嵌入式 Linux C 语言程序设计基础教程

作者：华清远见

第 3 章 数据的输入输出

本章目标

在上一章中，读者了解了嵌入式 Linux C 语言的数据相关的知识，包括数据类型、变量和常量等。本章继续介绍 C 语言中与数据有关的知识，数据的输入输出。通过本章的学习，读者将会掌握如下内容：

- 字符输出函数 putchar
- 格式化输出函数 printf
- 字符输入函数 getchar
- 格式化输入函数 scanf
- 字符串输入输出函数

3.1 数据的输出

在这里我们讨论的数据的输出，指的是如何把数据显示到标准输出，即显示器上。至于，如何把数据输出到文件中或者写到数据库中，不在本书的讨论范围内，读者可以参考“嵌入式 Linux 应用”方面的书。

首先，我们介绍一下字符型数据的输出，在 C 库中有专门的字符输出函数 getchar。

3.1.1 字符输出函数 putchar

头文件：stdio.h
函数原型：int putchar(int c)
函数参数：c 为字符常量或表达式
函数返回值：输出的字符
函数功能：在标准输出上显示一个字符

示例代码如下：

```
#include <stdio.h>

int main()
{
    int a = 65;
    char b = 'B';

    putchar(a);
    putchar('\n');
    putchar(b);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$ gcc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
A
B
```

3.1.2 格式化输出函数 printf

putchar 函数只能在终端输出一个字符型的数据，如果期望在终端输出若干个数据，且为任意类型，可以用 printf 函数。

头文件：stdio.h
函数原型：int printf (const char *format, …)
函数参数：format 指定输出格式，后面跟要输出的变量，为不定参，用“…”代表。
函数返回值：成功返回输出的字节数，失败返回-1（EOF）。
函数功能：格式化字符串输出
表 3-1 显示了目前 printf 支持的格式符。

表 3-1 printf 支持的格式

格 式 符	作 用	格 式 符	作 用
i, d	十进制整数	s	字符串
x, X	十六进制无符号整数	e, E	指数形式浮点小数
o	八进制无符号整数	f	小数形式浮点小数
u	无符号十进制整数	g	e 和 f 中较短一种
c	单一字符	%%	百分号本身

一个格式说明可以带有几个修饰符，用来指定显示宽度，小数尾数及左对齐等。请参照表 3-2。

表 3-2 printf 支持的格式说明符的修饰符

修 饰 符	功 能
m	输出数据域宽，数据长度<m，左补空格；否则按实际输出
.n	对实数，指定小数点后位数（四舍五入）
	对字符串，指定实际输出位数
-	输出数据在域内左对齐（缺省右对齐）
+	指定在有符号数的正数前显示正号(+)
0	输出数值时指定左面不使用的空位置自动填 0
#	在八进制和十六进制数前显示前导 0，0x
l	在 d, o, x, u 前，指定输出精度为 long 型
	在 e, f, g 前，指定输出精度为 double 型

示例代码如下：

```
#include <stdio.h>

int main()
{
    int a = 1234;
    float f = 123.456;
    char ch = 'a';
    char s[] = "Hello world!";

    printf("%8d,%2d\n", a, a);
    printf("%f,%8f,%8.1f,%.2f\n", f, f, f, f);
    printf("%3c\n", ch);
    printf("%s\n%15s\n%10.5s\n%2.5s\n%.3s\n", s, s, s, s, s);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$ gcc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
    1234,1234
123.456001,123.456001,    123.5,123.46
  a
Hello world!
```

```
   Hello world!
     Hello
Hello
Hel
```

在这个程序中，对于整数的输出，使用的是格式符%8d 和%2d。在这个例子中整数 a 的值为 1234，有 4 位，用%8d 输出，不够 8 位，左补 4 个空格；用%2d 输出，变量 a 本身就超过了 2 位，2 不起作用。

对于程序中的浮点数 f，f 的值为 123.456，有 7 位，用%f 输出，小数点后有 6 位；用%8f 输出，对小数点后的位数没有限制，还是规定的 6 位，加上整数部分和小数点，数值就有 10 位（123.456001）超过了 8，相当于 8 不起作用；用%8.1f 输出，要求小数点后有 1 位（四舍五入），总共 8 位，因此 123.5 左边补充了 3 个空格。；用%.2f 输出，限制了小数点后有 2 位小数，但是，对数据的总位数没有限制，因此输出了 123.46。

