aprenderaprogramar.com

Ejercicios resueltos de captura y gestión de errores en programación con pseudocódigo. (CU00246A)

Sección: Cursos

Categoría: Curso Bases de la programación Nivel II

Fecha revisión: 2024

Autor: Mario R. Rancel

Resumen: Entrega nº 45 del Curso Bases de la programación Nivel II

24

EJERCICIO

El siguiente pseudocódigo tiene una serie de errores. Indicar dónde se localizan y de qué tipo son.

```
PROGRAMA MED [Ejercicio aprenderaprogramar.com]
    Variables
        Enteras: n
        Reales: Precio1, Precio2, Precio 3
1.  Inicio [Calcula la media de tres precios]
        2.  n = 1
        3.  Mostrar "¿Precio 1? (€)" : Pedir Precio1 : n = n + 1
        4.  Mostrar "¿Precio 2? (€)" : Pedir Precio2 : n = n + 1
        5.  Mostrar "¿Precio 3? (€)" : Pedir Precio3 : n = n + 1
        6.  Media = Precio1 + Precio2 + Precio3 / n
        7.  Mostrar "La media es", Media, euros"
8.  Fin
```

SOLUCIÓN

Analizaremos los errores empezando por la base de la pirámide.

- Error de sintaxis: en la línea 7 hay unas comillas de cierre sin existir las de apertura. Lo correcto sería *Mostrar "La media es", Media, "euros".*
- Error por proceso no válido: uso de la variable *Media* sin haber sido declarada.
- Error lógico tipo resultado incorrecto: la línea 2 inicia *n* en un valor incorrecto. *n* debe partir de cero.
- Error lógico tipo resultado incorrecto: la línea 6 realiza una operación que no es la deseada. Lo que se pretende hacer es *(Precio1 + Precio2 + Precio3) / n.* Es algo que es próximo a un error de sintaxis, pero téngase en cuenta que se trata de una operación matemática válida y el ordenador no va a detectar error alguno. Sólo es el programador quien puede valorar la existencia de un error.

EJERCICIO

Determinar qué errores contiene el siguiente programa.

```
PROGRAMA BUC [Ejercicio aprenderaprogramar.com]
    Variables
        Enteras: i, j
1.  Inicio
        2.  Mientras j <> 5 Hacer
                3.  Mientras i <> 10 Hacer
                        4.  Mostrar j, "*", i, "=", j * i
                        5.  i = i + 1
                        6.  j = j + 1
                7.  Repetir
        8.  Repetir
9.  Fin
```

SOLUCIÓN

Con una verificación mental podemos comprobar que el programa mostrará *0 * 0 = 0, 1 * 1 = 1, 2 * 2 = 4, 3 * 3 = 9, ..., 9 * 9 = 81* y a continuación se generará un bloqueo por ser *j* distinto de 5 y no poder salir del bucle externo al tiempo que *i = 10* por lo que no puede entrar en el bucle interno. El resultado es una evaluación continua de *j* e *i*, pero al no existir cambios en estas variables se produce un bucle infinito.

Este bucle constituye un error lógico tipo bucle infinito generado por un mal diseño del algoritmo. Preferiblemente debemos tratar de que no se cuelen este tipo de errores en nuestros programas mediante la verificación previa de los algoritmos. Si existiera algún fallo de este tipo y tratamos de ejecutar el programa puede producirse un bloqueo que nos lleve a perder todo lo que no hubiéramos guardado. Un programador suele ir guardando distintas versiones del código (aunque sea en un procesador de textos) de forma que si en un momento dado desea volver a comenzar desde un punto base pueda hacerlo.

Es patente que los bucles son partes "delicadas" dentro de un programa debido a su estructura circular. Por tanto, se les debe prestar especial atención durante las verificaciones.

EJERCICIO

Determinar qué errores existen en este programa y de qué tipo son.

```
PROGRAMA R01 [Ejercicio aprenderaprogramar.com]
        1. Inicio
                2. Numero = 9
                3. Llamar Raiz
                4. Mostrar "La raíz de", Numero, "es", Resultado
        5. Fin

        Módulo Raiz(Numero: Reales)
                Variables
                        Reales: Resultado
                1. Resultado = SQR(Numero)
        FinMódulo
```

SOLUCIÓN

- Error por proceso no válido: en la línea 2 tenemos el uso de un parámetro de un módulo genérico como si de una variable global se tratara.
- Error por proceso no válido: en la línea 3 se llama a un módulo genérico sin pasarle el parámetro requerido (tipo real).
- Error por proceso no válido: en la línea 4 se usan un parámetro y una variable local del módulo *Raiz* como si de variables globales se tratara.

