aprenderaprogramar.com

Errores. Prevención, gestión y tipos. Verificación de algoritmos. (CU00241A)

Sección: Cursos

Categoría: Curso Bases de la programación Nivel II

Fecha revisión: 2024

Autor: Mario R. Rancel

Resumen: Entrega nº 40 del Curso Bases de la programación Nivel II

24

ERRORES. PREVENCIÓN, GESTIÓN Y TIPOS. VERIFICACIÓN DE ALGORITMOS.

INTRODUCCIÓN

Llamamos error a cualquier circunstancia que da lugar a un malfuncionamiento de un programa. Por malfuncionamiento se entiende una respuesta indeseada: desde meros problemas estéticos hasta graves fallos o bloqueos.

Intuitivamente tenemos muchas ideas relacionadas con los errores. Aplicado al devenir diario, estamos acostumbrados a hablar de "fallos" en las máquinas. De un coche por ejemplo, puede decirse que "está perfecto" o "tiene un pequeño fallo pues hace un ruidito la correa, pero anda bien". Otras veces diremos que "está teniendo problemas graves con el cambio de marchas" o "se paró y tuve que llamar a una grúa". Con los programas informáticos pasa algo similar: pueden presentar fallos leves o casi despreciables, fallos notables pero aún así funcionan y fallos totales en determinadas circunstancias. Sin duda, la mejor situación es la del programa libre de errores. O dicho de otra manera: el mejor error es el que no existe.

¿Cómo construir programas sin errores? Si tuviéramos la respuesta a esta pregunta quizás fuéramos presidentes de alguna multinacional, pues es algo a lo que muchísima gente intenta responder. Pero aún no teniéndola, quizás sí podemos indicar algunos factores importantes al respecto:

- La experiencia derivada de enfrentarnos a errores y buscarles solución nos permite mejorar como programadores. Un programador más experto comete menos errores.
- Existen metodologías contrastadas que son efectivas para evitar la aparición de errores. Son lo que globalmente se llama "buenas prácticas de programación" y comprenden multitud de pautas de diseño y construcción de programas, muchas de las cuales han sido ya expuestas. Estas pautas van desde meras cuestiones de presentación hasta aspectos fundamentales de la programación. Si algo hemos hecho en estas páginas es hablar de método y de diseño. No vamos a extendernos en esto ahora. Sí queremos recordar que cuestiones aparentemente sencillas como asumir que "se recomienda no escribir muchas órdenes en una sola línea" pueden suponernos grandes ventajas. Y ello, sin esfuerzo. Es suficiente con tener método.
- Es importante diseñar y verificar módulos o algoritmos independientes para luego realizar integraciones que habrá a su vez que verificar. Integrar partes no verificadas supone crear estructuras de difícil verificación y corrección.
- Programar requiere concentración y dedicación. Igual que no imaginamos una partida de ajedrez en un entorno de ruidos y distracciones, no pensemos que podemos programar en cualquier ambiente. Un buen programador distraído o preocupado en otras cosas puede ser un mal programador.

Tenemos claro que uno de nuestros objetivos ha de ser construir programas libres de errores. Ahora bien, somos humanos y hemos de asumirlo. Por suerte o por desgracia vamos a tener que convivir con errores, independientemente de que pongamos más o menos medios para evitarlos. Si esto es así, el programador no "odia" los errores. Al programador "le interesan" los errores desde dos puntos de vista:

a) Para su identificación y eliminación, evitando así que aparezcan en momentos inoportunos.
b) Para su gestión cuando, a pesar de todo, se produzcan.

De ello vamos a hablar en las siguientes páginas.

TIPOS DE ERRORES

Hay distintos tipos de errores y distintas formas de clasificarlos. Vamos a ver primero una serie de definiciones para luego ver las clasificaciones. Los cuatro tipos básicos de errores son: de sintaxis, por proceso no válido, lógicos tipo bucle infinito y lógicos tipo resultado incorrecto.

ERRORES DE SINTÁXIS

Son todos aquellos que se generan por infringir las normas de escritura de un lenguaje. Suelen deberse a olvidos o a desconocimiento (programadores principiantes) y comprenden falta o mal uso de elementos separadores (comas, puntos y comas, dos puntos, etc.), palabras mal escritas (por ejemplo *Mietras* en vez de *Mientras* o *Finelizar* en vez de *Finalizar*).

Suelen ser errores obvios y fáciles de detectar. La mayoría de los lenguajes tienen herramientas de ayuda para facilitar la escritura y la corrección del código desde el punto de vista sintáctico. Además, en general, no es posible la ejecución de un programa con errores de sintaxis y se insta al programador a su corrección en los distintos puntos donde pueda haber problemas por parte del programa de gestión del lenguaje que se esté utilizando.