下面举个例子，演示一下格式符“-”的用法：

```
#include <stdio.h>

int main()
{
    int a = 1234;
    float f = 123.456;
    char s[] = "Hello world!";

    printf("%8d,%-8d\n", a, a);
    printf("%10.2f,%-10.1f\n", f, f);
    printf("%10.5s,%-10.3s\n", s, s);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
    1234,1234
    123.46,123.5
     Hello,Hel
```

在程序中，整数 a 为 1234，有 4 位，用%8d 输出，左补 4 个空格，若加了“-”修饰，用%-8d 输出，左对齐，右补 4 个空格。关于浮点数输出，加“-”修饰，也是改变的对齐方式。对于字符串 s（“Hello world!”），用%10.5s 输出，实际输出 5 个字符 Hello，共输出 10 个，因此左补 5 个空格，用%-10.3s 输出，实际输出 3 个字符 Hel，右补 7 个空格。

下面举个例子，演示一下格式符“0”、“+”、“#”的用法：

```
#include <stdio.h>

int main()
{
    int a = 1234;
    float f = 123.456;

    printf("%08d\n", a);
    printf("%010.2f\n", f);
```

```
    printf("%0+8d\n", a);
    printf("%0+10.2f\n", f);

    a = 123;
    printf ("%o,%#o,%X,%#X\n", a, a, a, a);
    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$ gcc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
00001234
0000123.46
+0001234
+000123.46
173,0173,7B,0X7B
```

在此程序中，格式符“%08d”和“%010.2f”，“0”起的作用是，左面不使用的空位置自动填 0。格式符“%0+8d”和“%0+10.2f”，“+”起的作用是，正数前面显示“+”号。

3.2 数据的输入

前文已经介绍了数据输出，现在继续介绍数据输入。这里所说的数据输入，是指如何得到从键盘上输入的数据。关于如何读取文件中的数据或者读取数据库中的数据，不在本书的讨论范围，读者可以参考“嵌入式 Linux 应用”方面的书。

3.2.1 字符输入函数 getchar

首先，我们介绍一下字符型数据的输入，在 C 库中有专门的字符输入函数。

头文件：stdio.h

函数原型：int getchar(void)

函数参数：无

函数返回值：成功，返回读到的字符，失败或读到结束符返回 EOF(-1)

函数功能：在键盘上读一个字符

提到字符型数据，就必须要熟悉 ASCII 表。在计算机中，所有的数据在存储和运算时都要使用二进制数表示（因为计算机只能识别 0 和 1）。像 a、b、A、B 这样的英文字母、以及 0、1、2 等数字，还有一些常用的符号（例如*、#、@等）在计算机中存储时都要使用二进制数来表示。关于具体用哪个数字表示哪个符号，就是编码问题，大家的程序若想互相通信，必须遵照相同的规则。于是，美国国家标准学会(American National Standard Institute, ANSI)制定了美国标准信息交换代码，即 ASCII 编码（ASCII 表）。

ASCII 表中共 256 个字符，ASCII 码值从 0 到 255。

getchar 函数返回值的含义是存储从键盘上读取的字符，返回值的类型确是 int，很多人不理解，认为返回值应该是 char 类型。char 类型的全称是 signed char 型，表示范围-128～127，而字符的 ASCII 码的范围是 0～255，显然，char 型的范围太小了，不能存储所有的值。若返回值的类型是 unsigned char，范围恰好为 0～255。那为什么 getchar 函数的返回值的类型不是 unsigned char 呢？读者需要想到一种特殊情况，即 getchar 函数失败或读到了结束符，用-1 表示，unsigned char 不能存储负数。因此，返回值的类型是 int，范围够大，还可以存储-1。

示例代码如下：

```
#include <stdio.h>

int main()
{
    int ch;
    printf("Enter a character:");

    while ((ch = getchar()) != EOF)
        printf ("%c--->%#x\n", ch, ch);

    printf ("end main\n");

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
Enter a character:a
a--->0x61

--->0xa
b
b--->0x62

--->0xa
end main
```

可以看出，输入字符 a 时，第一次 getchar 函数读到了字符 a，第二次 getchar 函数，读到\n。按 ctrl+d 键，getchar 函数返回 EOF，输入结束，程序退出。

