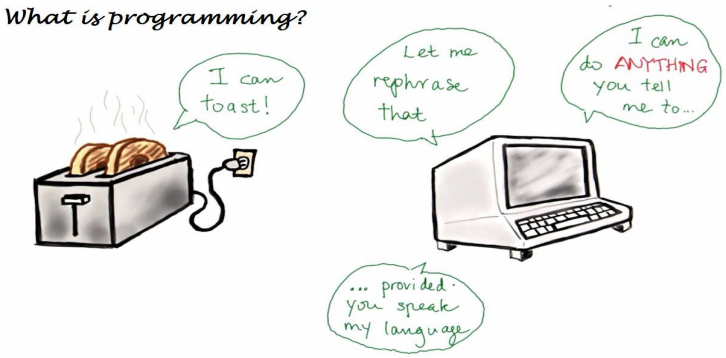


Tuần 3

Cấu trúc lặp

CT101 – Lập trình căn bản

Khoa CNTT&TT – ĐHCT

Mục đích

- Cung cấp khái niệm về điều khiển thứ tự thực hiện câu lệnh
- Giới thiệu lệnh lặp, sự cần thiết của cấu trúc lặp
- Giới thiệu các lệnh lặp trong C và cách sử dụng

Yêu cầu

- Hiểu khái niệm thứ tự thực hiện các câu lệnh
- Biết cách sử dụng lệnh lặp
- Phân biệt được các lệnh lặp trong C
- Ứng dụng cấu trúc lặp để viết chương trình

Nội dung

- Thứ tự thực hiện các câu lệnh
- Sự cần thiết của cấu trúc lặp
- Lệnh lặp `for`
- Lệnh lặp `while`
- Lệnh lặp `do while`
- Câu lệnh `break`

Thứ tự thực hiện câu lệnh

Điểm bắt đầu của một chương trình C

```
#include <stdio.h>
int main (void)
{
  int value1, value2, sum;
  value1 = 50;
  value2 = 25;
  sum = value1 + value2;
  printf ("The sum of %d and %d is %d\n",value1, value2, sum);
  return 0;
}
```

Thứ tự thực hiện các lệnh

Thứ tự thực hiện câu lệnh

- Các dạng điều khiển thứ tự thực hiện câu lệnh:
  - Thực hiện tuần tự dãy các câu lệnh
  - Thực hiện nhánh các câu lệnh tùy theo giá trị của biểu thức điều kiện (cấu trúc rẽ nhánh)
  - Lặp lại một dãy các câu lệnh (cấu trúc lặp)

```
Statement1
Statement2
Statement3
Statement4
Statement5
Statement6
Statement7
Statement8
```

Sự cần thiết của cấu trúc lặp

Ví dụ: tính tổng các số nguyên từ 1 đến n

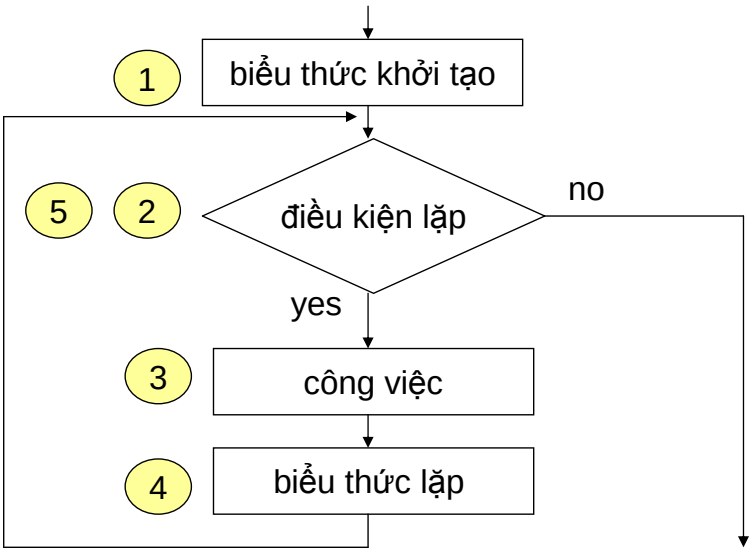
```
// Chương trình với n = 8
#include <stdio.h>
int main (void) {
    int tong;
    tong = 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8;
    printf ("Tong cac so nguyen tu 1 den 8 la %d", tong);
    return 0;
}
```

Làm thế nào nếu cần tính với n = 200, n=1000?

