**Laboratorio A - Analizador sintáctico descendente recursivo**

**Mini-C# y Mini-Java**

Fecha de entrega: 10 de septiembre, 18:45 horas

**El objetivo**

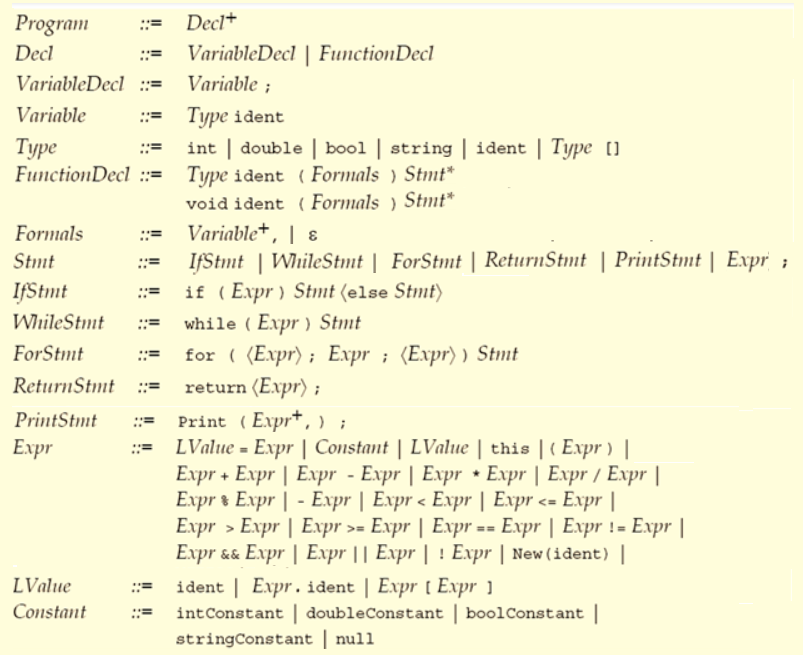
Este laboratorio consistirá en analizar sintácticamente un programa escrito en el lenguaje asignado, implementando el método de análisis sintáctico descendente recursivo. Deberán hacer uso de su analizador léxico de la fase anterior. La finalidad es poder determinar que el programa fuente escrito está sintácticamente correcto utilizando este método.

**Estructura Lexicográfica**

Todas las especificaciones de la fase #1, con el agregado de la palabra reservada **Print** para ambos lenguajes.

**Gramática de Mini-C# y Mini-Java**

La gramática base es la siguiente:



* x significa que x es un terminal, un token devuelto por el analizador léxico. Los terminales están en minúscula excepto aquellos que son palabras reservadas
* *x* en itálico es un no terminal. Todos los no terminales tienen la primera letra en mayúscula
* <x> significa cero o una ocurrencia de x, es decir, x es opcional
* x\* significa cero o más ocurrencias de x
* x+ significa una o más ocurrencias de x
* | significa las alternativas de las producciones
* ε significa épsilon, la ausencia de tokens.

Tome en cuenta que esta gramática no está en una notación usual de definición. Por ejemplo, las producciones con \*, +, <> deberá modificarlas para implementar la gramática en su analizador. Puede rediseñar la gramática si lo considera pertinente, solo asegúrese de reconocer el mismo lenguaje que reconoce la gramática base.

**Implementación**

1. Para este laboratorio, se le asignará una combinación de sentencias a implementar en el lenguaje asignado. Solo deberá implementar en su analizador las sentencias que se le asignen.
2. El programa lee un archivo de entrada en el lenguaje asignado y debe indicar en pantalla únicamente si está sintácticamente correcto o no. No debe realizar ninguna acción adicional si está correcto.
3. De encontrar algún error, el escáner debe continuar hasta el final de archivo. Los errores deberá reportarlos en pantalla en un formato descriptivo, indicando el token incorrecto o el error encontrado.

**Requerimientos generales**

* El ejecutable debe ser llamado minic o minij, debe pedir el archivo de entrada; si es correcto indicarlo en pantalla y si es incorrecto mostrar los errores en pantalla
* El proyecto se entrega en el mismo repositorio en GitHub de la fase anterior, sobre la rama master o principal. Por tanto, se deberá crear una rama con su analizador léxico y una nueva donde incorpore esta fase. Se calificará el proyecto hasta el último commit realizado antes de la hora de entrega. Se revisará el historial del repositorio. No se aceptarán cambios en el momento de la calificación.
* En el repositorio deberá incluir un archivo README.txt donde se detalle cómo su analizador maneja los errores.

**¡Buena suerte!**