3.2.2 格式化输入函数 scanf

getchar 函数只能从键盘读到一个字符型的数据，如果期望读到若干个数据，且为任意类型，可以用 scanf 函数。

头文件：stdio.h

函数原型：int scanf (const char *format, …)

函数参数：format 指定输入格式，后面跟要输入的变量的地址表，为不定参，用”…”代表。

函数返回值：成功返回输入的变量的个数，失败返回-1（EOF）。

函数功能：按指定格式从键盘读入数据，存入地址表指定存储单元中，并按回车键结束。

目前，scanf 支持的格式字符很多，详情参照表 3-3。

表 3-3 scanf 函数支持的格式说明符

i, d	十进制整数	c	单一字符
x, X	十六进制无符号整数	s	字符串
o	八进制无符号整数	e	指数形式浮点小数
u	无符号十进制整数	f	小数形式浮点小数

表 3-4 列车了 scanf 函数支持的格式说明符可以带的修饰符。

表 3-4　　scanf 函数支持的格式说明符的修饰符

修　饰　符	功　　能
h	用于 d，o，x 前，指定输入为 short 型整数
l	用于 d，o，x 前，指定输入为 long 型整数
	用于 e，f 前，指定输入为 double 型实数
m	指定输入数据宽度，遇空格或不可转换字符结束
*	抑制符，指定输入项读入后不赋给变量

示例代码如下：

```
#include <stdio.h>

int main()
{
    int a, b, c;
    printf ("input a b c:");

    scanf("%d", &a);
    scanf("%x", &b);
    scanf("%o", &c);

    printf("a=%d, b=%d, c=%d\n", a, b, c);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input a b c:15 15 15
a=15, b=21, c=13
```

可以看出，当输入的格式说明是%x 时，输入的数字被看做十六进制数，十六进制的 15 就是十进制的 21。当输入的格式说明是%o 时，输入的数字被看做八进制数，八进制的 15 就是十进制的 13。

下面再看一个格式说明符修饰符的例子：

```
#include <stdio.h>

int main()
{
    int yy, mm, dd;
    int a;
    float c;

    printf ("input year month day:");
    scanf ("%4d%2d%2d", &yy, &mm, &dd);
    printf("%d-%d-%d\n", yy, mm, dd);
```

```
    printf ("input  int float:");
    scanf ("%3d%*4d%f", &a, &c);
    printf("a=%d, c=%f\n", a, c);

    return 0;
}
```

在此程序中，需要输入多个变量，就涉及了怎么去分隔输入值的问题。输入分隔符的指定：一般以空格、TAB 或回车键作为分隔符。

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input year month day:2012 12 26
2012-12-26
input int float:9 9.1234
a=9, c=0.123400
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input year month day:2012 12 26
2012-12-26
input int float:1234567890
a=123, c=890.000000
```

在上面程序中，%*4d 比较特殊，*是抑制符，4d 指定输入项中 4 个数字读入后不赋给变量。

关于 scanf 函数有一些特别需要注意的地方：

① 用“%c”格式符时，空格和转义字符作为有效字符输入

示例代码如下：

```
#include <stdio.h>

int main()
{
    char ch1, ch2, ch3;
    printf ("input three characters:");

    scanf ("%c%c%c", &ch1, &ch2, &ch3);
    printf ("ch1=%c, ch2=%c, ch3=%c\n", ch1, ch2, ch3);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input three characters:a b c
ch1=a, ch2= , ch3=b
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input three characters:a\tb
ch1=a, ch2=\, ch3=t
```

② 输入数据时，遇到以下情况认为该数据结束：

- 空格、TAB 或回车
- 宽度结束
- 非法输入

关于非法输入，比如：程序需要输入一个浮点数，用户输入的是字母，这就属性非法输入。读者可以通过下面的示例，来深入理解：

```
#include <stdio.h>

int main()
{
    char ch1, ch2, ch3;
    printf("input three characters:");

    scanf("%c%c%c", &ch1, &ch2, &ch3);
    printf("ch1=%c, ch2=%c, ch3=%c\n", ch1, ch2, ch3);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input three characters:1234 w 123.y2
a=1234, b= , c=0.000000
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input three characters:1234w34.x6
a=1234, b=w, c=34.000000
```

③ scanf 函数返回值是成功输入的变量的个数，当遇到非法输入时，返回值会小于实际变量个数。示例程序如下：

