```
CATEGORIES
expression;
statement;
definition;
type;

NODES
program -> definition*;

varDefinition:definition -> string type;
structDefinition:definition -> string attrDefinition*;
functionDefinition:definition -> string params:varDefinition* type?
definitions:varDefinition* statement*;

attrDefinition -> string type;

read:statement -> expression;
print:statement -> expression*;
println:statement -> expression*;
printsp:statement -> expression*;
return:statement -> expression?;
assignment:statement -> left:expression right:expression;
while:statement -> expression statement*;
ifelse:statement -> cond:expression tr:statement* fs:statement*;
functionCallStatement:statement -> string expression*;

intLiteral:expression -> int;
floatLiteral:expression -> float;
charLiteral:expression -> string;
arrayAccess:expression -> expr1:expression expr2:expression;
fieldAccess:expression -> expr:expression string;
not:expression -> expression;
logic:expression -> left:expression operator:string right:expression;
arithmetic:expression -> left:expression operator:string
right:expression;
variable:expression -> string;
cast:expression -> type expression;
functionCallExpression:expression -> string expression*;

intType:type -> ;
floatType:type -> ;
charType:type -> ;
arrayType:type -> int type;
structType:type -> string;
voidType:type -> ;
errorType:type ->;

// ----------------------
ATTRIBUTE GRAMMAR Identification

varDefinition -> scope:int;
variable -> varDefinition;
functionCallStatement -> functionDefinition;
functionCallExpression -> functionDefinition;
```

```
attrDefinition -> structDefinition;
fieldAccess -> attrDefinition;
structType -> structDefinition;
arrayAccess -> varDefinition;

// -----------------------
ATTRIBUTE GRAMMAR TypeChecking

expression -> expressionType:type;
expression -> lvalue:boolean;

// -----------------------
ATTRIBUTE GRAMMAR MemoryAllocation
definition -> [inh] address:int;        // Inherited attribute
attrDefinition -> [inh] address:int;    // Inherited attribute


// -----------------------
CODE SPECIFICATION Mapl

run[program]
execute[statement]
value[expression]
address[expression]

metadata[program]
metadata[varDefinition]
```