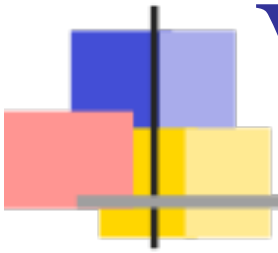
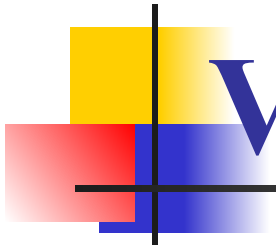


Vòng lặp

Chương 6

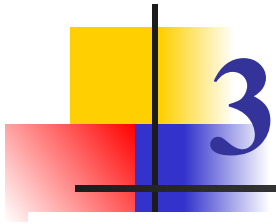
Mục tiêu của bài học

- Tìm hiểu về vòng lặp ‘for’ trong C
- Làm việc với toán tử dấu phẩy (,)
- Tìm hiểu về các vòng lặp lồng nhau
- Tìm hiểu về vòng lặp ‘while’ và ‘do-while’
- Làm việc với các lệnh break và continue
- Tìm hiểu về hàm exit()



Vòng lặp là gì?

Một đoạn mã lệnh trong chương trình thực hiện lặp đi lặp lại cho đến khi một điều kiện xác định được thỏa mãn



3 kiểu cấu trúc vòng lặp

Vòng lặp for

Vòng lặp while

Vòng lặp do….while

Vòng lặp for

Cú pháp:

```
for (initialize counter; conditional test; re-evaluation
    parameter){
        statement
}
```

- *initialize counter* là một lệnh gán để khởi tạo biến điều khiển của vòng lặp trước khi đi vào vòng lặp
- *conditional test* là một biểu thức quan hệ để chỉ định khi nào vòng lặp sẽ kết thúc
- *re-evaluation parameter* định nghĩa cách thức thay đổi của biến điều khiển vòng lặp mỗi khi vòng lặp được thực thi

Vòng lặp for (tt.)

- Ba phần của vòng lặp **for** phải được phân cách bởi dấu chấm phẩy(;)
- Phần lệnh tạo nên thân vòng lặp có thể là một lệnh đơn hoặc một lệnh ghép (một tập nhiều lệnh)
- Vòng lặp **for** tiếp tục được thực thi khi biểu thức kiểm tra điều kiện vẫn có giá trị **true**. Khi điều kiện trở thành **false**, chương trình thực hiện lệnh theo sau vòng lặp **for**

Vòng lặp for - Ví dụ

```
/*This program demonstrates
the for loop in a C program*/
   #include <stdio.h>
    main()     {
         int count;
         printf("\tThis is a \n");
         for(count = 1;count
  <=6;count++)
              printf("\n\t\t nice");
         printf("\n\t\t world. \n");
    }
```

Toán tử dấu phẩy

Vòng lặp **for** có thể được mở rộng bằng cách chứa nhiều giá trị khởi tạo và nhiều biểu thức tăng trị trong đặc tả của vòng lặp for

Cú pháp: **exprn1 , exprn2 ;**

```
#include <stdio.h>
   main()     {
        int i, j , max;
        printf("Please enter the maximum value \n");
        printf("for which a table can be printed:");
        scanf("%d", &max);
        for(i = 0 , j = max ; i <=max ; i++, j--)
            printf("\n%d + %d = %d",i, j, i + j);
   }
```

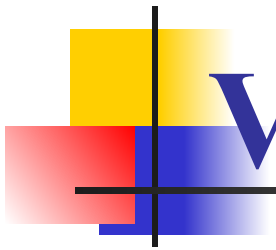
Vòng lặp for lồng nhau

Các vòng lặp for lồng nhau khi nó có dạng như sau

```
for (i = 1; i<max1; i++)     {
        …
          for (j = 0; j < = max2; j++){
          …
          }
          …
}
```

