Fundamentos del Software. Relación de Problemas 3

▼ **1.- Un procesador (CPU) puede interpretar y ejecutar directamente las instrucciones de un programa en:**

c) Sólo lenguaje máquina. En este tema hemos visto que siempre se deben traducir las instrucciones de alto nivel a bajo nivel para que la CPU las pueda desempeñar.

▼ **2.- ¿Es lo mismo un token que un lexema? Muestre algún ejemplo.**

No es lo mismo. Un lexema es una secuencia de caracteres del alfabeto con significado propio, mientras que un token es un concepto asociado a un conjunto de lexemas. Por ejemplo, el token OP_COMP (operador comparativo) tiene asociados lexemas como ≤, ==, etc.

▼ **3.- ¿El compilador es la única utilidad necesaria para generar un programa ejecutable en una computadora?**

No, el compilador siempre requiere el enlazador. Aunque el compilador se encargue de traducir a lenguaje máquina el programa, necesita que el enlazador añada las bibliotecas necesarias a los ficheros objeto del compilador para que el ejecutable final se pueda generar.

▼ **4.- El análisis léxico es una etapa de la compilación cuyo objetivo es:**

b) Descomponer el programa fuente en sus componentes léxicos (tokens).

▼ **5.- El análisis sintáctico es una etapa de la compilación cuyo objetivo es:**

a) Extraer la estructura de cada sentencia, reconociendo los componentes léxicos (tokens) del lenguaje. Es decir, comprueba si la estructura de la secuencia de tokens no viola el conjunto de reglas gramaticales aplicables.

▼ **6.- Para el siguiente código que aparece a la izquierda en lenguaje C++ (fichero test.cpp), indique el nombre**

de la fase en la que el compilador produce el mensaje de error que aparece a la derecha y explique la naturaleza del mismo:

a) Este error se producirá en el análisis sintáctico, pues este error se trata de un fallo de estructura (fallo de asignación), y a la derecha del identificador += debería haber alguna variable para que dicha estructura sea correcta.

b) Este error se producirá en el análisis semántico, ya que este controla el significado de las construcciones sintácticas. En este caso la construcción sintáctica en concreto es una asigación, pero carece de significado ya que se está intentando hacer entre un dato un tipo int y un char*, los cuales son incompatibles.

c) Este error se producirá en el análisis léxico debido a que ¬ no se corresponde con ningún token y hará que el programa fuente no se pueda descomponer en tokens.

▼ **7.- Muestre un ejemplo a partir de una sentencia en lenguaje C++ en la que un error léxico origine un error sintáctico derivado y otro error léxico que no derive en error sintáctico.**

Error léxico que deriva en un error sintáctico

Si estas fuesen 2 líneas de un programa:

double diner¬o;

dinero += 2.3;

En la primera línea habría un error léxico ya que ¬ no se corresponde con ningún token y, además, en la siguiente línea se produciría un error sintáctico ya que la estructura dinero += 2.3 no sería correcta al no estar declarada la variable dinero, y entonces el operador += no estaría siendo usado correctamente.

Error léxico que no deriva en un error sintáctico

Si se escribe alguna letra con tilde, el analizador léxico dará error, pero esto no afecta a ninguna estructura de ninguna sentencia y por lo tanto, no podrá derivar en un error sintáctico.

▼ 8.- Muestre un ejemplo a partir una sentencia de en lenguaje C++ en la que un error léxico origine un error sintáctico y semántico derivados y otro error léxico que no los derive.

Este ejemplo origina un error sintáctico y semántico:

```
class Index {};

main()
{
  ...
  Index ejemplo;
  Index index;
  ...
  ejemplo = Index;
  ...
}
```

Este ejemplo no los deriva:

```
for(int i=0; i<4; i++)
{...}
```

En el primero hemos confundido una minúscula con una mayúscula, y en lugar de poder usar el operador =, estamos diciéndole que lo iguale a una clase, cosa que no permite la sintaxis y es incorrecta semánticamente.

Sin embargo el segundo simplemente es escribir mal el for y no lo reconoce como nada, solo una palabra mal escrita.

▼ 9.- ¿Sería siempre posible realizar la depuración de un archivo objeto? Razone la respuesta.

No, ya que un archivo objeto es resultado de la compilación, y la depuración es una técnica que en muchos casos solo se puede llevar a cabo en archivos ejecutables, y los archivos objeto no lo son.

▼ 10.- Dado un programa escrito en lenguaje ensamblador de una arquitectura concreta, ¿sería directamente interpretable ese código por esa computadora? En caso contrario ¿qué habría que hacer?

