Ngôn ngữ lập trình C

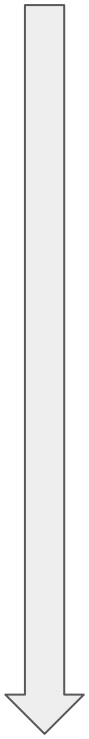
Bài 4. Nhập và xuất theo định dạng

Soạn bởi: TS. Nguyễn Bá Ngọc

2021

Thứ tự ưu tiên và chiều thực hiện chuỗi toán tử

Ưu tiên cao (thực hiện trước)



Ưu tiên thấp (thực hiện sau)

Tên toán tử	**Ký hiệu & ví dụ**	**Thực hiện chuỗi toán tử**
Tăng 1 (hậu tố) Giảm 1 (hậu tố)	++ (x++) -- (x--)	*Trong phạm vi hiện tại hiếm khi cần quan tâm (thường chỉ áp dụng 1 lần cho 1 toán hạng)*
Tăng, giảm 1 (tiền tố) Toán tử dấu (tiền tố) Toán tử sizeof	++, -- (++x, --x) +, - (+x, -x) sizeof (sizeof(int))	
Ép kiểu	() (double)5	Phải -> Trái
Nhân Chia Phần dư	* (x * y) / (x / y) % (x % y)	Trái -> Phải
Cộng Trừ	+ (x + y) - (x - y)	Trái -> Phải
Dịch sang trái Dịch sang phải	<< (x << y) >> (x >> y)	Trái -> Phải
AND theo bit	& (x & y)	Trái -> Phải
XOR theo bit	^ (x ^ y)	Trái -> Phải
OR theo bit	| (x | y)	Trái -> Phải
Gán đơn giản Gán kết hợp	= *=, /=, %=, +=, -=, <<=, >>=, &=, ^=, |=	Phải -> Trái

Các toán tử trong cùng 1 dòng có cùng thứ tự ưu tiên. Các toán tử ở dòng trên có thứ tự ưu tiên cao hơn các toán tử ở dòng dưới.

Nội dung

- Xuất theo định dạng
- Nhập theo định dạng

Nội dung

- Xuất theo định dạng
- Nhập theo định dạng

Xuất theo định dạng

Có thể xuất nội dung văn bản ra màn hình (- luồng xuất tiêu chuẩn/stdout) theo định dạng với hàm printf (thư viện stdio.h).

- Hàm printf có nguyên mẫu giản lược:

```
int printf (const char *format, ...);
```

- Các tham số:
  - format - Bố cục xuất
    - Mẫu nội dung văn bản cần xuất, có thể chứa các đặc tả xuất;
    - Về mặt cú pháp là chuỗi ký tự chỉ đọc, hàm printf nhận diện các đặc tả xuất và thay thế bằng các giá trị của các tham số tương ứng.
  - ... - Danh sách linh động các biểu thức tương ứng với các đặc tả xuất (nếu có) trong bố cục xuất .
- Giá trị trả về: Số lượng ký tự đã được in ra hoặc một số âm nếu có lỗi xuất dữ liệu, kiểu int.

Cấu trúc đặc tả xuất

% [stt-tham số $] *các cờ* **chiều rộng** [. độ chính xác] định kiểu

- Thành phần định kiểu là bắt buộc, các thành phần còn lại có thể có hoặc không.
- stt-tham số $ - Số thứ tự của tham số có thể được thiết lập cho giá trị xuất, chiều rộng và độ chính xác.
- các cờ - Điều chỉnh hình thức xuất giá trị;
- chiều rộng - Điều chỉnh số lượng ký tự tối thiểu được sử dụng;
- độ chính xác - Số lượng chữ số tối thiểu đối với số nguyên/ hoặc số chữ số phần thập phân (sau làm tròn) đối với số thực;
- định kiểu - bao gồm ký tự định kiểu và có thể có các ký tự hiệu chỉnh độ dài (đứng trước ký tự định kiểu nếu có).

Ký tự định kiểu và định dạng mặc định

Trong trường hợp đơn giản nhất đặc tả xuất có thể chỉ bao gồm dấu % và ký tự định kiểu.

