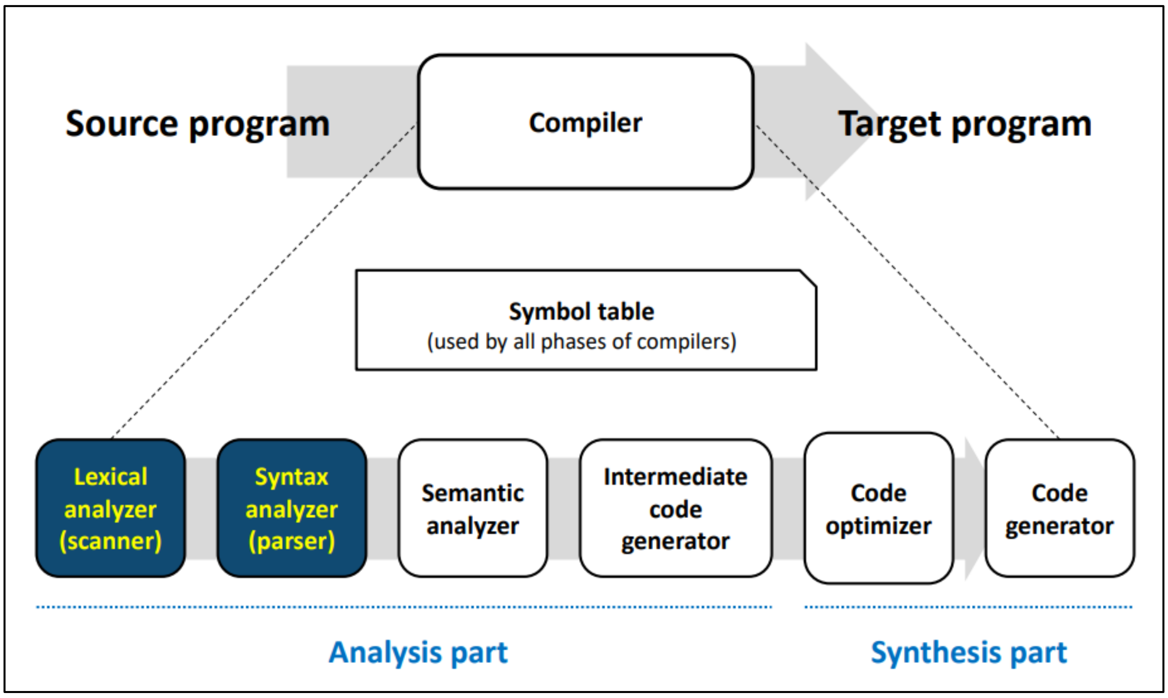

Compiler Homework #2

SYNTAX ANALYZER

Team 1
Leo PAOL – Christophe MEI
50181573 - 50181587

CGF changes

We did an elimination of the epsilons in the statements.

01 : CODE→ VDECL CODE | FDECL CODE | VDECL | FDECL

03 : FDECL → vtype id lparen ARG rparen lbrace BLOCK RETURN rbrace |
vtype id lparen rparen lbrace BLOCK RETURN rbrace |
vtype id lparen ARG rparen lbrace RETURN rbrace |
vtype id lparen rparen lbrace RETURN rbrace

04 : ARG → vtype id MOREARGS | vtype id

05 : MOREARGS → comma vtype id MOREARGS | vtype id

06 : BLOCK → STMT BLOCK | STMT

08 : STMT → if lparen COND rparen lbrace BLOCK rbrace else lbrace BLOCK rbrace |
if lparen COND rparen lbrace rbrace else lbrace BLOCK rbrace |
if lparen COND rparen lbrace BLOCK rbrace else lbrace rbrace |
if lparen COND rparen lbrace rbrace else lbrace rbrace

09 : STMT → while lparen COND rparen lbrace BLOCK rbrace |
while lparen COND rparen lbrace rbrace

How it works

First, we read the output of the lexical analyser and convert the token names into the names used in the CFG given with the homework subject. We fill the array "tokens".

Then we enter in the main function called "check_syntax". In this function we use the array "slr" representing the SLR table and the array "cfg" representing the CFG to shift and reduce our terminals. To achieve this, we use the list "stack", the variable "current_state" representing the line to read in the SLR table, the well-called "next_sym" variable and the "shift_cursor" index.

We use all of this like it has been seen during class.

Testing

Test files 1, 2 and 3 are valid.

Test files 4, 5 and 6 are invalid.

Test 4 : initialisation of variable is not possible during declaration.

Test 5 : if statement must have else.

Test 6 : it's not possible to have cumulative parenthesis.