



Đầu vào và đầu ra trong 'C'

Phiên 4



Mục tiêu

- Để hiểu các hàm I/O được định dạng -
scanf() và printf()
- Để sử dụng các hàm I/O ký tự -
getchar() và putchar()



Đầu vào/Đầu ra chuẩn

- Trong C, thư viện chuẩn cung cấp các thói quen cho đầu vào và đầu ra
- Thư viện chuẩn có các hàm cho I/O xử lý đầu vào, đầu ra và thao tác ký tự và chuỗi
- Đầu vào chuẩn thường là bàn phím
- Đầu ra chuẩn thường là màn hình (còn gọi là bảng điều khiển)
- Đầu vào và đầu ra có thể được định tuyến lại từ hoặc đến các tệp thay vì các thiết bị tiêu chuẩn



Tập Tiêu đề <stdio.h>

- #bao gồm <stdio.h>
 - Đây là lệnh tiền xử lý
- **stdio.h** là một tập tin và được gọi là tập tin tiêu đề
- chứa các macro cho nhiều hàm đầu vào/đầu ra được sử dụng trong 'C'
- **printf(), scanf(), putchar(), getchar()** các hàm được thiết kế sao cho chúng yêu cầu các macro trong stdio.h để thực thi đúng



Đầu vào/Đầu ra được định dạng

- **inf()** – cho đầu ra được định dạng
- **quét()** – cho đầu vào được định dạng
- ***Định dạng chỉ định*** chỉ định định dạng mà các giá trị của biến sẽ được nhập vào và in ra



lệnh inf()-1

- được sử dụng để hiển thị dữ liệu trên đầu ra chuẩn – console

Cú pháp-printf (“chuỗi điều khiển”, danh sách đối số);

- Danh sách đối số bao gồm các hằng số, biến, biểu thức hoặc hàm được phân tách bằng dấu phẩy
- Phải có một lệnh định dạng trong chuỗi điều khiển cho mỗi đối số trong danh sách
- Các lệnh định dạng phải khớp với danh sách đối số về số lượng, loại và thứ tự
- Chuỗi điều khiển phải luôn được đặt trong dấu ngoặc kép, đó là dấu phân cách của nó



inf ()-2

Chuỗi điều khiển bao gồm một hoặc nhiều của ba loại mặt hàng:

1.Chữ nhân vật:

bao gồm các ký tự có thể in được

2.Lệnh định dạng:

bắt đầu bằng dấu % và theo sau là mã định dạng - phù hợp với mục dữ liệu

3.Ký tự không in được: Bao gồm các tab, khoảng trống và dòng mới

Mã định dạng-1

Định dạng	inf()	quét()
Ký tự đơn	%c	%c
Sợi dây	%S	%S
Số nguyên thập phân có dấu	%d	%d
Dấu chấm động (ký hiệu thập phân)	%f	%f hoặc %e
Dấu chấm động (ký hiệu thập phân)	%lf	%lf
Dấu chấm động (ký hiệu mũ)	%e	%f hoặc %e
Số thực (%f hoặc %e, tùy theo số nào ngắn hơn)	%g	
Số nguyên thập phân không dấu	%u	%u
Số nguyên thập lục phân không dấu (sử dụng "ABCDEF")	%x	%x
Số nguyên bát phân không dấu	%o	%o

Trong bảng trên c, d, f, lf, e, g, u, s, o và x là các chỉ định loại



Mã định dạng-2

Mã định dạng	Quy ước in ấn
%d	Số chữ số trong số nguyên
%f	Phần nguyên của số sẽ được in như vậy. Phần thập phân sẽ bao gồm 6 chữ số. Nếu phần thập phân của số nhỏ hơn 6, nó sẽ được đệm bằng số 0 ở bên phải, nếu không nó sẽ được làm tròn ở bên phải.
%e	Một chữ số bên trái dấu thập phân và 6 chữ số đến bên phải, như trong %f ở trên