Trong C có 3 cấu trúc lặp: for, while, do

Câu lệnh for

for (*biểu thức khởi tạo; điều kiện lặp; biểu thức lặp*)
công việc



Câu lệnh for

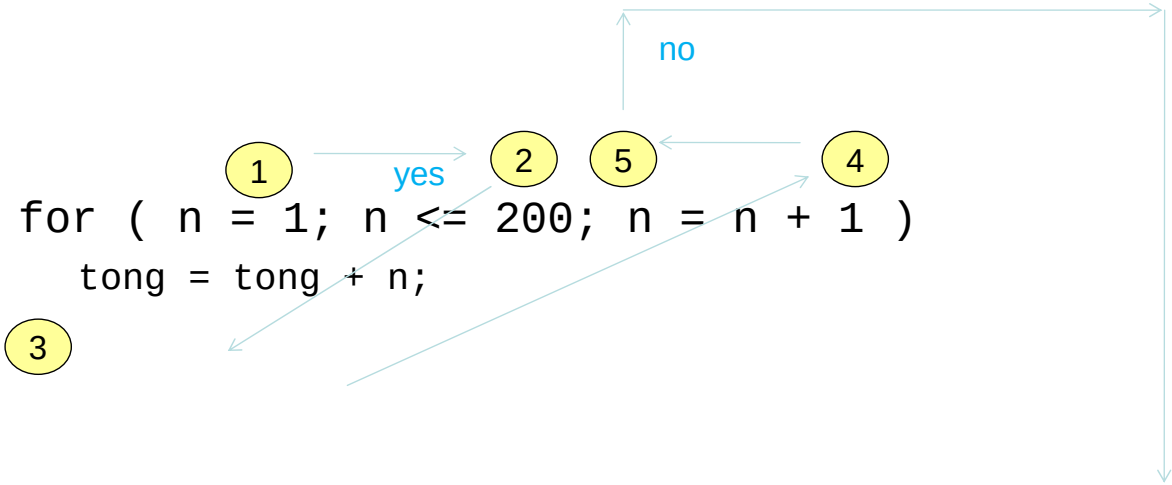
1. Bắt đầu vòng lặp, biểu thức khởi tạo được thực hiện đầu tiên. Biểu thức này thường dùng để đặt giá trị ban đầu cho biến *điều khiển* vòng lặp.
2. Tiếp theo, biểu thức điều kiện lặp được định trị. Nếu kết quả biểu thức là sai (bằng 0) vòng lặp dừng. Ngược lại, một lần lặp mới sẽ bắt đầu.
3. Khi một lần lặp bắt đầu, các câu lệnh trong thân vòng lặp (công việc) sẽ được thực hiện.
4. Biểu thức lặp được thực hiện. Thông thường biểu thức này dùng để thay đổi giá trị biến điều khiển vòng lặp
5. Quay lại bước 2.

Câu lệnh for

```
/* Chương trình tính tổng các số nguyên từ 1 đến 200 */

#include <stdio.h>
int main (void)
{
    int n, tong;
    tong = 0;
    for ( n = 1; n <= 200; n = n + 1 )
        tong = tong + n;
    printf ("Tong cac so nguyen tu 1 den 200 la %d",tong);
    return 0;
}
```

Câu lệnh for



Câu lệnh for

```
// Program to generate a table of triangular numbers
#include <stdio.h>
int main (void)
{
    int n, triangularNumber;
    printf ("TABLE OF TRIANGULAR NUMBERS\n\n");
    printf (" n Sum from 1 to n\n");
    printf ("--- ---------------\n");
    triangularNumber = 0;
    for ( n = 1; n <= 10; ++n ) {
        triangularNumber += n;
        printf (" %i %i\n", n, triangularNumber);
    }
    return 0;
}
```

Thân của vòng lặp trong trường hợp này là một khối gồm 2 câu lệnh

Câu lệnh for

```
#include <stdio.h>
int main (void)
{
    int n, number, triangularNumber, counter;
    for ( counter = 1; counter <= 5; ++counter ) {
        printf ("What triangular number do you want? ");
        scanf ("%i", &number);
        triangularNumber = 0;
        for ( n = 1; n <= number; ++n )
                triangularNumber += n;
        printf ("Triangular number %i is %i\n\n", number,
                                        triangularNumber);
    }
    return 0;
}
```

