***1.***

#include <stdio.h>

main()

{

int a=0;

while (a<=10)

{

printf("%d ",a);

a++;

}

return 0;

}

***2.***

#include <stdio.h>

main()

{

int a=0;

while (a<=100)

{

if((a%2==0) || (a%3==0))

printf("%d ",a);

a++;

}

return 0;

}

***3.***

#include <stdio.h>

main()

{

int a=0;

scanf("%d",&a);

while ((a<0) || (a>10))

{

scanf("%d",a);

}

printf("diem la %d",a);

return 0;

}

#include <stdio.h>

main()

{

int a=0;

long dem = 0;

while (a<=1+a)

{

dem++;

a++;

}

printf("%d",dem);

return 0;

}

#include <stdio.h>

main(){

int a;

do

{

scanf("%d",&a);

}

while ((a<0) || (a>10));

printf("diem la %d",a);

return 0;

}

**A.Lệnh rẽ nhánh không điều kiện**

**1.Lệnh break:**

Thông thường lệnh break dùng để thoát khỏi vòng lặp không xác định điều kiện dừng hoặc bạn muốn dừng vòng lặp theo điều kiện do bạn chỉ định Việc dùng lệnh break để thoát khỏi vòng lặp thường sử dụng phối hợp với lệnh if. Lệnh break dùng trong for, while, do…while, switch. Lệnh break thoát khỏi vòng lặp chứa nó.  
Trong bài trước chúng ta đã sử dụng lệnh break trong switch để nhảy bỏ các câu lệnh kế tiếp còn lại.

**2.Lệnh goto**

Lệnh goto dùng để chuyển tới một câu lệnh nào đó đứng sau nhãn đã chỉ định. Nhãn là tên hợp lệnh đứng trước lệnh mà ta cần nhảy tới. Lệnh goto kết hợp với lệnh if cũng có thể tạo ra một vòng lặp.  
Cú pháp lệnh goto: **goto nhan;**  
Cú pháp nhãn: **nhan : lệnh;**

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | #include <stdio.h>    int main(){      int a, kt;      tieptuc : printf("Nhap a = ");      scanf("%d", &a);        switch(a) {          case 1: printf("Mot"); break;          case 2: printf("Hai"); break;          case 3: printf("Ba"); break;          case 4: printf("Bon"); break;          case 5: printf("Nam"); break;          default: printf("Khong phai so tu 1 den 5");      }        printf("\nBan muon tiep tuc? 1 de tiep tuc, 0 de dung: ");      scanf("%d", &kt);        if(kt) goto tieptuc;        return 0;  } |

**Tuy nhiên khi sử dụng lệnh goto bạn cần chú ý các điểm sau:**  
-Nếu lệnh goto và nhãn nằm trong 1 hàm thì lệnh goto chỉ được phép nhảy trong hàm đó và không được phép nhảy sang hàm khác.  
-Không được dùng lệnh goto nhảy từ ngoài vào trong một khối nhưng được phép nhảy từ trong khối ra ngoài.

**3.Lệnh continue**

Được dùng trong vòng lặp for, while, do…while. Khi lệnh continue thi hành quyền điều khiển sẽ trao qua cho biểu thức điều kiện của vòng lặp gần nhất. Nghĩa là lộn ngược lên đầu vòng lặp, tất cả những lệnh đi sau trong vòng lặp chứa continue sẽ bị bỏ qua không thi hành.

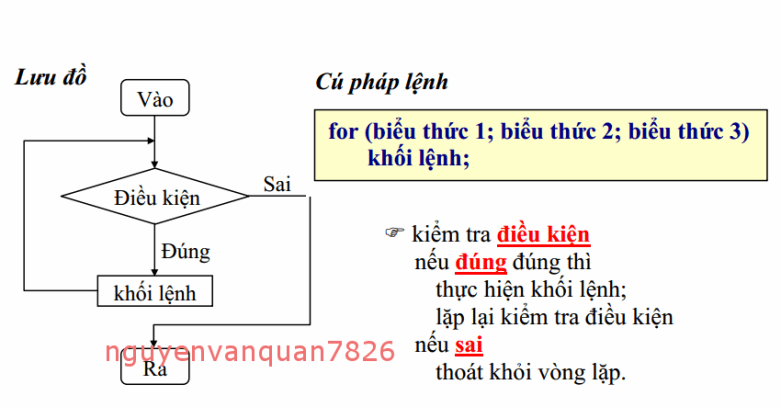
Bây giờ chúng ta sẽ đi tìm hiểu các lệnh vòng lặp để từ đó thấy rõ hơn được vai trò của các lệnh rẽ nhánh không điều kiện

[qads]

**B.Các lệnh vòng lặp**

**1.Lệnh for**

Vòng lặp xác định thực hiện lặp lại một số lần xác định của một (hoặc một chuỗi) hành động.

