Comănac Dragoș-Mihail

https://github.com/ComanacDragos/ToyLanguageCompiler

Statement: **Implement a parser algorithm (cont.)**

PART 3: Deliverables

1. Algorithms corresponding to *parsing table* (if needed) and *parsing strategy*

2. Class *ParserOutput* - DS and operations corresponding to choice 2.a/2.b/2.c (Lab 5) (required operations: transform parsing tree into representation; print DS to screen and to file)

Implementation

The following classes are added:

Node
  Long id = nextId++;
  Symbol value;
  List<Node> children;

public Node(Symbol value, List<Symbol> symbols) // creates recursively the tree

public class TreeGenerator
  Node root; // root of the tree
  Long nextId = 0L; // id generator for nodes
  Deque<NonTerminalWithProduction> productionStack;
  Grammar grammar;

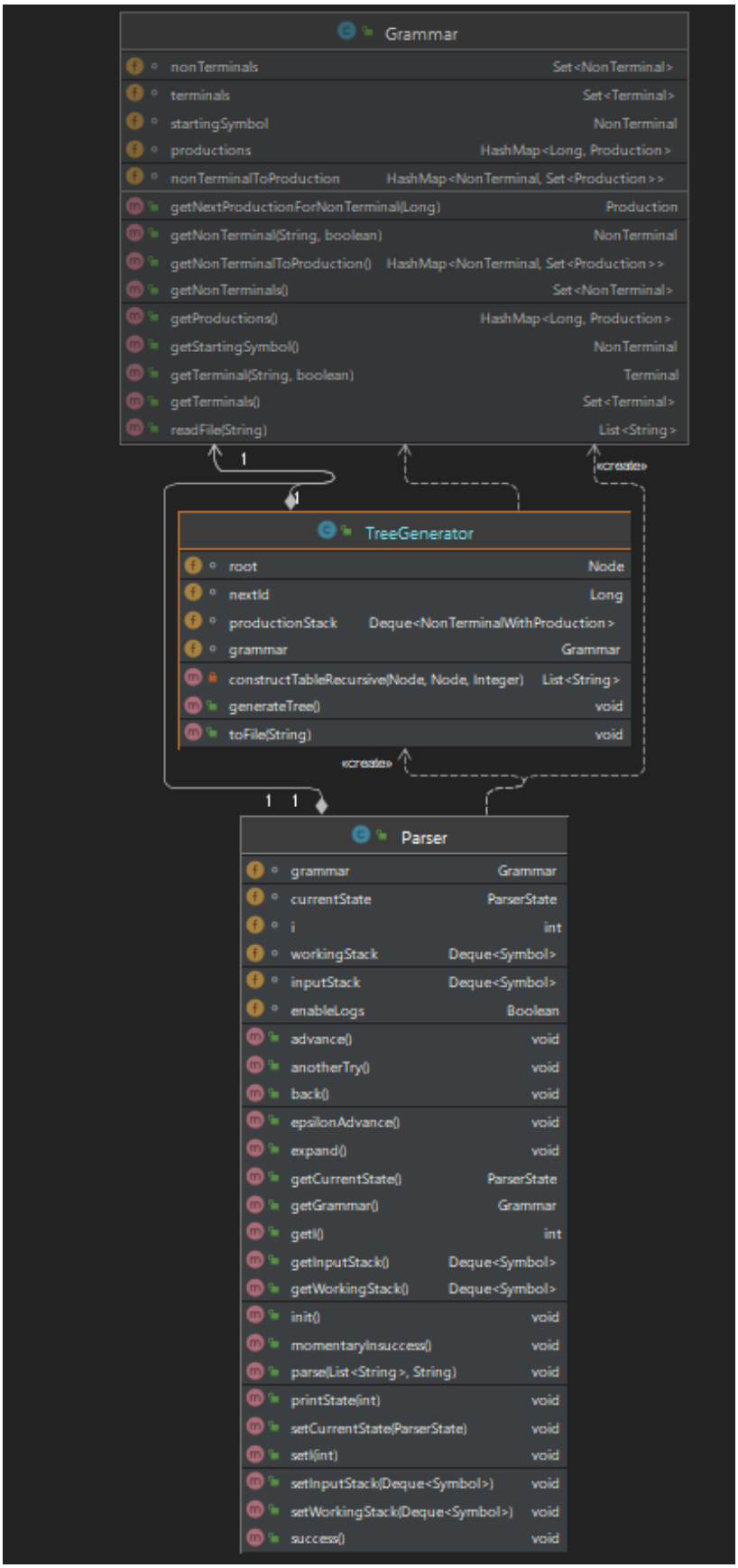
public TreeGenerator(Deque<Symbol> workingStack, Grammar grammar) // instantiate the productionStack from a given workingStack so that it contains only the NonTerminalWithProduction classes