No, no sería directamente interpretable por la computadora, ya que es necesario que esté escrito en el lenguaje de más bajo nivel (lenguaje máquina). Por tanto, lo que habría que hacer sería traducirlo a dicho lenguaje.

▼ **11.- ¿Sería necesario usar siempre el enlazador para obtener un programa ejecutable?**

No sería necesario. Si en el lenguaje con el que se esté trabajando, un fichero ejecutable puede depender únicamente de un archivo objeto, que no esté referenciado a librerías externas, entonces no sería necesario el enlazador para obtener el programa ejecutable.

▼ **12.- Dado un único archivo objeto, ¿podría ser siempre un programa ejecutable y correcto simplemente añadiendo la información de cabecera necesaria?**

No, ya que podría darse la existencia de referencias externas a librerías desconocidas. En cambio, si lo anterior no se da, sí se podrá ejecutar teniendo solo la información de cabecera (dando por supuesta la existencia del programa principal main).

▼ **13.- Dado un programa ejecutable que requiere de una biblioteca dinámica, ¿por qué no es necesario recompilar el código fuente de dicho programa si se modifica la biblioteca?**

No es necesario porque el código del programa no alberga el código de la biblioteca, de manera que toda biblioteca dinámica está preparada en caso de ser requerida y cargada en tiempo de ejecución (esto implica que no tiene que ser enlazada en tiempo de compilación, de ahí que no haga falta recompilar).

▼ **14.- Indique en qué fase del proceso de traducción y ejecución de un programa se realizará cada una de las siguientes tareas:**

(a) Enlazar una biblioteca estática.

Fase de ejecución → Enlazado

(b) Eliminar los comentarios del código fuente.

Fase de Análisis → Analizador léxico

(c) Mensaje de error de que una variable no ha sido declarada.

Fase de Análisis → Analizador semántico

(d) Enlazar una biblioteca dinámica.

Fase de Ejecución → Carga y ejecución

▼ 15.- Indique en qué fase o fases del proceso de compilación de un lenguaje de programación de alto nivel se detectarían los siguientes errores:

a) Una variable no está definida → En el análisis semántico, ya que una variable no declarada da lugar a una construcción que carece de significado correcto, y del significado es de lo que se encarga el analizador semántico.

b) Aparece un carácter o símbolo no esperado → En el análisis léxico ya que dicho carácter no se corresponderá con ningún token conocido.

c) Aparecen dos identificadores consecutivos → En el análisis sintáctico ya que poner dos identificadores seguidos no es una estructura correcta.

d) Aparecen dos funciones denominadas bajo el mismo nombre → En el análisis semántico, ya es incoherente nombrar dos funciones de la misma forma. Cuando se vaya a llamar a una de ellas, habrá problemas para determinar cuál de ellas usar.

e) Aparece el final de un bloque de sentencias pero no el inicio del mismo → En el análisis sintáctico ya que será un bloque de sentencias incompleto y será incompatible con las reglas gramaticales que comprueba el analizador sintáctico.

f) Aparece un paréntesis cerrado y no se ha podido emparejar con su correspondiente paréntesis abierto → En el análisis sintáctico ya que es una secuencia de tokens que el analizador léxico puede descomponer pero cuando actúa el analizador sintáctico, detecta que falta un paréntesis que cierra y por lo tanto, no será una secuencia de tokens correcta con la gramática.

g) Una llamada a una función que no ha sido definida → En el análisis semántico, pues no tiene sentido llamar a una función que no ha sido definida, esto da lugar a una construcción sin significado correcto y el analizador semántico la detecta.

h) En la palabra reservada main aparece un carácter extraño no esperado, por ejemplo main → En el análisis léxico, ya que es un carácter que no se esperaba

en ese lugar y mai¿n no se corresponderá con ningún token en concreto.

▼ **16.- ¿Todo error sintáctico origina un error semántico? En caso contrario, demuéstrelo usando algún contraejemplo**

No tiene por qué, he aquí un ejemplo:

double variable = 0.45;

variable *=;

Evidentemente hay un error sintáctico ya que falta un dato de tipo int o double a la derecha del operador *= para que la estructura sea coherente con la gramática, pero la semántica no se ve alterada ya que usar el operador *= con "variable" tiene un significado correcto.

Realizado por: Lorena Cáceres Arias, Lucía Cepeda González, José Alberto Hoces Castro y Quintín Mesa Romero