Vòng lặp lồng nhau. Nên lùi vào một cấp khi viết các lệnh con của lệnh khác

Vòng lặp không dừng

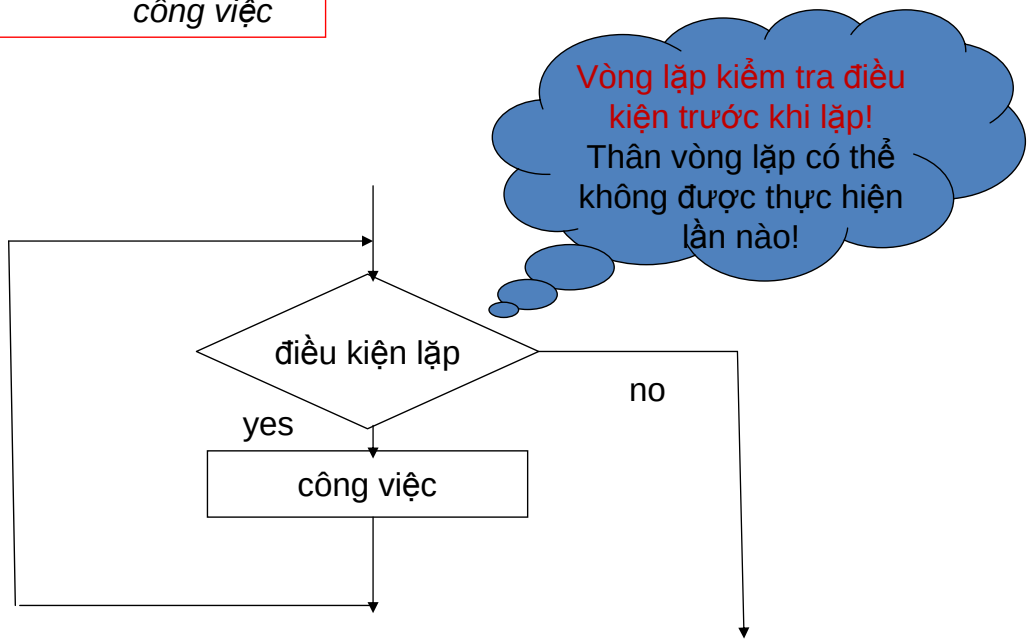
```
#include <stdio.h>
int main (void)
{
    int  i, n = 5, sum =0;
    for ( i = 1; i <= n; n = n + 1 ){
      sum = sum + i;
       printf (“%d  %d  %d\n", i , sum, n);
    }
    return 0;
}
```

Có điều gì không ổn ở đây?
Vòng lặp có kết thúc không?

Nhiệm vụ của người lập trình là phải thiết kế giải thuật đúng đắn để vòng lặp dừng sau một số bước hữu hạn !

Câu lệnh while

while (*điều kiện lặp*)
    công việc



Câu lệnh while

```
/* Program to find the greatest common divisor
of two nonnegative integer values */
#include <stdio.h>
int main (void)
{
    int u, v, temp;
    printf ("Please type in two nonnegative integers.\n");
    scanf ("%d%d", &u, &v);
    while ( v != 0 ) {
        temp = u % v;
        u = v;
        v = temp;
    }
    printf ("Their greatest common divisor is %d\n", u);
    return 0;
}
```

Câu lệnh while

```
// Program to reverse the digits of a number
#include <stdio.h>
int main (void)
{
    int number, right_digit;
    printf ("Enter your number.\n");
    scanf ("%d", &number);
    while ( number != 0 ) {
        right_digit = number % 10;
        printf ("%d", right_digit);
        number = number / 10;
    }
    printf ("\n");
    return 0;
}
```