```
#include <stdio.h>

int main()
{
    int a, b, n;

    printf("input numbers:");
    while ((n = scanf("%d%d", &a, &b)) == 2)
    {
        printf("a=%d, b=%d\n", a, b);
        printf("input numbers:");
    }
    printf("n=%d\n", n);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
```

```
input numbers:9 5
a=9, b=5
input numbers:3 6
a=3, b=6
input numbers:1 q
n=1
```

可以看出，需要输入 2 个整数，当正常输入时，scanf 函数返回 2。若有一个变量是非法输入（输入字母）时，返回值为 1。我们经常利用 scanf 的返回值来构造循环。

④ 使用输入函数可能会留下垃圾，请看下面的程序：

```
#include <stdio.h>

int main()
{
    int a;
    char ch;

    printf("input a number:");
    scanf("%d", &a);
    printf("a=%d\n", a);

    printf("input a character:");
    scanf("%c", &ch);
    printf("ch=%c %d\n", ch, ch);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input a number:9
a=9
input a character:ch=
 10
```

在这个程序中，当输入了一个数字后，换行符\n 还在缓冲区中，接下来程序需要输入一个字符型变量时，并没有停顿，让用户输入，而是直接把残留的\n 取走了。所以，这个程序中换行符就是垃圾字符。解决这个问题，有两个办法。

第一、调用 getchar 函数，清除垃圾字符

```
#include <stdio.h>

int main()
{
    int a;
    char ch;

    printf("input a number:");
    scanf("%d", &a);
    printf("a=%d\n", a);

   getchar();
```

```
    printf("input a character:");
    scanf("%c", &ch);
    printf("ch=%c %d\n", ch, ch);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input a number:9
a=9
input a character:a
ch=a 97
```

第二、用格式串中空格或“%*c”来“吃掉”

```
int main()
{
    int a;
    char ch;

    printf("input a number:");
    scanf("%d", &a);
    printf("a=%d\n", a);

   printf("input a character:");
    scanf("%*c%c", &ch);
    printf("ch=%c %d\n", ch, ch);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
input a number:9
a=9
input a character:a
ch=a 97
```

3.3 数据输入输出综合示例

问题 1：输入三角形的三条边长 a，b，c，计算三角形的面积 area。计算公式如下：

$$s=(a+b+c)/2$$

$$area=\sqrt{s\times(s-a)\times(s-b)\times(s-c)}$$

示例程序：

```
#include <math.h>
#include <stdio.h>
```

```
int main()
{
    float a, b, c, s, area;

    scanf("%f%f%f", &a, &b, &c);
    s = 1.0 / 2 * (a + b + c);
    area = sqrt(s * (s-a) * (s-b) * (s-c));

    printf("a=%7.2f,b=%7.2f,c=%7.2f\n", a, b, c);
    printf("area=%7.2f\n", area);

    return 0;
}
```

注意这个程序，用到了数学函数库中的函数 sqrt，编译程序时，要加链接选项-lm。

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$ gcc area.c  -o area  -lm -Wall
linux@ubuntu:~/book/ch3$./area
3 4 6
a=   3.00,b=   4.00,c=   6.00
area=   5.33
```

问题 2：从键盘输入 a、b、c 的值，求一元二次方程 $y=ax^2+b$ 的根，计算公式如下：

$$x1=\frac{-b+\sqrt{b^2-4ac}}{2a}$$

$$x2=\frac{-b-\sqrt{b^2-4ac}}{2a}$$

示例程序：

```
#include <math.h>
#include <stdio.h>

int main()
{
    float a, b, c, disc, x1, x2, p, q;

    scanf("%f%f%f", &a, &b, &c);
    disc = b*b - 4*a*c;
    p = -b/(2*a);
    q = sqrt(disc)/(2*a);

    x1 = p + q;
    x2 = p - q;

    printf("\n\nx1=%5.2f\nx2=%5.2f\n", x1, x2);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc equation.c  -o equation  -lm -Wall
linux@ubuntu:~/book/ch3$./equation
1 3 2
```

```
x1=-1.00
x2=-2.00
```

3.4 字符串输入输出函数

关于字符串的输入输出，除了用 scanf 函数和 printf 函数外，C 库中还提供了专门的字符串处理函数：

① 字符串输出函数 puts

头文件：stdio.h

函数原型：int puts(const char *s)

