**Capítulo 2**

**Ejercicios:**

**2.7** Identifique y corrija los errores en cada uno de los siguientes enunciados (Nota: pudieran existir más de un error por cada enunciado):

a) scanf(“d”, value);

b) printf(“The product of %d and %d is %d”\n, x, y);

c) firstNumber + secondNumber = sumOfNumbers;

d) if(number => largest)

largest == number;

e) \*/ Program to determine the largest of three integers /\*

f) Scanf(“%d”, anInteger);

g) printf(“Remainder of %d divided by %d is\n”, x, y, x % y);

h) if(x = y);

printf(%d is equal to %d\n”, x, y);

i) printf(“The sum is %d\n,” x + y);

j) printf(“The value you entered is: %d\n, &value);

**2.8** Llene los espacios vacíos en cada uno de los siguientes:

a) \_\_\_\_\_\_\_ se utilizan para comentar un programa y mejorar su legibilidad.

b) La función utilizada para imprimir información en la pantalla es \_\_\_\_\_\_.

c) Un enunciado C que toma una decisión es \_\_\_\_\_\_\_.

d) Los cálculos por lo regular se ejecutan por enunciados \_\_\_\_\_.

e) La función \_\_\_\_\_ introduce valores del teclado.

**2.9** Escriba solo un enunciado de C o una línea que cumpla con cada uno de lo siguiente:

a) Imprima el mensaje “Enter two numbers.”

b) Asigne el producto de las variables b y c a la variable a.

c) Declare que un programa ejecuta un cálculo de muestra de nómina (es decir, utiliza texto que auxilia a documentar un programa.)

d) Introduzca dos valores enteros del teclado y coloque estos valores en las variables enteras a, b y c.

**2.10** Declare cuáles de las siguientes son verdaderos y cuáles son falsos. Explique sus respuestas.

a) Los operadores C se evalúan de izquierda a derecha.

b) Los siguientes son todos nombres válidos de variables: \_under\_bar\_, m928134, t5, j7, her\_sales, his\_account\_total, a, b, c, z, z2.

c) El enunciado printf(“a = 5;”); es un ejemplo típico de un enunciado de asignación.

d) Una expresión aritmética válida en C que no contenga paréntesis se evalúa de izquierda a derecha.

e) Los siguientes son todos nombres inválidos de variables: 3g, 87, 67h2, h22, 2h.

**2.11** Llene los espacios vacíos en cada uno de los siguientes:

a) Que operaciones aritméticas tienen el mismo nivel de precedencia que la multiplicación? \_\_\_\_\_\_

b) ¿Cuándo los paréntesis están anidados, que conjunto de paréntesis serán calculados en primer término en una expresión aritmética? \_\_\_\_\_\_\_\_\_\_

c) Una posición en la memoria de la computadora que puede contener valores diferentes en tiempos diferentes a lo largo de la ejecución de un programa se conoce como \_\_\_\_\_\_\_\_.

**2.12** ¿Qué es lo que, si es que es algo, se imprime cuando se ejecutan cada uno de los enunciados de C siguientes? Si no se imprime nada, entonces conteste “nada”. Suponga que x = 2 y y = 3.

a) printf(“%d”, x);

b) printf(“%d”, x + x);

c) printf(“x =”);

d) printf(“x = %d”, x);

e) printf(“%d = %d”, x + y, y + x);

f) z = x + y;

g) scanf(“%d%d”, &x, &y);

h) /\* printf(“x + y = %d”, x + y); \*/

i) printf(“\n”);

**2.13** ¿Que es, si es que es algo, de los enunciados siguientes contienen variables involucradas en lectura destructivas?

a) scanf(“%d%d%d%d%d”, &b, &c, &d, &e, &f);

b) p = i + j + k + 7;

c) printf(“Destructive read-in”);

d) printf(“a = 5”);

**2.14** Dada la ecuación , cual de los que siguen, si es que existe alguno, son enunciados correctos de C correspondientes a esta ecuación?

a) y = a \* x \* x \* x + 7;

b) y = a \* x \* x \* (x + 7);

c) y = (a \* x) \* x \* (x + 7);

d) y = (a \* x) \* x \* x + 7;

e) y = a \* (x \* x \* x) + 7;

f) y = a \* x \* (x \* x + 7);

**2.15** Declare el orden del cálculo de los operadores de cada uno de los enunciados de C siguientes, y muestre el valor de x después que se ejecute cada uno de ellos.

a) x = 7 + 3 \* 6 / 2 – 1;

b) x =2 % 2 + 2 \* 2 – 2 / 2;

c) x = (3 \* 9 \* (3 + (9 \* 3 / (3))));

**2.22** Qué es lo que imprime el código siguiente?

printf(“\*\n\*\*\n\*\*\*\n\*\*\*\*\n\*\*\*\*\*\n”);

**2.28** Distinga entre los términos error fatal y error no fatal. ¿Por qué preferiría experimentar un error fatal en vez de un error no fatal?

