Apellido: Guaymas **Nombres:** Matías Julián **Legajo:** 230611/0

Conteste las preguntas CON LAPICERA EN ESTA HOJA, de otra forma no se considerará respondida la pregunta.

CyPLP 2025 — EMT - 1 - A

Verdadero o Falso

Un error de tipo semántico siempre es detectado por el compilador . ❑ V ■ F ✓

Pregunta

```
1  program EjemploErrores;
2  var
3    numero: integer;
4    texto: string;
5
6  begin
7    numero := 10;
8    texto := "Hola mundo" (A)
9
10   writeln(numero + texto) (B)
11
12   resultado := numero + texto; (C)
13
14   writeln(resultado); (D)
15 end.
```

Dado el siguiente código escrito en Pascal que es un lenguaje compilado:

1. Indique, si es que existen los errores y su tipo junto con el momento en que se producen, justificando la respuesta.

Respuesta

Existen varios errores: En la línea 8 (A) hay dos errores sintácticos: se usan comillas dobles en vez de usar comillas simples para encerrar al string y falta el ';' (punto y coma) al final de la línea. Ambos errores son detectados en compilación.

En la línea 10 (B) hay un error sintáctico: falta el punto y coma al final de la línea (detectado en compilación) y un error semántico estático (captado antes de la ejecución) ya que se intenta sumar un entero con un string, siendo una operación inválida para Pascal. ✓

En la línea (12) hay dos errores semánticos estáticos: "resultado" no está definido como variable (estático: detectado en compilación) y se vuelve a sumar dos tipos de datos distintos (string y entero); pues en la línea 8 (ya detectado en compilación).

En la línea 14 "resultado" no está declarada como variable y se intenta imprimir su valor, dando error semántico estático (detectado en compilación).

(A) Con respecto a la sintaxis están bien

Pregunta

Mencione los mecanismos que conoce por los cuales es posible definir la sintaxis de un lenguaje de programación atendiendo o no, cuestiones contextuales como el tipo de las variables o su inicialización. En el caso de ser más de uno los mecanismos mencione las diferencias entre estos de forma breve.

Respuesta

• BNF: Son gramática libre de contexto formal. Donde se definen reglas o producciones. Usa met. símbolos
• EBNF: es BNF extendido donde se agregan operadores como * y + para que adquieran
ofrecen simplicidad y claridad a la hora de hacer las reglas.
• Diagramas de Conway: Diagrama en el cual se parte de un estado inicial y se define
un diagrama por producción que hayé. Es un grafo sintáctico se visualiza más que BNF y EBNF-
• Arbol de derivación sintáctico: Estructura jerárquica donde se descompone cada regla en forme
de arbol. Es útil para corroborar la gramática del lenguaje.
• Lenguaje Informal: Manera se explica la sintaxis en palabras.
BNF y EBNF son gramáticas formales y las otras no. Los diagramas son los más visuales,
Diagramas y árbol son más legibles que los otros. EBNF es más conciso que BNF.
BNF y EBNF son ideales para compiladores, diagramas para enseñanza, lenguaje natural)
Pregunta dare documentación y árboles para semántica

```
1  int c;
2  int j[10];
3  int main()
4  {
5      printf("Hello, World!\n");
6      int z=0;
7      static int c=2;
8      int *p;
9      p=&z;
10     return 0;
11 }
```

De acuerdo al código que se muestra a la izquierda:

1. identifique las variables según su tipo al momento de la alocación
2. Indique características mínimas de cada una,
3. indique su alcance en términos léxicos.

Respuesta

int c: automática. Es var global.
int j[10]: automática. Es var global.
int z: automática. Es var local al main.

Static int c: estática. Es var local.
int *p: automática. Es var local. p1; es dinámica.
Automática se aloca en la pila de memoria.
Estática: se aloca en la memoria de código.
Dinámico: se aloca en el heap.
Una variable es una 5-upla: tiene un tipo l-valor, r-valor, alcance y nombre.
Una variable es global cuando puede ser utilizada en todo el programa, y todos pueden modificarla. Una variable es local cuando la variable es conocida solo en el bloque en el que fue definida y no por fuera.

1. B

2. B

3. R

No menciona todas las gramáticas en función a cuestiones contextuales.

No menciona las gramáticas sensibles al contexto como la Gramática de Atributos.

No menciona las diferencias entre gramáticas libres y sensibles al contexto.

No explica que BNF usa recursión, EBNF usa repetición.

4. R+

No especifica los momentos de alocación, cuándo se aloca y cuándo se desaloca cada variable.

No indica que la estática c tiene como momento de alocación todo el programa, desde el inicio hasta el final.

No menciona que la global c está enmascarada por la c de main, en su alcance.

Entre las características no indica el r-valor de cada una.