

Entrada y Salida básicas - Procedimientos de Entrada y salida

A diferencia de muchos lenguajes de programación, C no contiene instrucciones de entrada ni de salida. C es un lenguaje transportable en extremo, lo que significa que un programa de C compilado y ejecutado en una computadora puede compilarse y ejecutarse también en otra computadora. La mayoría de las incompatibilidades entre las distintas computadoras reside en su manera de efectuar su E/S (entrada/salida). Cada dispositivo diferente requiere de un método distinto de realizar su E/S. Cuando los diseñadores de C colocaron todas las prestaciones de E/S en funciones comunes a todas ellas, suministradas con el compilador de cada computadora -y no en sentencias de C- se propusieron asegurar que los programas no quedasen rígidamente ligados, en lo que refiere a su capacidad de entrada y salida de datos, a ningún hardware específico. Los compiladores que vienen con los lenguajes de programación orientados a Windows, tales como el de Visual C++, ofrecen funciones específicas que imitan y se basan sobre muchas de las funciones básicas de E/S que se trabajarán en este curso.

Dispositivos estándar

El lenguaje C siempre interpreta que la entrada viene de `stdin`, término obtenido de la abreviatura `standard input device` o dispositivo estándar de entrada. Este es generalmente el teclado, aunque puede redirigirlo hacia otro dispositivo, tal como un archivo de disco. C también considera que toda la salida va a `stdout`, o `standard output device` o dispositivo estándar de salida, cuya opción predeterminada es el monitor, aunque puede ser redirigido hacia la impresora, un archivo de disco, etc.

Función `printf()`: Permite la presentación de valores numéricos, caracteres y cadenas de texto por el archivo estándar de salida (pantalla). El prototipo de la función `printf` es el siguiente:

`printf(control,arg1,arg2...);`

Donde:

control : es la cadena de control en la cual se indica la forma en que se mostrarán los argumentos posteriores (si los hubiere). También se puede escribir una cadena de texto (sin necesidad de argumentos), o combinar ambas posibilidades, así como secuencias de escape.

arg_i : argumentos cuyos valores habrán de ser mostrados en la línea de salida. Si se utilizan argumentos se debe indicar en la cadena de control tantos caracteres de conversión (o modificadores) como argumentos se van a presentar.

El modificador o carácter de conversión está compuesto por el carácter `%` seguido por un carácter , que indica de que tipo de dato se trata.

Listado de los modificadores o caracteres de conversión más utilizados

%c	Un único carácter
%d	Un entero con signo, en base decimal
%u	Un entero sin signo, en base decimal
%o	Un entero en base octal
%x	Un entero en base hexadecimal
%e	Un número real en coma flotante, con exponente
%f	Un número real en coma flotante, sin exponente
%s	Una cadena de caracteres
%p	Un puntero o dirección de memoria

El formato completo de los modificadores:

% [signo] [longitud] [.precisión] [I/L] conversión

Entre el % y el carácter de conversión puede haber:

signo:

- : indica si el valor se ajustará a la izquierda.
- + : establece que el número siempre será impreso con signo.

longitud: especifica un ancho mínimo de campo. El argumento será impreso en por lo menos esta longitud. Si tiene menos caracteres que el ancho del campo, será rellenado a la izquierda. El carácter de relleno normalmente es espacio.

precisión: indica el número máximo de decimales que tendrá el valor.

l/L: utilizamos l cuando se trata de una variable de tipo long y L cuando es de tipo double

Función scanf(): Permite ingresar la información o datos en posiciones de memoria de la computadora a través del archivo estándar de entrada (teclado). El prototipo de la función scanf es el siguiente:

scanf(control, arg1, arg2...);

Donde:

control : es la cadena de control que indica, por regla general, los modificadores que harán referencia al tipo de dato de los argumentos. Los modificadores son los del listado previo.

arg_i : son los nombres o identificadores de los argumentos de entrada cuyos valores se incorporarán por teclado.

La principal característica de scanf es que "necesita conocer" la posición de la memoria en que se encuentra la variable para poder almacenar la información ingresada. Para indicarle esta posición se utiliza un operador de puntero, el operador unario de dirección: & (ampersand), que se coloca delante del nombre de cada variable.

Ejemplos , un poco de reflexión y de ejercitación

```
p1.  :  
      :  
      int longi = 25 ;  
      printf ( "\n longi = %-3d \n", longi);           // ¿cómo es la salida que produce esta línea?
```

```
p2.  :  
      :  
      float prom = 1258.32;  
      printf ( "longi=%-5d\t\nprom = %5.2f\n", longi, prom); // ¿y esta otra?
```

```
s1. scanf("%d %d %d", &var1, &var2, &var3);
```

Si Ud. no indica a scanf() cual es la dirección interna (la posición de memoria) de estas variables, el programa en cuestión no podría acceder luego a los valores tipeados desde el teclado.

Trabajo para Ud.

Edite un pequeño programa de prueba que tenga las proposiciones de los ejemplos. Identifique en las líneas printf() cada uno de los modificadores, cambie los valores de los ejemplos y compare las diferencias

Amplíe gradualmente su programa de prueba, incorpore al mismo las variables del ejemplo **s1** y pruebe leer y escribir distintos valores de los datos.

Bibliografía

Kernighan, B. W. y Ritchie, D. M. 2001 *El lenguaje de programación C*, Prentice Hall, México.

Perry G. 2000 *C con ejemplos*, Pearson Education SA, Perú.