Práctica. Tercera Fase

Desarrollo de constructores de ASTs para Tiny(0) y para Tiny(1)

En esta tercera parte debe realizarse el siguiente trabajo:

1) Desarrollo manual de un constructor de ASTs para **Tiny(0)**. Para ello, deberá entregarse:
   - Una memoria con las siguientes secciones:
     - Portada en la que aparezcan los nombres y apellidos de los integrantes del grupo, y el número de grupo.
     - Especificación de la sintaxis abstracta de **Tiny(0)** mediante la enumeración de las signaturas (cabeceras) de las funciones constructoras de ASTs.
     - Especificación del constructor de ASTs mediante una gramática s-atribuida.
     - Acondicionamiento de dicha especificación para permitir la implementación descendente.
   - Una implementación manual, en Java, del constructor de ASTs, extendiendo, para ello, el analizador descendente predictivo recursivo desarrollado en la segunda fase.
   - Un procesamiento que realice una impresión básica del programa leído.
   - Un programa principal que combine ambos componentes para (i) construir el AST asociado con un programa TINY(0); y (ii) realizar una impresión básica del programa leído mediante el procesamiento del AST construido. El programa deberá recibir como primer argumento en la línea de comandos el archivo a procesar. Como resultado imprimirá, bien un mensaje legible del primer error detectado, bien una impresión del programa leído en caso de que no se hayan detectado errores.
2) Desarrollo de constructores de ASTs descendentes y ascendentes para **TINY(1)**. Para ello, deberá entregarse:
   - Una memoria con las siguientes secciones:
     - Portada en la que aparezcan los nombres y apellidos de los integrantes del grupo, y el número de grupo.
     - Especificación de la sintaxis abstracta de **Tiny(1)** mediante la enumeración de las signaturas (cabeceras) de las funciones constructoras de ASTs.
     - Especificación del constructor de ASTs mediante una gramática s-atribuida.
     - Acondicionamiento de dicha especificación para permitir la implementación descendente.
   - Una implementación descendente del constructor de ASTs desarrollada con JavaCC.
   - Una implementación ascendente de dicho constructor desarrollada con CUP y JFlex.
   - Un procesamiento que realice una impresión básica del programa leído.
   - Un programa principal que integre ambos constructores, y también el procesamiento desarrollado. Dicho programa recibirá como argumentos (i) el archivo a analizar; (ii) una opción *op* que indique el constructor de ASTs a aplicar (si *op* es `desc` el constructor a aplicar será el descendente; si es `asc` será el ascendente). El programa producirá como salida, bien un mensaje legible del primer error (léxico o sintáctico) detectado, bien una impresión del programa leído en caso de que no se hayan detectado errores.

Fecha límite de entrega: **Domingo 1 de mayo de 2021, a las 11:59 pm**.

Modo de entrega: A través del campus virtual, en un único .zip. Dicho archivo debe contener: (i) un documento PDF `memoria_tiny_0.pdf` con la memoria requerida en el punto 1) del trabajo a realizar, y otro documento PDF `memoria_tiny_1.pdf` con la memoria requerida en el punto 2); (ii) una carpeta `implementación_tiny0`, en el interior de la cuál debe incluirse toda la implementación requerida para **Tiny(0)**; (iii) una carpeta `implementación_tiny1`, en el interior de la cuál debe incluirse la implementación requerida para **Tiny(1)**; (iv) una carpeta `pruebas_tiny_0` con distintos programas de prueba que permitan probar la implementación para **Tiny(0)**; y (v) una carpeta `pruebas_tiny_1` con distintos programas de prueba que permitan probar la implementación para **Tiny(1)**. La entrega debe ser realizada solamente por un miembro del grupo.