Đặc tả	**Kiểu tham số**	**Ý nghĩa**
%d, %i	int	Xuất số nguyên có dấu ở HCS 10
%u	unsigned int	Xuất số nguyên không dấu ở HCS 10
%o		Xuất số nguyên không dấu ở HCS 8
%x, %X		Xuất số nguyên không dấu ở HCS 16
%f, %F	double	Xuất số thực theo định dạng phổ thông
%e, %E		Xuất số thực theo định dạng khoa học (có phần mũ)
%g, %G		Tự lựa chọn giữa %f, %F hoặc %e, %E tùy theo giá trị và độ chính xác

Các mô tả chi tiết có thể được thêm vào tùy theo tình huống ứng dụng, ví dụ khi cần xuất các giá trị theo định dạng bảng, v.v.

Các hiệu chỉnh độ dài

Các ký tự hiệu chỉnh độ dài được thêm vào nếu kiểu của tham số có cùng biểu diễn như kiểu mặc định nhưng khác độ dài

- Đối với kiểu số nguyên:

Hiệu chỉnh	**Định kiểu: d hoặc i**	**u, o, x, hoặc X**
hh (C99)	signed char	unsigned char
h	short	unsigned short
l	long	unsigned long
L, ll, q	long long	unsigned long long
z	Kiểu có dấu của size_t	Kiểu không dấu của size_t

- Đối với kiểu số thực:

Hiệu chỉnh	**Định kiểu: f, F, e, E, g, G**
l	Không ảnh hưởng
L	long double

Lưu ý: Các tham số còn được ép kiểu tự động khi được truyền cho hàm printf (char, short => int; float => double)

Các cờ định dạng xuất (chọn lọc)

‘-’: Căn lề trái thay vì căn lề phải như được mặc định.

‘+’: Xuất dấu + cho số không âm và dấu - cho số âm (printf mặc định không hiển thị dấu cho số dương)

‘0’: Căn chỉnh bằng các chữ số 0 (không làm thay đổi giá trị số đọc được), cờ này bị bỏ qua nếu sử dụng cùng với cờ ‘-’ (căn lề trái).

Ví dụ 4-1. Xuất các giá trị số nguyên

vd4-1.c

```c
#include <stdio.h>

int main() {
   int a = 11, b = -3;
   int n, d;
   n = a / b;
   d = a - n * b;

   // %d là đặc tả xuất số nguyên có dấu
   int cc = printf("n = %d d = %d (%d)\n",
                   n,      d, a % b);
   printf("cc = %d\n", cc);
}
```

```
bangoc:$gcc -o prog vd4-1.c
bangoc:$./prog
n = -3 d = 2 (2)
cc = 17
```

Số lượng ký tự đã được xuất

Bố cục xuất

Các đặc tả xuất sẽ được thay thế bởi các giá trị của các biểu thức tương ứng.

Trong câu lệnh này chúng ta bỏ qua giá trị trả về của hàm printf

Ví dụ 4-2a. Căn chỉnh

vd4-2a.c

```c
#include <stdio.h>

int main() {
  printf("12345678901234567890\n");
  int i = 123;
  printf("%d| (%%d)\n", i);
  printf("%10d| (%%10d)\n", i);
  printf("%-10d| (%%-10d)\n", i);
  printf("%010d| (%%010d)\n", i);
  printf("%-10.5d| (%%-10.5d)\n", i);
  double d = 3.1415;
  printf("%f| (%%f)\n", d);
  printf("%10f| (%%10f)\n", d);
  printf("%-10f| (%%-10f)\n", d);
  printf("%010f| (%%010f)\n", d);
  printf("%10.3f| (%%10.3f)\n", d);
}
```

20 ký tự

```
bangoc:$gcc -o prog vd4-2a.c
bangoc:$./prog
12345678901234567890
123| (%d)
       123| (%10d)
123        | (%-10d)
0000000123| (%010d)
00123      | (%-10.5d)
3.141500| (%f)
  3.141500| (%10f)
3.141500   | (%-10f)
003.141500| (%010f)
     3.142| (%10.3f)
```