**Soluciones:**

**2.7** a) Falta el % y el &.

b) El \n está afuera de la cadena de control de formato. Falta un argumento.

c) La expresión aritmética debe ir en el lado derecho de un enunciado de asignación. Falta el “;”.

d) El operador relacional esta incorrecto (>=) y el operador de asignación esta incorrecto (=).

e) El comentario tiene los terminales (/\*) invertidos.

f) El nombre de la función “scanf” está mal escrito, está con mayúscula cuando debería ser con minúscula. Falta el &.

g) Falta el especificador de conversión %d para el argumento x % y.

h) El operador de igualdad esta incorrecto (==) y además tiene un punto y coma después del paréntesis derecho del if, dando la falsa intensión de enunciado vacío. Falta las dobles comillas en la cadena de control de formato del printf.

i) La coma está dentro de la cadena de control de formato, cuando debería estar afuera.

j) printf está mal escrito, debe estar todo en minúscula. Falta las dobles comillas para cerrar la cadena de control de formato. No debe colocarse el &.

**2.8** a) Comentarios

b) printf

c) if

d) De asignación

e) scanf.

**2.9** a) printf(“Enter two numbers\n”);

b) a = b \* c;

c) /\* Ejecuta un cálculo de muestra de nómina \*/

d) scanf(“%d%d%d”, &a, &b, &c);

**2.10** a) Falso. El operador de asignación se asocia de derecha a izquierda.

b) Verdadero. Todos los nombres de variables cumplen las características de identificadores válidos.

c) Falso. Es una llamada a una función de biblioteca.

d) Falso. Se evalúa según el nivel de preferencia de los operadores.

e) Falso. Muchos nombres de variables comienzan con un dígito, lo cual invalida el identificador de la variable.

**2.11** a) división y módulo.

b) Los pares de paréntesis más internos.

c) variable.

**2.12** a) 2.

b) 4.

c) x=.

d) x=2.

e) 5 = 5.

f) nada.

g) nada.

h) nada.

i) nada.

**2.13** Los incisos que involucran lectura destructiva son: inciso “a)” con las variables involucradas: b, c, d, e, f. El inciso “b)” con la variable involucrada: p.

**2.14** Enunciados correctos: a, d, e.

**2.15** a) Primero 3 \* 6 = 18, Segundo: 18 / 2 = 9, Tercero: 7 + 9 = 16, Cuarto: 16 – 1 = 15. El valor de x = 15.

b) Primero 2 % 2 = 0, Segundo: 2 \* 2 = 4, Tercero: 2 / 2 = 1, Cuarto: 0 + 4 = 4, Quinto: 4 – 1 = 3. El valor de x = 3.

c) Primero: (3) = 3, Segundo: 9 \* 3 = 27, Tercero: 27 / 3 = 9, Cuarto: 3 + 9 = 12, Quinto: 3 \* 9 = 27, Sexto: 27 \* 12 = 324. El valor de x = 324.

\*\*\* Continúan ejercicios prácticos hasta el ejercicio 2.22 \*\*\*

**2.22** Imprime:

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\* Continúan ejercicios prácticos hasta el ejercicio 2.28 \*\*\*

**2.28** Un *error fatal* es aquel que interrumpe y termina la ejecución de un programa al instante que se llega al error o incluso pudiera no compilar un programa, indicando donde está el error. El *error no fatal* es aquel donde el programa se compila y se ejecuta hasta su terminación, ocasionando errores lógicos en los resultados procesados por el programa.

- Es mejor encontrarse con un error fatal porque muestra donde está el error rápidamente y le indica al programador que hay un error a corregir. Diferente del error no fatal que pudiera pasar desapercibido por el programador y haciendo que el programa corra erróneamente, además de ser mas complicado encontrar el error porque no muestra en donde esta el error y por lo tanto el programador deberá buscarlo en cada enunciado del programa.