

ERRORES COMUNES EN EL PROCESO DE PROGRAMACION

Como introducción ilustramos los resultados de buscar en diferente bibliografía una clasificación de errores, que por ejemplo existen en sitios de Internet referidos a:

A) Entorno Visual Studio 2005 (Microsoft)

Se deja en claro que conviene conocer los tipos de errores de programación que deberá buscar y corregir, que pertenecen a tres categorías: *errores de compilación, errores en tiempo de ejecución y errores lógicos.*

Errores de compilación:

Los errores de compilación, también conocidos como *errores del compilador*, son errores que impiden que su programa se ejecute. Cuando se presiona F5 para ejecutar un programa, Visual Basic compila el código en un lenguaje binario que entiende el equipo. Si el compilador de Visual Basic se encuentra con código que no entiende, emite un error de compilador.

La mayoría de los errores del compilador se deben a errores cometidos al escribir el código. Por ejemplo, puede escribir mal una palabra clave, omitir alguna puntuación necesaria o intentar utilizar una instrucción End If sin antes utilizar una instrucción If.

Afortunadamente el Editor de código de Visual Basic fue diseñado para identificar estos errores antes de que se intente ejecutar el programa.

Errores en tiempo de ejecución:

Los errores en tiempo de ejecución son errores que aparecen mientras se ejecuta el programa. Estos errores aparecen normalmente cuando el programa intenta una operación que es imposible que se lleve a cabo. Un ejemplo de esto es la división por cero. Suponga que tiene la instrucción siguiente: `Speed = Miles / Hours`

Si la variable `Hours` tiene un valor de 0, se produce un error en tiempo de ejecución en la operación de división. El programa se debe ejecutar para que se pueda detectar este error y si `Hours` contiene un valor válido, no se producirá el error. Cuando aparece un error en tiempo de ejecución, puede utilizar las herramientas de depuración de Visual Basic para determinar la causa.

Errores lógicos:

Los errores lógicos son errores que impiden que el programa haga lo que estaba previsto. El código puede compilarse y ejecutarse sin errores, pero el resultado de una operación puede generar un resultado no esperado. Por ejemplo, puede tener una variable llamada `FirstName` y establecida inicialmente en una cadena vacía. Después en el programa, puede concatenar `FirstName` con otra variable denominada `LastName` para mostrar un nombre completo. Si olvida asignar un valor a `FirstName`, sólo se mostrará el apellido, no el nombre completo como pretendía. Los errores lógicos son los más difíciles de detectar y corregir, pero Visual Basic también dispone de herramientas de depuración que facilitan el trabajo.

Seguramente que la lectura de dicho contenido resultará algo novedoso sobre la mención de determinadas herramientas para depurar errores, que durante la cursada, basada en el paradigma imperativo no se mencionan. Es lógico porque Visual Studio hace uso del paradigma orientado a objetos. Pero es interesante como en otro entorno se clasifican a los mismos errores que podemos llegar a cometer en el paradigma imperativo.

B) Sobre programación en C y C++ (Curso práctico de programación en C y C++ de Jorge Badenas Carpio y otros. 2001)

En el desarrollo de un proyecto de software se pueden cometer errores en las distintas fases: especificación, análisis, diseño, codificación, pruebas y mantenimiento. En las fases de codificación, pruebas de programas y en el desarrollo del programa se comenten normalmente errores en la escritura

del código y en la estructuración del programa, que se denominan errores ortográficos, sintácticos y de estructura (son errores sintácticos especiales donde se implican llamadas a procedimientos externos, inclusión de instrucciones no propias del lenguaje, etc.) . Estos errores normalmente se detectan en la fase de compilación y se pueden corregir sin mayor problema.

Sin embargo el compilador no puede detectar los denominados errores lógicos, debidos a la utilización de instrucciones sintácticamente correctas pero dispuestas de forma distinta a lo especificado en el diseño, cuyo funcionamiento no se ajusta a lo esperado.

El compilador tampoco puede detectar que las peticiones de recursos externos al programa, cuando éste esté en funcionamiento, fallen porque los recursos no están disponibles o no son los adecuados. Estos errores se producen en tiempo de ejecución. De estos conceptos se llega a clasificar a los errores en tres grupos: los errores críticos, los errores no graves y los errores irrelevantes.

Errores irrelevantes:

Son los que no afectan al funcionamiento correcto del programa, aunque son errores y deban ser corregidos, por ejemplo ventanas de entrada de datos con errores ortográficos ó con un nombre de referencia distinto al que correspondería, afectando más a la estética que la propia operatividad del programa. Por ej: indicar la palabra DNI en lugar de Número de documento porque existen diferentes tipos de documento.

Errores no graves:

Son aquellos que sería deseable arreglar porque pueden afectar al funcionamiento correcto del programa, aunque en una sesión normal no ocurra el error.