Ví dụ 4.2b. Kiểu dữ liệu: đặc tả và đối số

vd4-2b.c

```c
#include <stdio.h>

int main() {
  printf("%d\n", 123);
  printf("%ld\n", 123l);
  printf("%f\n", 3.1415f);
  printf("%Lf\n", 3.1415L);

  // Lỗi không tương thích
  printf("%d\n", 123l);
  printf("%f\n", 3.1415L);
}
```

```
bangoc:$gcc -o prog vd4-2b.c
vd4-2b.c: In function ‘main’:
vd4-2b.c:14:12: warning: format ‘%d’ expects
argument of type ‘int’, but argument 2 has ty
pe ‘long int’ [-Wformat=]
   14 |   printf("%d\n", 123l);
      |            ~^     ~~~~
      |             |     |
      |             int   long int
      |            %ld
vd4-2b.c:15:12: warning: format ‘%f’ expects
argument of type ‘double’, but argument 2 has
 type ‘long double’ [-Wformat=]
   15 |   printf("%f\n", 3.1415L);
      |            ~^     ~~~~~~~
      |             |     |
      |             |     long double
      |             double
      |            %Lf
bangoc:$./prog
123
123
3.141500
3.141500
123
0.000000
```

Ví dụ 4-3. Số thứ tự tham số

"%1$*2$.*3$f"

- 1$ - Xuất giá trị của tham số thứ nhất;
- *2$ - Chiều rộng trường xuất bằng giá trị tham số thứ 2;
- .*3$ - Độ chính xác bằng giá trị tham số thứ 3;
- f - Số thực ở định dạng phổ thông.

vd4-3.c

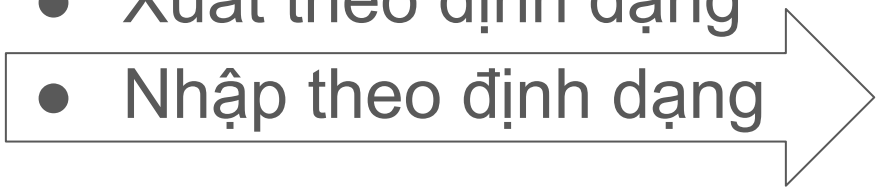
```
5  #include <stdio.h>
6
7  int main() {
8    int w = 10, p = 3;
9    double x = 3.1415;
10   printf("%1$*2$.*3$f\n", x, w, p);
11 }
```

Cho phép tùy chỉnh định dạng xuất ở thời gian thực thi

```
bangoc:$gcc -o prog vd4-3.c
bangoc:$./prog
     3.142
```

Nội dung

- Xuất theo định dạng
- Nhập theo định dạng



Nhập theo định dạng

Có thể nhập các giá trị dạng văn bản từ bàn phím (- luồng nhập tiêu chuẩn/stdin) theo định dạng với hàm scanf (thư viện stdio.h)

- Hàm scanf có nguyên mẫu giản lược:

```
int scanf (const char *format, ...);
```

- Các tham số:
  - format - Bố cục nhập;
    - Mô tả các giá trị trong luồng ký tự nhập, có thể chứa các đặc tả nhập cùng với các ký tự khác;
    - Về mặt cú pháp là chuỗi ký tự chỉ đọc;
    - Hàm scanf chuyển đổi các ký tự trong luồng nhập thành các giá trị theo các đặc tả nhập;
  - … - Các con trỏ tới các biến lưu giá trị nhập tương ứng với các đặc tả nhập (nếu có) trong bố cục nhập.
- Giá trị trả về: Số lượng giá trị đã đọc được hoặc EOF trong trường hợp chưa xử lý bất kỳ ký tự nhập nào, kiểu int.

scanf và printf

Cấu trúc của hàm scanf *có nhiều điểm tương đồng với hàm* printf *tuy nhiên ý nghĩa của các thành phần cụ thể có những khác biệt cơ bản.*

- Một số đặc tả nhập của scanf có thể bỏ qua các ký tự khoảng trắng (dấu cách, dấu tab, dấu xuống dòng, v.v., có thể kiểm tra bằng hàm isspace) số lượng không giới hạn.
  - Thiết kế này giúp cho việc nhập các giá trị có phần dễ hơn.
- Các tham số truyền cho scanf phải là ***các con trỏ*** tới các biến (do khi nhập hàm scanf phải lưu giá trị đọc được).
  - Hàm scanf lưu các giá trị vào các biến được trỏ tới bởi các tham số.