Lưu đồ vòng lặp for

**Các biểu thức :**

* Biểu thức 1: khởi tạo giá trị ban đầu cho biến điều khiển.
* Biểu thức 2: là quan hệ logic thể hiện điều kiện tiếp tục vòng lặp.
* Biểu thức 3: phép gán dùng thay đổi giá trịbiến điều khiển.

**Cách hoạt động:**

* Bước 1: Xác định giá trị của biểu thức 1
* Bước 2: Xác định giá trị của biểu thức 2 (kiểm tra điều kiện đúng hay sai tức 1 hoặc 0). Tùy vào giá trị của biểu thức 2 mà thực hiện theo 2 hướng:
  + + Nếu sai (0) sẽ thoát khỏi vòng lặp
  + + Nếu đung (1) sẽ thực hiện khối lệnh trong vòng for đến khi gặp dấu } ở cuối vòng for hoặc lệnh continue thì sẽ tưới bước 3.
* Bước 3: Tính giá trị biểu thức 3, sau đó quay lại bước 2.

**Lưu ý:**

* Biểu thức 1 bao giờ cũng chỉ được tính toán một lần khi gọi thực hiện for.
* Biểu thức 2, 3 và thân for có thể thực hiện lặp lại nhiều lần.
* Biểu thức 1, 2, 3 phải phân cách bằng dấu chấm phẩy ( ; ), các biểu thức có thể vắng mặt 1,2 hoặc cả 3 nhưng vẫn phải có dấy chấm phẩy này.
* Nếu biểu thức 2 không có, vòng for được xem là luôn luôn đúng. Muốn thoát khỏi vòng lặp for phải dùng một trong lệnh break, goto hoặc return.
* Với mỗi biểu thức có thể viết thành một dãy biểu thức con phân cách nhau bởi dấu phẩy. Khi đó các biểu thức con được xác định từ trái sang phải. Tính đúng sai của dãy biểu thức con trong biểu thức thứ 2 được xác định bởi biểu thức con cuối cùng.
* Trong thân for (khối lệnh) có thể chứa một hoặc nhiều cấu trúc điều khiển khác, vòng lặp khác.
* Khi gặp lệnh break, cấu trúc lặp sâu nhất sẽ thoát ra.
* Trong thân for có thểdùng lệnh goto để thoát khỏi vòng lặp đến vị trí mong muốn.
* Trong thân for có thể sử dụng return để trở về một hàm nào đó.
* Trong thân for có thể sử dụng lệnh continue để chuyển đến đầu vòng lặp (bỏ qua các câu lệnh còn lại trong thân).

**Ví dụ về lệnh for.**

Trong các ví dụ dưới đây mình sẽ thực hiện trên kiểu mảng (sẽ học ở các bài sau) nên nếu các bạn chưa rõ về mảng thì chỉ cần quan tâm tới cấu trúc lệnh **for** của chúng ta chứ không cần thiết quan tâm tới kiểu mảng.

**VD1**: Sử dụng vòng **for** in các số nguyên từ 1 đến 10

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 01  02  03  04  05  06  07  08  09  10 | #include <stdio.h>    int main(){      int i;      for (i = 1; i <= 10; i++) {          printf("%-5d", i);      }        return 0;  } |

**VD2:** Sử dụng vòng for in để đảo ngược mảng trong đó các biểu thức của vòng for là một dãy biểu thức con phân cách nhau bởi dấu phẩy.

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32 | #include <stdio.h>    int main(){      int i, j, n = 4, a[] = {1, 3, 7, 2}; // mang a gom 4 phan tu        /\* Xuat mang ban dau\*/        printf("Mang ban dau: ");      for (i = 0; i < n; i++) {          printf("%-5d",a[i]);      }      printf("\n");        /\* Dao nguoc mang\*/        for (i = 0, j = n - 1; i <= j; i++, j--) {          // doi cho 2 phan tu a[i] va a[j]          int temp = a[i];          a[i] = a[j];          a[j] = temp;      }        /\* Xuat mang sau khi dao nguoc\*/        printf("Mang dao nguoc: ");      for (i = 0; i < n; i++) {          printf("%-5d",a[i]);      }      printf("\n");        return 0;  } |

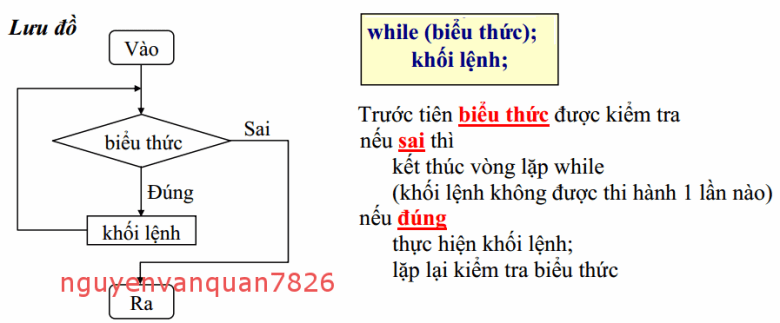
**VD3:** Sử dụng vòng for in để đảo ngược mảng trong đó vắng mặt biểu thức 2, ta sử dụng lệnh break để thoát khỏi vòng lặp