public void generateTree() // begins the generation of the tree

public void toFile(String outputDir) // writes to an output directory the table

private List<String> constructTableRecursive(Node currentNode, Node fatherNode, Integer positionRelativeToFather) // constructs the string representation of the table recursively

Comănac Dragoș-Mihail



Comănac Dragoș-Mihail

Testing

S A B
a b
S
S ::= a B | b A
A ::= a | a S | b A A
B ::= epsilon | b | b S | a B B

id	value	father	right-sibling
0	S	-1	-1
1	b	0	2
2	A	0	-1
3	a	2	4
4	S	2	-1
5	a	4	6
6	B	4	-1
7	b	6	8
8	S	6	-1
9	a	8	10
10	B	8	-1
11	a	10	12
12	B	10	14
13	epsilon	12	-1
14	B	10	-1
15	b	14	-1

Comănac Dragoș-Mihail

program statement_list statement simple_statement compound_statement simple_type array_type type expression binary_operator unary_operator declaration_statement iostatement assignment_statement if_statement else_branch while_statement expression' expression_simple id constant int char bool string float >> << while if else and or ! + - * / % > < >= <= != == = ; [ ] { } ( ) , ^
program
program ::= statement_list
statement_list ::= statement | statement statement_list
statement ::= simple_statement | compound_statement

simple_statement ::= assignment_statement ; | iostatement ; | declaration_statement ;

compound_statement ::= if_statement | while_statement

simple_type ::= bool | char | int | string | float

array_type ::= simple_type [constant]

type ::= simple_type | array_type

expression_simple ::= constant | id | id [constant] | id [id] | unary_operator expression | (expression)

expression' ::= binary_operator expression expression' | epsilon

expression ::= expression_simple expression'

declaration_statement ::= type id | type id = expression

iostatement ::= << id | << id [constant] | << id [id] | >> expression

assignment_statement ::= id = expression

if_statement ::= if (expression) { statement_list } else_branch
else_branch ::= epsilon | else { statement_list }

while_statement ::= while (expression) { statement_list }

unary_operator ::= !
binary_operator ::= + | - | * | / | ^ | % | and | or | > | < | >= | <=| != | ==

Comănac Dragoș-Mihail

id	value	father	right-sibling
0	program	-1	-1
1	statement_list	0	-1
2	statement	1	14
3	simple_statement	2	-1
4	declaration_statement	3	13
5	type	4	12
6	array_type	5	-1
7	simple_type	6	9
8	int	7	-1
9	[	6	10
10	constant	6	11
11	]	6	-1
12	id	4	-1
13	;	3	-1
14	statement_list	1	-1
15	statement	14	-1
16	simple_statement	15	-1
17	declaration_statement	16	28
18	type	17	21
19	simple_type	18	-1
20	int	19	-1
21	id	17	22
22	=	17	23
23	expression	17	-1
24	expression_simple	23	26
25	constant	24	-1
26	expression'	23	-1
27	epsilon	26	-1
28	;	16	-1