Bố cục nhập

- Đặc tả nhập được khớp với giá trị trong luồng ký tự nhập.
- Khi xử lý một dấu cách (hoặc bất kỳ ký tự trắng nào khác) trong bố cục nhập scanf bỏ qua các dấu khoảng trắng liên tiếp trong luồng ký tự nhập với số lượng bất kỳ.
- Các ký tự không phải khoảng trắng và không phải thành phần của đặc tả nhập phải được khớp chính xác với ký tự trong luồng nhập, hoặc sẽ phát sinh lỗi nhập.
  - *(Trong trường hợp phát sinh lỗi hàm scanf trả về ngay lập tức và lần gọi hàm tiếp theo sẽ bắt đầu đọc từ vị trí lỗi).*

Cấu trúc đặc tả nhập

`% các cờ` **`chiều rộng`** `định kiểu`

- *(Không có thành phần độ chính xác trong đặc tả nhập)*.
- Thành phần định kiểu là bắt buộc, các cờ và chiều rộng có thể có hoặc không.
- Các cờ:
  - ‘*’ - Khi xử lý đặc tả nhập có dấu ‘*” scanf đọc giá trị nhưng bỏ qua (không lưu) giá trị.
- Chiều rộng trường nhập:
  - Số nguyên - Đọc giá trị kết thúc khi đã đọc hết chiều rộng trường nhập (số lượng ký tự đã đọc == chiều rộng) hoặc đã đọc hết các ký tự trong giá trị (gặp một ký tự không hợp lệ).
- Định kiểu: Bao gồm ký tự định kiểu và có thể có các ký tự hiệu chỉnh độ dài (đứng trước ký tự định kiểu nếu có).

Ký tự định kiểu và định dạng mặc định

Đặc tả nhập đơn giản nhất có thể chỉ bao gồm dấu % và ký tự định kiểu (mặc định các thành phần còn lại)

Đặc tả nhập	**Kiểu tham số (con trỏ tới)**	**Ý nghĩa**
%d	int	Đọc 1 số nguyên có dấu HCS 10.
%i		Đọc 1 số nguyên được viết theo bất kỳ định dạng nào trong C.
%o	unsigned	Đọc 1 số nguyên không dấu HCS 8.
%u		Đọc 1 số nguyên không dấu HCS 10.
%x, %X		Đọc 1 số nguyên không dấu HCS 16.
%f, %F, %e, %E, %g, %G	float	Đọc 1 số thực dấu chấm động.
%n	int	Lưu số lượng ký tự đã đọc vào tham số.

Chuyển đổi các ký tự thành giá trị số

Định dạng giá trị số tương đương trong các trường hợp sau:

Đặc tả nhập	**Hàm chuyển đổi**
%d	strtol với base = 10
%i	strtol với base = 0
%o	strtoul với base = 8
%u	strtoul với base = 10
%x, %X	strtoul với base = 16
%a, %e, %f, %g, %A, %E, %F, %G	strtod

Các hiệu chỉnh độ dài

Các ký tự hiệu chỉnh độ dài được thêm vào nếu đối tượng lưu giá trị nhập có cùng biểu diễn như kiểu mặc định nhưng khác độ dài

- Các kiểu số nguyên:

Hiệu chỉnh	**Định kiểu: d, i, hoặc n (con trỏ tới)**	**u, o, x, hoặc X (con trỏ tới)**
hh (C99)	signed char	unsigned char
h	short	unsigned short
l	long	unsigned long
ll, L, q	long long	unsigned long long
z (C99)	size_t	size_t

- Đối với kiểu số thực:

Hiệu chỉnh	**Định kiểu: f, F, e, E, g, G (con trỏ tới)**
l	double
L	long double

Ví dụ 4.4a. Giá trị trả về của scanf

vd4-4a.c

```
3  #include <stdio.h>
4
5  int main() {
6    int a, b, c;
7    int n = scanf("%d%d%d", &a, &b, &c);
8    printf("n = %d\n", n);
9    printf("a = %d b = %d c = %d\n", a, b, c);
10 }
```

&a có giá trị là con trỏ tới a…

Nhập 3 số nguyên được ngăn cách bởi khoảng trắng, giá trị trả về của scanf được lưu trong n.

```
bangoc:$gcc -o prog vd4-4a.c
bangoc:$./prog
1 2 3
n = 3
a = 1 b = 2 c = 3
bangoc:$./prog
1 3.14 2
n = 2
a = 1 b = 3 c = 4096
bangoc:$./prog
n = -1
a = 100 b = 0 c = 4096
```