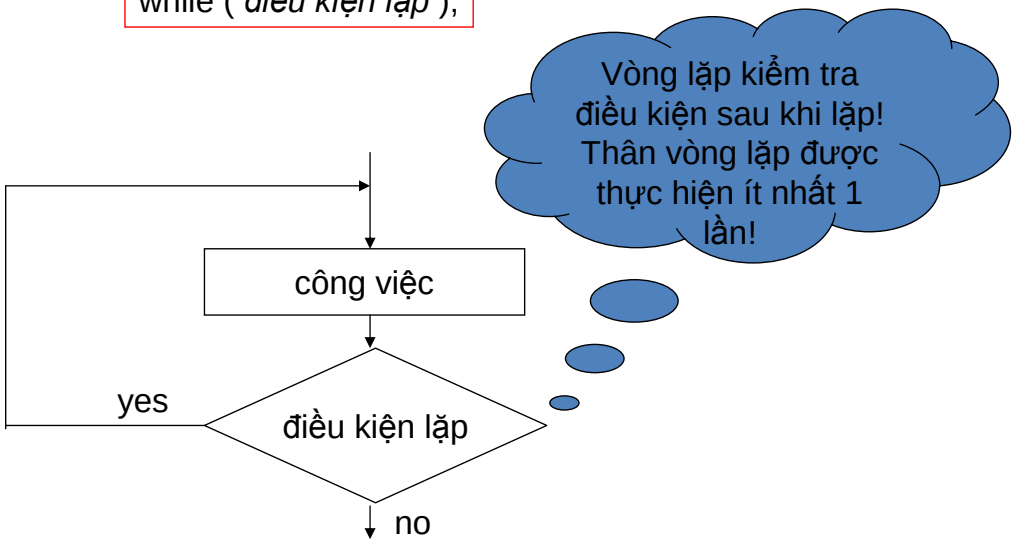

Câu lệnh while

```
// Program to reverse the digits of a number
#include <stdio.h>
int main (void)
{
    int number, right_digit;
    printf ("Enter your number.\n");
    scanf ("%d", &number);
    while ( number != 0 ) {
        right_digit = number % 10;
        printf ("%d", right_digit);
        number = number / 10;
    }
    printf ("\n");
    return 0;
}
```

Chương trình còn đúng nếu number = 0 hay không?

Câu lệnh do while

```
do
        công việc
while ( điều kiện lặp );
```



Câu lệnh do while

```
// Program to reverse the digits of a number
#include <stdio.h>
int main ()
{
    int number, right_digit;
    printf ("Enter your number.\n");
    scanf ("%i", &number);
    do {
        right_digit = number % 10;
        printf ("%i", right_digit);
        number = number / 10;
    }
    while ( number != 0 );
    printf ("\n");
    return 0;
}
```

So sánh while và for

```
#include <stdio.h>
int main (void)
{
   int count = 1;
   while ( count <= 5 ) {
      printf ("%d\n", count);
      ++count;
   }
   return 0;
}
```

```
#include <stdio.h>
int main (void)
{
   int count;
   for ( count=1; count<=5;
                  count++ ) {
      printf ("%d\n", count);
   }
   return 0;
}
```


Câu lệnh break

```
while ( number != 0 ) {
   // Statements to do something in loop
   printf("Stop, answer 1:  ");
   scanf ("%d", &answer);
   if(answer == 1)
      break; // very bad idea to do this
}
```

Câu lệnh break làm ngưng lập tức việc thực hiện vòng lặp bất kể giá trị biểu thức điều kiện lặp

Tổng kết

- Sử dụng vòng lặp khi cần lặp đi lặp lại một hay nhiều câu lệnh
- Trong C có 3 câu lệnh lặp: `for`, `while`, `do while` Chọn vòng lặp nào ?
- Chọn theo cách xác định điều kiện lặp
  - Kiểm tra điều kiện trước khi lặp: for, while
  - Kiểm tra điều kiện sau khi lặp: do
- Chọn theo số lần lặp:
  - Không xác định trước: while
  - Xác định trước: for

Kiểm tra kiến thức

1. Trong C có bao nhiêu cấu trúc lặp ?

a. 1 b. 2 c. 3 d. 4

2. Số lần lặp ít nhất của vòng lặp while là bao nhiêu?

a. 0 b. 1 c. 2 d. 3

3. Có thể dùng vòng lặp for để thay thế vòng lặp while (và ngược lại) được không ?

a. Không b. Được

4. Nếu thân vòng lặp có từ 2 câu lệnh trở lên ta phải làm gì?

a. Viết lệnh bình thường b. Viết xuống dòng

c. Dùng cặp ngoặc { } để bao các câu lệnh lại d. Tất cả đều sai

5. Các loại vòng lặp có thể dùng lồng nhau được không?

a. Được b. Không

Bài tập tổng kết

1. Viết chương trình tính các tổng sau (với n là số nguyên không âm nhập từ bàn phím):

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

2. Viết chương trình nhập vào một dãy gồm n số nguyên, tìm số lớn nhất của dãy

3. Viết chương trình tìm số nguyên dương k nhỏ nhất sao cho $2^k > n$ với n là một số nguyên dương nhập từ bàn phím

4. Viết chương trình tìm số nguyên dương N nhỏ nhất sao cho 1+1/2+...+1/N > S, với S nhập từ bàn phím

5. Viết chương trình xác định một số nguyên dương N nhập từ bàn phím có phải là số nguyên tố không