấy hiệu tìm được lại trừ cho b... tiếp tục cho đến khi hiệu của nó nhỏ hơn b. Số lần thực hiện được các phép trừ ở trên sẽ bằng trị của phép chia nguyên.

24. Tìm số nguyên dương N nhỏ nhất sao cho

$1+1/2+ ...+1/N > S$, với S nhập từ bàn phím.

25. Viết chương trình tính $P=2*4*6*...*(2n)$, n nhập từ bàn phím.

26. Viết chương trình tìm UCLN và BCNN của hai số a và b theo thuật toán sau (Ký hiệu UCLN của a, b là (a,b) còn BCNN là [a,b])

- Nếu a chia hết cho b thì (a,b) = b
- Nếu a = b*q + r thì (a,b) = (b,r)
- [a,b] = a*b/(b,r)

27. Viết chương trình nhập vào một số nguyên dương n, in ra màn hình các số nguyên tố p <= n. Số nguyên p gọi là số nguyên tố nếu p chỉ chia hết cho một và chia hết cho bản thân nó.

28. Viết chương trình tính gần đúng căn bậc hai của một số dương a theo phương pháp Newton : Trước hết cho $x_0=(1 + a)/2$ sau đó là công thức truy hồi: $x_{n+1}=(x_n + a/x_n)/2$

Nếu: $\left|\frac{x_{n+1} - x_n}{x_n}\right| < e$ thì căn bậc hai của a bằng x_{n+1}

Trong đó e là một hằng số cho trước làm độ chính xác.

29. Viết chương trình tính gần đúng căn bậc n của một số dương a theo phương pháp Newton : Trước hết cho x_0= a/n sau đó là công thức truy hồi:

$$x_{k+1} = \left| \frac{(n-1)\, x_k^{\,n} + a}{n x_k^{\,n-1}} \right|$$

Nếu $|a - x_n^{\,n}| < e$ thì x_n là căn bậc n của a. Trong đó e là một hằng số cho trước làm độ chính xác. Nếu a < 0 và n chẵn thì không tồn tại căn.