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33 | #include <stdio.h>    int main(){      int i, j, n = 4, a[] = {1, 3, 7, 2}; // mang a gom 4 phan tu        /\* Xuat mang ban dau\*/        printf("Mang ban dau: ");      for (i = 0; i < n; i++) {          printf("%-5d",a[i]);      }      printf("\n");        /\* Dao nguoc mang\*/        for (i = 0, j = n - 1; ; i++, j--) {          if (i >= j) break;          // doi cho 2 phan tu a[i] va a[j]          int temp = a[i];          a[i] = a[j];          a[j] = temp;      }        /\* Xuat mang sau khi dao nguoc\*/        printf("Mang dao nguoc: ");      for (i = 0; i < n; i++) {          printf("%-5d",a[i]);      }      printf("\n");        return 0;  } |

**VD4**: Minh họa lệnh continue trong vòng for.

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 01  02  03  04  05  06  07  08  09  10  11  12  13 | #include <stdio.h>    int main(){      int i, n = 4, a[] = {1, -3, -7, 2}; // mang a gom 4 phan tu        printf("Cac so duong trong mang: ");      for (i = 0; i < n; i++) {          if (a[i] <= 0) continue;          printf("%-5d",a[i]);      }        return 0;  } |

**2. Lệnh while**

Lưu đô lệnh while

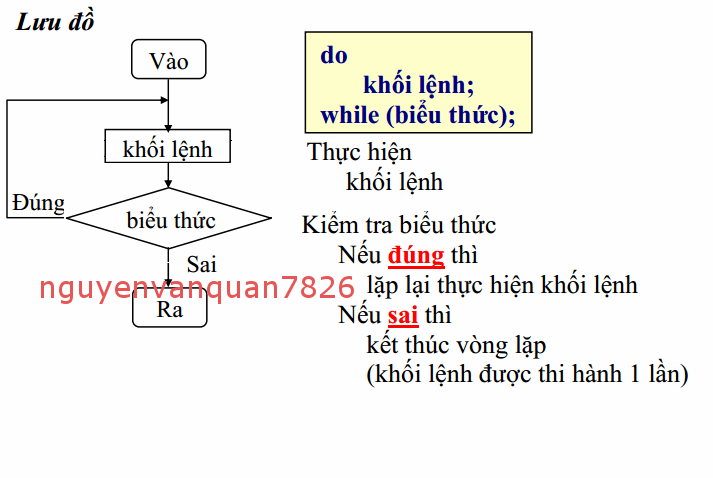
Ta có thể hiểu đơn giản: **Trong khi biểu thức còn đúng thì làm lệnh, khi sai thì thoát**

* Biểu thức: có thể là một biểu thức hoặc nhiều biểu thức con. Nếu là nhiều biểu thức con thì cách nhau bởi dấu phẩy (,) và tính đúng sai của biểu thức được quyết định bởi biểu thức con cuối cùng.
* Trong thân while (khối lệnh) có thể chứa một hoặc nhiều cấu trúc điều khiển khác.
* Trong thân while có thể sử dụng lệnh continue đểchuyển đến đầu vòng lặp (bỏqua các câu lệnh còn lại trong thân).
* Nếu biểu thức là một hằng khác không thì nó luôn đúng và vòng lặp diễn ra vô hạn. Muốn thoát khỏi vòng lặp while tùy ý có thể dùng các lệnh break, goto, return như với lệnh for.

VD: Nhập mật khẩu:

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 01  02  03  04  05  06  07  08  09  10  11  12  13  14  15  16  17 | #include <stdio.h>  #define PASS "nguyenvanquan7826" // dat pass    int main(){      char pass[100];        /\* Ham strcmp  tra ve ket qua la 0 khi 2 chuoi bang nhau  \*/        while(strcmp(pass, PASS) != 0) {          printf("Nhap pass: ");          gets(pass);      }        printf("Mat khau dung roi!");        return 0;  } |

**3. Lệnh do while**

Lưu đồ lệnh do-while

Ta có thể hiểu đơn giản: **Làm lệnh trong khi biểu thức đúng, hễ sai thì thoát.**

Từ đây ta có nhận xét: vòng lặp while thì có thể không thực hiện lệnh nào nếu ngay từ ban đầu biểu thức sai, còn lệnh do while thì sẽ thực hiện lệnh ít nhất 1 lần dù biểu thức có sai ngay từ đầu.

VD:

|  |  |
| --- | --- |
| code by nguyenvanquan7826 | |
| 1  2  3  4  5  6  7  8  9 | #include <stdio.h>    int main(){      do {          printf("khoi lenh duoc thuc hien\n");      } while(5 > 8);        return 0;  } |