Tokens

You can define tokens in the first section of the Yacc file.

- %token ID
- %token CLASS

To determine precedence and associativity, you may use either of the following, instead of the above method.

- %left IF
- %right PROGRAM
- %nonassoc ELSE

You do not need either the %right or the %nonassoc associativity in this project.

The later a token is defined, the higher priority it will have.

- %left ADDITION SUBTRACTION
- %left MULTIPLICATION DIVISION

Grammar rules

You can define grammar rules in the second section of the Yacc file.

- statements_list : statements_list SEMICOLON statement | statement

C code blocks can be added to be executed **right before shifting the next token**.

- statements_list : statements_list { printf(“Hi.”); } SEMICOLON statement
  | statement { printf(“Hi.”); }

λ or ε have no dedicated symbols. Just leave empty space instead.

- dec_list : dec | declist dec |

You may use comments for better readability (.,.)

- dec_list : dec | declist dec | /* Lambda */

The starting rule is determined by defining it as %start in the first section of the Yacc file.

- %start program