Ví dụ 4.4b. Kiểu dữ liệu và đặc tả

vd4-4b.c

```c
#include <stdio.h>

int main() {
  int i;
  long l;
  float f;
  double d;
  scanf("%d%ld%f%lf", &i, &l, &f, &d);
  printf("i = %d l = %ld ", i, l);
  printf("f = %f d = %f\n", f, d);

  // không tương thích kiểu
  scanf("%d%d%f%f", &i, &l, &f, &d);
  printf("i = %d l = %ld ", i, l);
  printf("f = %f d = %f\n", f, d);
}
```

Cảnh báo biên dịch

```
bangoc:$gcc -o prog vd4-4b.c
vd4-4b.c: In function ‘main’:
vd4-4b.c:15:13: warning: format ‘%d’ expects argument of type ‘int *’, but argument 3 has type ‘long int *’ [-Wformat=]
   15 |   scanf("%d%ld%f%f", &i, &l, &f, &d);
      |            ~^      ~~
      |             |      |
      |             int *  long int *
      |            %ld
vd4-4b.c:15:17: warning: format ‘%f’ expects argument of type ‘float *’, but argument 5 has type ‘double *’ [-Wformat=]
   15 |   scanf("%d%d%f%f", &i, &l, &f, &d);
      |              ~^         ~~
      |               |         |
      |               float *   double *
      |              %lf
bangoc:$./prog
1 2 3.1 3.2
i = 1 l = 2 f = 3.100000 d = 3.200000
3 4 5.1 5.2
i = 3 l = 4 f = 5.100000 d = 3.199999
```

Lưu ý: Với kiểu double, chúng ta sử dụng đặc tả %lf để nhập nhưng có thể xuất với đặc tả %f.

Ví dụ 4.5. Cờ, chiều rộng và các ký tự

vd4-5.c

```c
#include <stdio.h>
int main() {
  int a, b, cc;
  float x;
  int n = scanf("%2d%3d%*d%f%n", &a, &b, &x, &cc);
  printf("n = %d cc = %d\n", n, cc);
  printf("a = %d b = %d x = %f\n", a, b, x);
}
```

```
bangoc:$gcc -o prog vd4-5.c
bangoc:$./prog
12345 67 8.9
n = 3 cc = 12
a = 12 b = 345 x = 8.900000
```

Đọc **2 chữ số đầu tiên** của 1 số có 5 chữ số và lưu vào a, **3 chữ số tiếp theo** lưu vào b, bỏ qua số nguyên tiếp theo, lưu số thực tiếp theo vào x, lưu số lượng ký tự đã đọc vào cc.
với luồng ký tự đầu vào bao gồm 1 dòng
12345 67 8.9
hàm scanf trả về 3 (số lượng giá trị đã được đọc và được lưu)...

Ví dụ 4-6. Số thực, chiều rộng và độ chính xác

vd4-6.c

```c
#include <stdio.h>

int main() {
  int w, p;
  double v;
  printf("Giá trị số: ");
  scanf("%lf", &v);
  printf("Độ rộng & độ chính xác: ");
  scanf("%d%d", &w, &p);
  printf("%1$*2$.*3$f\n", v, w, p);
}
```

Nhập số thực và xuất giá trị với độ rộng và độ chính xác được nhập từ bàn phím

```
bangoc:$gcc -o prog vd4-6.c
bangoc:$./prog
Giá trị số: 3.1415
Độ rộng & độ chính xác: 10 6
  3.141500
```

Các khái niệm

- Đặc tả xuất - Output conversion specification;
- Đặc tả nhập - Input Conversion specification;
- Chuỗi bố cục xuất - Output Template string;
  - Tương đương với Chuỗi định dạng xuất - Output format string
- Chuỗi bố cục nhập - Input Template string;
  - Tương đương với Chuỗi định dạng nhập - Input format string

Bài tập 4.1

Nhập 2 số thực a, b là chiều dài và chiều rộng của 1 hình chữ nhật.

Xuất ra màn hình diện tích của hình chữ nhật làm tròn tới 2 chữ số phần thập phân.

Gợi ý: Có thể sử dụng đặc tả xuất %.2f