**Definiendo reglas sintácticas**

Para el análisis sintáctico, las **terminales** serán los tokens generados en el análisis lexicográfico. Regularmente se escriben con inicio en mayúscula para diferenciarlos de los no terminales.

*Por ejemplo:*

terminal Comillas, T\_dato, Cadena, If, Else, Do, While, For,

Igual, Suma, Resta, Multiplicacion, Division, Op\_logico, Op\_relacional,

Op\_atribucion, Op\_incremento, Op\_booleano, Parentesis\_a, Parentesis\_c,

Llave\_a, Llave\_c, Corchete\_a, Corchete\_c, Main, P\_coma, Identificador, Frase

Numero, ERROR;

Los **no terminales** son los nombres para los axiomas que permiten generar las producciones. Se deberán de construir de acuerdo con las producciones que requiera el lenguaje. Regularmente se escriben con mayúsculas para diferenciarlos de los terminales.

*Por ejemplo:*

non terminal INICIO, SENTENCIA, DECLARACION, DECLARACION\_FOR, IF, IF\_ELSE,

WHILE, DO\_WHILE, FOR, SENTENCIA\_BOOLEANA, SENTENCIA\_FOR;

Uno de los no terminales señalará el **inicio** de la secuencia para el análisis sintáctico.

start with INICIO;

**Actividad**

**Instrucciones**: Genera las siguiente producciones o reglas sintácticas, en su caso la explicación correspondiente. Para indicar que se terminó el axioma usa un punto y coma. Respeta el uso de mayúsculas y minúsculas conforme a los terminales y no terminales definidos previamente.

***Ejemplo***

INICIO ::=

T\_dato Main Parentesis\_a Parentesis\_c Llave\_a SENTENCIA Llave\_c |

Main Parentesis\_a Parentesis\_c Llave\_a SENTENCIA Llave\_c

;

***Ejercicios***

//1. La sentencia puede contener una estructura antecedida de una sentencia o sin sentencia. Las estructuras para considerar son: declaración, if, if\_else, while, do\_while, for

SENTENCIA ::=

SENTENCIA DECLARACION | DECLARACION |

SENTENCIA IF | IF |

SENTENCIA IF\_ELSE | IF\_ELSE |

SENTENCIA WHILE | WHILE |

SENTENCIA DO\_WHILE | DO\_WHILE |

SENTENCIA FOR | FOR

;

//2. La declaración puede ser definida por tipo de dato o cadena, después debe llevar el identificador, al final puede o no llevar una asignación a un número a una frase según sea el caso.

DECLARACION ::=

T\_DATO IDENTIFICADOR Op\_atribucion NUMERO |

T\_DATO IDENTIFICADOR |

CADENA IDENTIFICADOR Op\_ atribucion FRASE |

CADENA IDENTIFICADOR

;

//3. El if debe contener una sentencia booleana entre paréntesis y una sentencia entre llaves.

IF ::=

Parentesis\_a SENTENCIA BOOLEANA Parentesis\_c Llave\_a SENTENCIA Llave\_c

;

//4. La sentencia booleana debe contener un operador booleando o un identificador seguido de un operador relacional con un operador booleano o un número o un identificador o comillas más identificador más comillas o comillas más comillas

SENTENCIA\_BOOLEANA ::=

Op\_booleano |

Identificador Op\_relacional Op\_booleano |

Identificador Op\_relacional Numero |

Identificador Op\_relacional Identificador |

Identificador Op\_relacional Comillas Comillas |

Identificador Op\_relacional Comillas Identificador Comillas

;

//5. Es parecido al if, pero incluye la parte del else

IF\_ELSE ::=

If Paréntesis\_a SENTENCIA\_BOOLEANA Paréntesis\_c

Llave\_a SENTENCIA Llave\_c

Else Llave\_a SENTECIA Llave\_c

;

//6. Utilizar la estructura de C++ para while

WHILE ::=

While Paréntesis\_a SENTENCIA\_BOOLEANA Paréntesis\_c

Llave\_a SENTENCIA Llave\_c

;

//7. Utilizar la estructura de C++ para do while

DO\_WHILE ::=

Do Llave\_a SENTENCIA Llave\_c

While Paréntesis\_a SENTENCIA\_BOOLEANA Paréntesis\_c

;

//8. Utilizar la estructura de C++ para for

FOR ::= For Parentesis\_a SENTENCIA\_FOR Parentesis\_c Llave\_a SENTENCIA Llave\_c

;

//9. Las dos primeras secciones, usar lo que se tiene hasta ahora y la última sección usar un nuevo axioma llamado declaración\_for. Considera que en la primera sección se puede declarar o no la variable.

SENTENCIA\_FOR ::=

T\_dato Identificador Igual Numero P\_coma SENTENCIA\_BOOLEANA P\_coma DECLARACION\_FOR |

Identificador Igual Numero P\_coma SENTENCIA\_BOOLEANA P\_coma DECLARACION\_FOR

;

//10. Es la tercera sección del for. Considera que puede usarse el operador de incremento o algún operador de atribución.

DECLARACION\_FOR ::=

Identificador Op\_atribucion Numero |

Identificador Op\_incremento |

Op\_incremento Identificador

;