Téngase en cuenta que aparte de olvidos, escritura incorrecta de signos, palabras mal escritas, etc. la omisión de términos obligatorios también será considerada error de sintaxis. Por ejemplo un *Si* que no se cierra con un *FinSi* o un *Repetir* que no se corresponde con un *Mientras.*

ERRORES POR PROCESOS NO VÁLIDOS

Cada lenguaje tiene sus particularidades o formas de concepción. Cualquier sintaxis aparentemente correcta pero que infrinja las normas de construcción de programas que define el lenguaje dará lugar a un error. Las posibilidades son muy variadas. Por citar algunas habituales señalaremos:

- Indeterminaciones matemáticas (p. ej. *SQR(a)* con $a < 0$).
- Asignar un valor a la variable que no coincide con el tipo declarado para la variable.
- Uso de variables no declaradas.
- Modificaciones no permitidas del número de localizadores de un array dinámico.
- Llamadas a módulos que no existen.
- No pasar los parámetros o pasar un número incorrecto de parámetros a un módulo genérico.
- Tratar de extraer datos o usar ficheros que no se encuentran.

Los procesos no válidos pueden deberse al programa o a circunstancias ajenas al programa, como que el usuario no introduzca el dato esperado, que un señalero no esté donde tiene que estar, que los datos no estén ordenados como tenían que estarlo, que un archivo no contenga la cantidad de datos que se espera o que un dato o fichero no esté donde se espera. Suelen ser errores que se pueden detectar y gestionar. Su localización se ve facilitada por las herramientas de corrección o depuración del lenguaje que utilicemos, como los ya comentados “paso a paso”.

ERRORES LÓGICOS

Son todos aquellos derivados de un mal diseño de los algoritmos o la arquitectura modular. Las posibilidades son muchas y los efectos posibles también. Pueden ir desde el bloqueo o detención indeseada del programa hasta un resultado incorrecto, una parte del programa que no se ejecuta, etc.

Los errores lógicos que afectan al desarrollo del programa son más fáciles de detectar que aquellos que no generan aparentemente problemas. A su vez, de entre los que aparentan no suponer problema, los que generan muy pequeñas disfunciones o desviaciones del resultado resultan más difíciles de detectar que los que dan lugar a resultados ostensiblemente desviados de lo esperado.

Los errores lógicos son quizás los que más grandes quebraderos de cabeza originan a los programadores, los más difíciles de evitar y los más difíciles de detectar. Un programa lleno de errores de sintaxis pero de lógica exquisita es una bendición comparado con un programa lleno de errores de lógica pero de perfecta sintaxis.

Evitar errores lógicos no pasa ya por conocer bien el lenguaje y las formas de construcción del lenguaje. Pasa por ser un buen programador, un buen pensador. Y eso es difícil de enseñar o de aprender. Partamos de estrategias y métodos que se saben eficientes y dejemos a la experiencia realizar el resto.

Vamos a dividir los errores lógicos en dos tipos.

1. ***Errores lógicos tipo bucle infinito:*** son aquellos que dan lugar a una parada o bloqueo del programa. Incluiremos aquí procesos en realidad no infinitos pero que consumen un tiempo desmedido respecto a lo esperado.
2. ***Errores lógicos tipo resultado incorrecto:*** no bloquean el flujo del programa pero dan lugar a un resultado desviado. El grado de desviación escaso puede dificultar su detección y corrección. El ordenador no puede saber que existe un error pues sólo el programador que idea el proceso puede decir si éste es correcto o no.

Hasta aquí lo que hemos llamado cuatro tipos básicos de errores. Cuando existe un procedimiento a seguir previsto ante la presencia de uno de estos errores decimos que se trata de un error gestionado. Ser gestionado es un atributo del error que cambia sus propiedades. La naturaleza sigue siendo la misma pero los efectos no, al ser “controlados” o “encauzados”. Lo ideal es que todos los errores posibles sean gestionados, pero esto no garantiza calidad ni buen funcionamiento del programa. Únicamente asegura que el programador y/o usuario mantienen un cierto control del flujo del programa con el fin de poder obtener resultados parciales, reorientar procesos o poder proceder a una salida controlada.

En la gestión del error normalmente interviene el propio programa. Cuando se produce el error, éste es detectado y se procede a su gestión. Por tanto un error lógico tipo resultado incorrecto no puede ser gestionado por esta vía (a la que llamaremos gestión vía detección) puesto que para el ordenador ese error no existe. La única posibilidad de gestionarlos es a su vez la vía lógica a través de módulos de control de resultados, lo que según qué casos puede resultar muy complicado.

Próxima entrega: CU00242A

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