Chuỗi điều khiển các ký tự đặc biệt

\\	để in \ tính cách
\ "	để in " tính cách
%%	để in % tính cách

chuỗi điều khiển & mã định dạng

KHÔNG	Các tuyên bố	Điều khiển Sợi dây	Kiểm soát cái gì chuỗi chứa	Ý nghĩa định sách	Giải thích của sự tranh luận định sách	Màn hình Trưng bày
1.	<code>printf("%d",300);</code>	<code>%d</code>	Bao gồm định dạng chỉ lệnh	300	Không thay đổi	300
2.	<code>printf("%d",10+5);</code>	<code>%d</code>	Bao gồm định dạng chỉ lệnh	10 + 5	Sự biểu lộ	15
3.	<code>printf("Chào buổi sáng ông Lee.");</code>	Tốt Buổi sáng Ông Lee.	Gồm có văn bản chỉ rõ ký tự	Không	Không	Tốt Chào buổi sáng ông. Lý.
4.	<code>int đếm = 100;</code> <code>printf("%d",đếm);</code>	<code>%d</code>	Bao gồm định dạng chỉ lệnh	đếm	biến đổi	100
5.	<code>printf("\nhello");</code>	\nhào	Bao gồm không in ký tự & văn bản nhân vật	Không	Không	xin chào trên một dòng mới
6.	# định nghĩa str "Good Apple" <code>printf("%s",chuỗi);</code>	<code>%S</code>	Bao gồm định dạng chỉ lệnh	Đường	Biểu tượng không thay đổi	Quả táo tốt
7. <code>int đếm số con;</code> <code>đếm=1;</code> <code>sud_nim=100;</code> <code>printf("%d %d\n",đếm, số</code> <code>lượng học sinh);</code>	<code>%d %d</code>	Bao gồm định dạng lệnh và trình tự thoát	đếm, Số lượng	hai biến	0, 100



Ví dụ cho printf()

Chương trình hiển thị số nguyên, số thực, số ký tự và chuỗi

bao gồm <stdio.h>

hàm main() không có giá trị

{

số nguyên a = 10;

float b = 24.67892345; char

ch = 'A';

printf("Dữ liệu số nguyên = %d", a); printf("Dữ liệu số
thực = %f", b); printf("Ký tự = %c", ch); printf("Lệnh
này in ra chuỗi"); printf("%s", "Lệnh này cũng in ra
một chuỗi");

}



Các trình sửa đổi trong printf()-1

1. '-' Bộ sửa đổi

Mục dữ liệu sẽ là *căn trái* trong trường của nó, mục sẽ được in bắt đầu từ vị trí ngoài cùng bên trái của trường đó.

2. Bộ điều chỉnh độ rộng trường

Có thể sử dụng với kiểu float, double hoặc mảng char (chuỗi). Bộ điều chỉnh độ rộng trường, là một số nguyên, định nghĩa , định nghĩa *tối thiểu* chiều rộng trường cho mục dữ liệu.



Các trình sửa đổi trong printf()-2

3. Bộ điều chỉnh độ chính xác

Bộ sửa đổi này có thể được sử dụng với kiểu float, double hoặc mảng char (chuỗi). Nếu được sử dụng với kiểu dữ liệu float hoặc double, chuỗi chữ số sẽ chỉ ra *tối đa* số chữ số được in bên phải số thập phân.

4. Bộ sửa đổi '0'

Mặc định phần đệm trong một trường được thực hiện bằng khoảng trắng. Nếu người dùng muốn đệm một trường bằng số không thì phải sử dụng trình sửa đổi này

5. 'l' Bổ nghĩa

Bộ sửa đổi này có thể được sử dụng để hiển thị số nguyên dưới dạng long int hoặc đối số có độ chính xác kép. Mã định dạng tương ứng là %ld



Các trình sửa đổi trong printf()-3

6. Trình sửa đổi 'h'

Bộ sửa đổi này được sử dụng để hiển thị các số nguyên ngắn. Mã định dạng tương ứng là %hd

7. '*' Bộ sửa đổi

Nếu người dùng không muốn chỉ định trước độ rộng trường nhưng muốn chương trình chỉ định thì sử dụng trình sửa đổi này



Ví dụ cho các trình sửa đổi

/* Chương trình này trình bày cách sử dụng Modifiers trong printf() */

bao gồm <stdio.h>

hàm main() không có giá trị

{

printf("Số 555 ở nhiều dạng khác nhau:\n");

printf("Không có bất kỳ ký tự sửa đổi nào: \n");

printf("[%d]\n",555);

printf("Với - sửa đổi :\n"); printf("[%%-d]\n",555);

printf("Với chuỗi số 10 làm ký tự sửa đổi :\n");

printf("[%10d]\n",555);

printf("Với 0 là số sửa đổi: \n");

printf("[%0d]\n",555);

printf("Với 0 và chuỗi chữ số 10 làm số sửa đổi :\n"); printf("[%010d]\n",555);

printf("Với -, 0 và chuỗi chữ số 10 làm ký tự sửa đổi: \n");

printf("[%010d]\n",555);