Errores críticos:

Son aquello que hacen imposible el funcionamiento correcto del programa mientras no se corrijan o incluso hacen imposible el propio funcionamiento del programa. Los errores críticos más comunes en los lenguajes de programación, sobre todo de tipo funcional ó imperativo son:

- 1- Errores de división por cero ó por un divisor que vale 0.
- 2- Errores con punteros: punteros mal asignados ó no inicializados. Son los más difíciles de detectar.
- 3- Errores de salida de un bucle: ejecución de un bucle infinito debido a la errónea especificación de las condiciones de terminación ó de variaciones de contadores ó índices de control de iteraciones.
- 4- Errores de desbordamiento de pila: exceso de llamadas recursivas a una misma función.
- 5- Errores de límites: intento de acceso fuera de los límites de un vector.
- 6- Errores por falta de memoria: en la asignación dinámica de memoria el operador ó función de reserva de memoria no devuelve un valor correcto porque no se ha liberado memoria no usada anteriormente y no se puede asignar la suficiente.
- 7- Errores de objetos temporales: mal uso de constructores y destructores de objetos.
- 8- Errores de conversión: pérdida de signo ó de los decimales en las conversiones de datos definidas por el usuario.
- 9- Errores de portabilidad: uso de instrucciones ó recursos recursivos del lenguaje dependiente del sistema operativo, que no son compatibles con otros sistemas operativos.
- 10- Errores de disgregación de decisiones: separación de las instrucciones de evaluación de una condición y de las instrucciones de acción consecuente si se cumple la condición.

Como resultado de la lectura de esta segunda fuente de clasificación, considerando los paradigmas imperativo, orientado a objetos y funcional, vemos la especificación de varios errores que fueron considerados durante el desarrollo de las unidades temáticas previas, aunque algunos no se consideraron en el desarrollo de un algoritmo, dado que al hacer uso de un pseudocódigo no tenemos en cuenta los errores que se producen durante la ejecución en máquina, que sean independiente de una correcta lógica (por ejemplo falta de memoria).

Por lo tanto para tener una clasificación de errores producidos en el desarrollo de programas en el paradigma imperativo se considerará suficiente la siguiente clasificación que se especifica en el libro “Tutorial on Software Design Techniques” editado por I.E.E.E. (1990), dado que respeta en gran parte los errores mencionados en la introducción:

TIPO DE ERRORES	Ejemplos
Por referencia de datos	<ol style="list-style-type: none"> 1- Uso de variables sin valor asignado 2- Índices de arreglos fuera de rango 3- Índices de arreglos no enteros 4- Valor de punteros sin dirección válida 5- Diferente estructura interna usada en la lectura ó grabación con las estructuras externas 6- La estructura de un dato no es la misma para todos los módulos
Por declaración de datos	<ol style="list-style-type: none"> 1- Variables no declaradas 2- Variables con el mismo nombre 3- Arreglos ó cadena de caracteres no iniciados 4- Valores iniciales no apropiados 5- Longitudes y tipo de variables no apropiados para resolver el problema
De computación	<ol style="list-style-type: none"> 1- Cálculos con variables no numéricas 2- Cálculos entre variables de diferente tipo 3- Operaciones entre variables de diferentes longitudes 4- Variables con longitud no apropiada al os valores en una expresión 5- Resultados que produzcan “overflow” 6- Divisiones por cero 7- Valor de índice fuera de rango
De Comparación	<ol style="list-style-type: none"> 1- Comparaciones entre variables no consistentes por el concepto que representan 2- Variables a comparar de diferente tipo de dato 3- Expresiones lógicas incorrectas 4- Mezcla de comparaciones y expresiones lógicas
De Control de Proceso	<ol style="list-style-type: none"> 1- No termina alguna repetición 2- No termina el programa 3- No se ejecuta por lo menos una vez una repetición 4- Mal los cierres de las estructuras de control 5- Existen decisiones incompletas en su expresión lógica
En las interfases	<ol style="list-style-type: none"> 1- Número no adecuado de argumentos al llamar a una subrutina 2- Argumentos de diferente tipo tanto en la llamada como en el encabezamiento del procedimiento 3- No se respeta el orden de los parámetros entre la llamada y el encabezamiento
De Entrada – Salida	<ol style="list-style-type: none"> 1- Atributos incorrectos en los archivos 2- Instrucciones incorrectas de apertura de archivos 3- Especificación de formato diferentes en las instrucciones de lectura y grabación 4- Variables de diferente tamaño con respecto a los valores de los registros externos 5- No se consideró condición de fin de archivo 6- Errores en el texto de reportes ó mensajes
Otros	<ol style="list-style-type: none"> 1-Son importantes los mensajes de aviso ó advertencia en el caso de producirse 2- Se omitió alguna función en la codificación

