Se proporciona a continuación una gramática en notación ECMAScript que describe una porción reducida y adaptada del lenguaje JavaScript.

```
Program ::= { Statement }
 Statement ::= WhileStatement | AssignmentStatement | ConsoleStatement
 WhileStatement ::= "while" "(" Expression ")" Block
 AssignmentStatement ::= Identifier "=" Expression ";"
 ConsoleStatement ::= "console" "." "log" "(" Expression ")" ";"
 Block ::= "{" { Statement } "}"
 Expression ::= Term { ( "+" | "-" | "*" | "/" ) Term }
 Term ::= Identifier | Number | "(" Expression ")"
 Identifier ::= letter { letter | digit | "_" }
 Number ::= digit { digit }
 letter ::= "a" | ... | "z" | "A" | ... | "Z"
 digit ::= "0" | ... | "9"
```

Utilizando ANTLR4 con JavaScript, , implemente un analizador que procese un archivo de entrada (input.txt) con código fuente escrito en este sub-lenguaje de JavaScript.

El analizador deberá realizar las siguientes tareas:

- Análisis léxico y sintáctico: realizar el análisis léxico y sintáctico sobre el código fuente e informar si la entrada es correcta o contiene errores. En caso de errores, indicar la línea en la que ocurren y la causa del problema.
- Tabla de lexemas-tokens: Generar una tabla que contenga los lexemas y sus respectivos tokens reconocidos durante el análisis léxico.
- Árbol de análisis sintáctico: Construir y mostrar el árbol de análisis sintáctico concreto de la entrada. Puede representarse en formato de texto.
- Interpretación: Mostrar en la salida el código fuente (input.txt) en lenguaje JavaScript y ejecutarlo como lo haría un intérprete básico.