Una escritura correcta del pseudocódigo puede ser:

```
PROGRAMA R01 [Ejercicio aprenderaprogramar.com]

    1. Inicio
        2. Llamar Raiz(9)
    3. Fin

    Módulo Raiz(Numero: Reales)
        Variables
            Reales: Resultado
        1. Resultado = SQR(Numero)
        2. Mostrar "La raíz de", Numero, "es", Resultado
    FinMódulo
```

En este caso, desde el algoritmo principal se llama al módulo *Raiz* pasándole el parámetro requerido y éste nos presenta el resultado a través de la variable local *Resultado.*

EJERCICIO

Se sabe que el módulo *CalculaSuma* mostrado a continuación debe devolver un valor *Suma* positivo. Establecer un control de errores vía lógica para impedir que se presenten resultados iguales o menores que cero.

```
Módulo CalculaSuma(Numero: Entero)
    Variables
        Enteras: i, Suma
    1. Suma = 0
    2. Desde i = 1 hasta Numero Hacer
        Suma = Suma + Valor(i)
      Siguiente
    3. Mostrar "La suma vale", Suma
FinMódulo
```

SOLUCIÓN

```
Módulo CalculaSuma(Numero: Entero) [Ejercicio aprenderaprogramar.com]
    Variables
        Enteras: i, Suma
    1. Suma = 0
    2. Desde i = 1 hasta Numero Hacer
        Suma = Suma + Valor(i)
      Siguiente
    3. Si Suma > 0 Entonces
        Mostrar "La suma vale", Suma
      SiNo
        Mostrar "Se ha producido un error y no se pueden mostrar resultados"
      FinSi
FinMódulo
```

EJERCICIO

Se ha desarrollado un programa (*Programa R02)* que consta de dos módulos llamados *GeneraDato* y *Raiz.* A través de *GeneraDato* se solicita un número entero entre 1 y 20 al usuario mientras que *Raiz* devuelve la raíz cuadrada de ese número entero.

Introducir un módulo de gestión de errores que impida la parada de este programa en caso de que el usuario no introduzca el tipo de dato correcto.

Nota: Vamos a suponer que si el usuario introduce un tipo de dato no esperado (real, alfanumérico...) se genera un error. No siempre será así, ya que depende del tipo de entrada que haya y del lenguaje de programación que se utilice.

```
PROGRAMA R02 [Ejercicio aprenderaprogramar.com]

    Variables
        Enteras: Dato

    1. Inicio
        2. Llamar Generadato
        3. Llamar Raiz(Dato)
    4. Fin

    Módulo Generadato
        1. Mientras Dato < 1 ó Dato > 20
            Mostrar "Introduzca un número entero entre 1 y 20"
            Pedir Dato [Aquí puede originarse el error]
          Repetir
    FinMódulo

    Módulo Raiz(Numero: Enteros)
        Variables
            Reales: Resultado
        1. Resultado = SQR(Numero)
        2. Mostrar "La raíz de", Numero, "es", Resultado
    FinMódulo
```

SOLUCIÓN

Si el usuario no introduce un entero se espera un error al tratar de hacer una asignación no viable, y la variable *Dato* continúa con su valor anterior. Introducimos un módulo de gestión de error de modo que se devuelve el control al bucle y se vuelve a pedir la entrada al usuario tantas veces como entradas incorrectas se den.

```
PROGRAMA R03 [Ejercicio aprenderaprogramar.com]

        Variables
                Enteras: Dato

        1. Inicio
                2. Activar ErrorControl(Llamar GestionError)
                3. Llamar Generadato
                4. Llamar Raiz(Dato)
        5. Fin

        Módulo Generadato
                1. Mientras Dato < 1 ó Dato > 20
                        Mostrar "Introduzca un número entero entre 1 y 20"
                        Pedir Dato
                   Repetir
        FinMódulo
-------------------------------------------------------------------------------
        Módulo Raiz(Numero: Enteros)
                Variables
                        Reales: Resultado
                1. Resultado = SQR(Numero)
                2. Mostrar "La raíz de", Numero, "es", Resultado
        FinMódulo

        Módulo GestionError
                1. Mostrar "Se ha producido un error. Es posible que los
                        datos que usted proporciona no se ajusten a lo
                        solicitado. Compruébelo por favor"
        FinMódulo
```

Comentarios: En general, preferimos evitar que se produzca el error antes que introducirlo como un elemento "natural" en el programa, reservando la gestión de errores para situaciones realmente imprevisibles o sobre las que no tenemos capacidad de actuación.

Próxima entrega: CU00247A

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:
http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=36&Itemid=60