功能：在标准输出上显示字符串 s

参数：s 为需要输出的字符串。

返回值：成功返回一个非 0 的数字；失败返回-1 或 EOF。

示例程序如下：

```
#include <stdio.h>

int main()
{
char s[]="welcome";
  puts(s);

    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c  -o test -Wall
linux@ubuntu:~/book/ch3$./test
welcome
```

可以看出，puts 函数在输出字符串时，会自动追加换行符'\n'。使用时，注意字符数组必须以'\0'结束。

② 字符串输入函数 gets

头文件：stdio.h

函数原型：char *gets(char *s)

功能：从键盘输入一以回车结束的字符串放入字符数组中，并自动加'\0'

参数：s 为字符数组，存储输入的字符串

返回值：成功返回字符数组的起始地址，失败或输入结束返回 NULL。

示例程序如下：

```
#include <stdio.h>

#define N 20

int main()
{
    int i = 0;
    char s[N] = {0};

    printf(">");
```

```
    while (gets(s) != NULL)
    {
        printf(">");
       // scanf("%s", s);
        printf("i = %d :%s\n", i, s);
        i++;
    }

    printf("end main\n");
    return 0;
}
```

程序执行结果如下：

```
linux@ubuntu:~/book/ch3$cc test.c -Wall
linux@ubuntu:~/book/ch3$./a.out
>how are you
>i = 0: how are you
aa bb cc
>i = 1 :aa bb cc
end main
```

编译这个程序时，出现了下面的警告：

```
warning: the 'gets' function is dangerous and should not be used.
```

gets 函数的参数中，不含长度控制。当输入字符时，最多只能输入 N-1（留一个位置存字符串的结束符\0）。假如输入字符超过了 N-1，则多余的符号也会被存到字符数组中，这样就会造成内存的访问越界，结果是不可预料的。所以，使用此函数时，会有警告。我们在使用该函数时，一定要注意数组的长度。

还可以看出，gets 函数并不以空格作为字符串输入结束的标志，而只以回车作为输入结束。这是与 scanf 函数不同的。

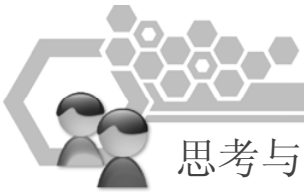
小结

本章也是嵌入式 Linux C 语言中很基础，必须要熟练掌握的一章。

首先，介绍了只能输出字符型数据的函数 putchar，然后介绍了格式化输出函数 printf，重点介绍了该函数的格式符。

接下来继续介绍了数据输入，包含字符输入函数 getchar 和格式化输入函数 scanf，重点介绍了 scanf 函数的格式符及清除垃圾符号等问题。

最后介绍了字符串输入输出函数。



思考与练习

1. 输入下面的程序，运行出结果。

```
int main(int argc,char **argv)
{
        char  c1, c2;
        c1=97;
```

```
        c2=98;
        printf(“%c %c\n”, c1, c2);

       return 0;
}
```

2．分析下程序，写出运行结果，再输入计算机运行，将得到的结果与你分析得到的结果比较对照。

```
int main(int argc,char **argv)
{
    char c1 = ’ a’ , c2 = ’ b’ , c3 =’ c’ , c4=’ \101’ , c5=’ \116’ ;

    printf(“a%c b%c\tabc\n”,c1,c2,c3);
    printf(“\t\b%c %c”,c4,c5);

    return 0;
}
```

3．分析下程序，写出运行结果，再输入计算机运行，将得到的结果与你分析得到的结果比较对照。

```
int main(int argc,char **argv)
{
    int i , j , m , n ;

   i=8;
   j=10;
   m=++i;
   n=j++;

   printf(“%d,%d,%d,%d\n”,i,j,m,n);
  return 0;
}
```

联系方式

集团官网：www.hqyj.com　　嵌入式学院：www.embedu.org　　移动互联网学院：www.3g-edu.org

企业学院：www.farsight.com.cn　　物联网学院：www.topsight.cn　　研发中心：dev.hqyj.com

集团总部地址：北京市海淀区西三旗悦秀路北京明园大学校内 华清远见教育集团

全国免费咨询电话：400-706-1880

双休日及节假日请致电值班手机：15010390966

在线咨询：张老师 QQ（619366077），王老师 QQ（2814652411），杨老师 QQ（1462495461）

企业培训洽谈专线：010-82600901

院校合作洽谈专线：010-82600350，在线咨询：QQ（248856300）