hacen que el archivo fuente sea más grande, pero por lo general esto no tiene importancia. Resumiendo, se deben usar comentarios y espacios en blanco para que sea fácil, en la medida de lo posible, la comprensión y el mantenimiento del código fuente.

2. ¿Cuál es la diferencia entre un enunciado y un bloque?

   Un bloque es un grupo de enunciados encerrados dentro de llaves ({}). Un bloque puede ser usado en muchos lugares donde puede ser usado un enunciado.

3. ¿Cómo se sabe cuáles funciones de biblioteca están disponibles?

   La mayoría de los compiladores vienen con un manual dedicado específicamente a la documentación de las funciones de biblioteca. Por lo general, vienen en orden alfabético. Otra manera de conocer las funciones de biblioteca disponibles es comprar un libro que las liste. El apéndice E, "Prototipos de función y archivos de encabezado", y el apéndice F, "Funciones comunes en orden alfabético", listan las funciones por categoría y, desde luego, en orden alfabético, respectivamente. Después de que comience a entender más del C, es buena idea leer estos apéndices para que no reescriba una función de biblioteca. (¡No vuelva a inventar el hilo negro!)

Taller

El taller proporciona un cuestionario que le ayudará a reafirmar su comprensión del material tratado así como ejercicios para darle experiencia en el uso de lo que ha aprendido.

Cuestionario

1. ¿Cómo se llama a un grupo de uno o más enunciados del C encerrados entre llaves?
2. ¿Cuál es el único componente obligatorio de todo programa en C?
3. ¿Cómo se añaden comentarios al programa y para qué se usan?
4. ¿Qué es una función?
5. El C proporciona dos tipos de funciones. ¿Qué son y cómo se diferencian?
6. ¿Para qué se usa la directiva `#include`?
7. ¿Se pueden anidar los comentarios?
8. ¿Los comentarios pueden ser más grandes que una línea?
9. ¿Qué otro nombre se le da a los archivos de inclusión?

10. ¿Qué es un archivo de inclusión?

Ejercicios

1. Escriba el programa más pequeño posible.
2. Usando el siguiente programa, conteste las preguntas:
   a. ¿Qué líneas contienen enunciados?
   b. ¿Qué líneas contienen definiciones de variables?
   c. ¿Qué líneas contienen prototipos de función?
   d. ¿Qué líneas contienen definiciones de función?
   e. ¿Qué líneas contienen comentarios?

```
1: /* EX2-2.C */
2: #include <stdio.h>
3:
4: void display_line(void);
5:
6: main()
7: {
8:     display_line();
9:     printf("\n Teach Yourself C In 21 Days!\n");
10:     display_line();
11:
12:     return 0;
13: }
14:
15: /* Imprime una línea de asteriscos */
16: void display_line(void)
17: {
18:     int counter;
19:
20:     for( counter = 0; counter < 21; counter++ )
21:         printf("*" );
22: }
23: /* Fin del programa */
```

3. Escriba un ejemplo de un comentario.

4. ¿Qué hace el siguiente programa? (Tecléelo, compílelo y ejecútelo.)

```
1: /* EX2-4.C */
2: #include <stdio.h>
3:
4: main()
5: {
6:     int ctr;
7:
8:     for( ctr = 65; ctr < 91; ctr++ )
9:         printf("%c", ctr );
10:
11:     return 0;
12: }
13: /* Fin del programa */
```

5. ¿Qué hace el siguiente programa? (Tecléelo, compílelo y ejecútelo.)

```
1: /* EX2-5.C */
2: #include <stdio.h>
3: #include <string.h>
4: main()
5: {
6:     char buffer[256];
7:
8:     printf( "Enter your name and press <Enter>:\n");
9:     gets( buffer );
10:
11:     printf( "\nYour name has %d characters and spaces!",
                        strlen( buffer ));
12:
13:     return 0;
14: }
```

La diferencia tiene que ver con los apuntadores y el alcance de la variable. Los apuntadores y el alcance de la variable son dos aspectos muy importantes de la programación en C, y son tratados en los Días 9 y 12, "Apuntadores" y "Alcance de las variables", respectivamente. Por el momento, basta saber que usando `#define` para crear constantes se logra que los programas sean más fáciles de leer.

Taller

El taller proporciona un cuestionario que le ayudará a reafirmar su comprensión del material tratado y ejercicios para darle experiencia en el uso de lo que ha aprendido.

Cuestionario

1. ¿Cuál es la diferencia entre una variable entera y una de punto flotante?
2. Dé dos razones para usar una variable de punto flotante de doble precisión (tipo `double`) en vez de una variable de punto flotante de precisión sencilla (tipo `float`).
3. ¿Cuáles son las cinco reglas que indica el estándar ANSI que siempre serán ciertas cuando se ubica espacio para las variables?
4. ¿Cuáles son las dos ventajas de usar una constante simbólica en vez de una literal?
5. Muestre dos métodos para definir una constante simbólica llamada `MAXIMUM` y que tenga un valor de 100.
6. ¿Qué caracteres son permitidos en los nombres de variables del C?
7. ¿Qué reglas hay que seguir para la creación de nombres para variables y constantes?
8. ¿Cuál es la diferencia entre una constante simbólica y una literal?
9. ¿Cuál es el valor mínimo que puede contener una variable de tipo `int`?

Ejercicios

1. ¿Qué tipo de variable sería más adecuado para guardar los siguientes valores?
   a. La edad de una persona redondeada a años.
   b. El peso de una persona en libras.
   c. El radio de un círculo.
   d. Su salario anual.

e. El costo de una cosa.

f. La calificación máxima de un examen (suponga que es siempre 100).

g. La temperatura.

h. El valor neto de una persona.

i. La distancia a una estrella, en millas.

2. Determine nombres de variable adecuados para los valores del ejercicio 1.
3. Escriba declaraciones para las variables del ejercicio 2.
4. ¿Cuáles de los siguientes nombres de variable son válidos?

```
a. 123variable
b. x
c. anotación_total
d. Peso_en_#s
e. uno
f. costo-bruto
g. RADIO
h. Radio
i. radio
j. ésta_es_una _variable_para_guardar_el_ancho_de_una_caja
```