Vòng lặp for lồng nhau - Ví dụ

```
#include <stdio.h>
    main()  {
         int i, j, k;
         i = 0;
         printf("Enter no. of rows :");
         scanf("%d", &i);
         printf("\n");
         for (j = 0; j < i ; j++){
              printf("\n");
              for (k = 0; k <= j; k++) /*inner for
loop*/
              printf("*");
         }
    }
```



Vòng lặp while

Cú pháp

```
while (condition is true)
        statement ;
```

Vòng lặp while lặp lại các lệnh trong khi một biểu thức điều kiện mang giá trị True

Vòng lặp while -Ví dụ

```
/*A simple program using the while loop*/
   #include <stdio.h>
   main() {
        int count = 1;
        while( count <= 10)  {
          printf("\n This is iteration
                        %d\n",count);
          count++;
        }
        printf("\n The loop is
completed.\n");
  }
```

Vòng lặp do…while

Cú pháp

```
do{
    statement;
} while (condition);
```

- Trong vòng lặp do while phần thân của vòng lặp được thực thi trước khi biểu thức điều kiện được kiểm tra
- Khi điều kiện mang giá trị False, vòng lặp **do while** sẽ được kết thúc, và điều khiển chuyển đến lệnh xuất hiện ngay sau lệnh while

Vòng lặp do…while - Ví dụ

```
#include <stdio.h>
 main (){
   int num1, num2;
 num2 = 0;
   do     {
        printf( "\nEnter a number : ");
        scanf(“%d”,&num1);
        printf( " No. is %d",num1);
        num2++;
   } while (num1 != 0);
   printf ("\nThe total numbers entered were
              %d",--num2);
/*num2 is decremented before printing because count
for last integer (0) is not to be considered */
   }
```

Các lệnh chuyển điều khiển

return expression

- Lệnh return được sử dụng để trở về từ một hàm
- Thực hiện lệnh return để trở về vị trí mà tại đó hàm được gọi
- Lệnh return có thể có một giá trị đi cùng, giá trị này được trả về cho chương trình gọi

Các lệnh chuyển điều khiển (tt.)

goto label

- Lệnh goto chuyển điều khiển đến một câu lệnh bất kỳ khác bên trong cùng một hàm trong một chương trình C
- Điều này thật ra vi phạm đến qui luật của một ngôn ngữ lập trình cấu trúc.
- Chúng làm giảm độ tin cậy của chương trình và chương trình khó bảo trì.

Các lệnh chuyển điều khiển (tt.)

break statement

- Lệnh break được sử dụng để kết thúc một mệnh đề ***case*** trong câu lệnh ***switch***
- Nó cũng có thể được sử dụng để kết thúc ngang giữa vòng lặp
- Khi gặp lệnh break, vòng lặp sẽ kết thúc ngay và điều khiển được chuyển đến lệnh kế tiếp bên ngoài vòng lặp

Lệnh break – Ví dụ

```
#include <stdio.h>
    main (){
         int count1, count2;
         for(count1 = 1, count2 = 0;
                count1 <=100; count1++){
               printf("Enter %d count2:
",                          count1);
               scanf("%d", &count2);
               if(j==100) break;
         }
    }
```

Các lệnh chuyển điều khiển (tt.)

continue statement

- Lệnh ***continue*** dùng để bắt đầu thực hiện lần lặp kế tiếp của vòng lặp
- Khi gặp lệnh ***continue***, các câu lệnh còn lại trong thân vòng lặp bị bỏ qua và điều khiển được chuyển đến lần lặp kế tiếp

Lệnh continue – Ví dụ

```
#include <stdio.h>
    main ()     {
          int num;
          for(num = 1;num<=100; num++){
               if(num % 9 == 0)
                    continue;
               printf("%d\t",num);
          }
    }
```

Các lệnh chuyển điều khiển (tt)

hàm **exit()**

- Hàm exit() được sử dụng để thoát khỏi chương trình
- Sử dụng hàm này sẽ kết thúc ngay chương trình và điều khiển được chuyển về cho hệ điều hành