}



quét()

- Được sử dụng để chấp nhận dữ liệu

Định dạng chung của hàm scanf()

quét chuỗi điều khiển", danh sách đối số);

- Định dạng được sử dụng trong câu lệnh printf() cũng được sử dụng với cú pháp tương tự trong câu lệnh scanf()



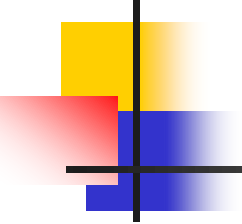
Sự khác biệt trong danh sách đối số giữa printf() và scanf()

- printf() sử dụng tên biến, hằng số, hằng số tượng trưng và biểu thức
- scanf() sử dụng con trỏ tới các biến

Khi sử dụng scanf() hãy tuân theo các quy tắc sau đối với danh sách đối số:

- Nếu bạn muốn đọc giá trị của một biến có kiểu dữ liệu cơ bản, hãy thêm dấu phẩy trước tên biến **&** biểu tượng
- Khi đọc giá trị của một biến có kiểu dữ liệu phái sinh, không sử dụng **&** trước tên biến

Sự khác biệt trong các lệnh định dạng của printf() và scanf()



- Không có tùy chọn %g
- Mã định dạng %f và %e có hiệu lực giống nhau



Ví dụ cho scanf()

bao gồm <stdio.h>

hàm main() không có giá trị

{

số nguyên a;

phao d;

char ch, tên[40];

printf("Vui lòng nhập dữ liệu\n"); scanf("%d %f %c %s", &a, &d, &ch, name);

printf("\n Các giá trị được chấp nhận là:
%d, %f, %c, %s", a, d, ch, tên);

}



Bộ đệm I/O

- được sử dụng để đọc và ghi các ký tự ASCII

MỘT **đệm** là một vùng lưu trữ tạm thời, trong bộ nhớ, hoặc trên thẻ điều khiển cho thiết bị

I/O đệm có thể được chia nhỏ thành:

- Giao diện điều khiển I/O
- **Tập đệm I/O**



Giao diện điều khiển I/O

- Các chức năng I/O của bảng điều khiển hướng các hoạt động của chúng tới đầu vào và đầu ra chuẩn của hệ thống

Trong 'C' đơn giản nhất chức năng I/O của bảng điều khiển là:

- **lấy char()** – đọc một (và chỉ một) ký tự từ bàn phím
- **putchar()** – mà đầu ra một ký tự duy nhất trên màn hình



lấy char()

- Được sử dụng để đọc dữ liệu đầu vào, từng ký tự một từ bàn phím
- Bộ đệm các ký tự cho đến khi người dùng nhập ký tự trả về
- hàm `getchar()` không có đối số, nhưng dấu ngoặc đơn vẫn phải có



Ví dụ cho getchar()

```
/* Chương trình để chứng minh cách sử dụng getchar() */  
# bao gồm <stdio.h>
```

hàm main() không có giá trị

```
{  
    chữ cái char;  
    printf("\nVui lòng nhập bất kỳ ký tự nào : "); letter =  
    getchar();  
    printf("\nKý tự bạn nhập là %c", letter);  
}
```




putchar()

- Hàm xuất ký tự trong 'C'
- đòi hỏi một lập luận

Đối số của hàm putchar() có thể là:

- Hằng số ký tự đơn
- Một chuỗi thoát hiểm
- Một biến ký tự



tùy chọn và hiệu ứng putchar()

Lý lẽ	Chức năng	Tác dụng
biến ký tự	putchar(c)	Hiển thị nội dung của biến ký tự c
hằng số ký tự	putchar('A')	Hiển thị chữ A
hằng số số	putchar('5')	Hiển thị chữ số 5
trình tự thoát	putchar('\t')	Chèn một ký tự khoảng cách tab vào vị trí con trỏ
trình tự thoát	putchar('\n')	Chèn một ký tự trả về tại vị trí con trỏ



putchar()

/* Chương trình này trình bày cách sử dụng hằng số và chuỗi thoát trong putchar() */

bao gồm <stdio.h>

hàm main() không có giá trị

{

đặt('H'); đặt('\n'); đặt('\t');

putchar('E'); putchar('\n'); putchar('\t'); putchar('\t');

putchar('L'); putchar('\n'); putchar('\t'); putchar('\t');

putchar('\t'); putchar('L'); putchar('\n');

đặt char('\t'); đặt char('\t'); đặt char('\t'); đặt char('\t');

putchar('O');

}

Ví dụ