Examen Ordinaria DLP 2022-23

# Ejercicio 1

Dada la siguiente gramática, dibuje el árbol de sintaxis de la siguiente entrada: “**3+2\*3+(5\*2)”**

expression: expression (‘+’ | ‘-’) expression

| ‘(’ expression ‘)’

| expression (‘\*’ | ‘/’)

| expression

| INT\_CONSTANT

# Ejercicio 2

Crear una gramática en BNF que represente el siguiente subconjunto JSON:

* Un documento JSON está formado por un valor, que puede ser un string, un array o un objeto.
* Los arrays están formados por uno o más valores entre corchetes y separados por comas.
* Un objeto está formado por uno o más campos entre llaves y separados por comas. Un campo consiste en una clave (string) seguido de dos puntos y un valor.

# Ejercicio 3

Dada la siguiente AG:

(1) Program: program → expresión\*

(2) Arithmetic: expression1 → expression2 expression3

(3) FuncInvocation: expression1 → expression2 expression3\*

(4) Variable: expression → ID

(5) IntLiteral: expression → INT\_CONSTANT

Defina una AG para anotar todos los nodos de la expresión AST con un atributo “distancia” que almacene un número entero con la distancia de ese nodo al raíz del mismo (Programa). De este modo, el nodo raíz de una expresión tendrá distancia = 1, los hijos de éste distancia=2, sus nietos distancia=3…

# Ejercicio 4

Dada la siguiente AG:

(1) Arithmetic: expression1 → expression2 expression3

(2) FuncInvocation: expression1 → expression2 expression3\*

(3) FieldAccess: expression1 → expression2 ID

(4) Variable: expression → ID

(5) IntLiteral: expression → INT\_CONSTANT

Defina una AG para comprobar los tipos de este lenguaje (los métodos ya están previamente definidos).

# Ejercicio 5

Dadas las siguientes produciones de la AG:

(1) FuncInvocation: expression1 → expression2 expression3\*

(2) FuncInvocation: statement → expression1 expression2\*

Definir las plantillas de Código. El uso de parámetros deberá ser por valor y no existe conversión de los argumentos a los parámetros.