\\ BNF language description

<program> ::= <stmtList>

<stmtList> ::= <stmt> <stmtList> | <stmt> |<epsilon>

<stmt> ::= <declaration> “;” | <assignment> “;” | <printStmt> |
<iterative> | <block> | <conditional> “;”

<declaration> ::= “int” <id> | “bool” <id> | “int” <id> = <expression> |
“bool” <id> = <expression>

<assignment> ::= <id>”=”<expression> | “++”<id> | “--"<id> | <id>”++” | <id>”--”

<printStmt> ::= “print” <expression>

<block> ::= “{“ <stmtList> “}”

<conditional> ::= “if” <expression> “then” <block> |
“if” <expression> “then” <block> “else” <block>

<iterative> ::= “while” <expression> <block> |
“for” <declaration> “;” <expression> “;” <assignment> <block>

<expression> ::= <expression> “or” <conjunction> | <conjunction>

<conjunction> ::= <conjunction> “and” <equality> | <equality>

<equality> ::= <equality> “=” <comparator> |<equality> “!=” <comparator> |
<comparator>

<comparator> ::= <add_sub> “>” <add_sub> |< add_sub > “<” < add_sub > |
< add_sub >

< add_sub > ::= < add_sub > “+” < product > | < add_sub > “-” <product> |
<product>

<product> ::= <product> “*” <negation> | <product> “/” <negation> |
<product> “mod” <negation> | <negation>

<negation> ::= “not” “(“<negation >”)” | “-” < negation > | <exponent>

<exponent> ::= <base> “^” <exponent> | <base>

<base> ::= <integer> |<boolean> |“(“<Expr>”)” |
“|” <expression> “|” | <PrePost> | <id>

<PrePost> ::= “++” <id> | “--” <id> | <id> “++” | <id> ”--”

<Epsilon> ::=

\\ Regular expression definition -
id
integer
boolean

\\ Reserved symbols = {
“;”,
“=”,
“print”,
“int”,
“bool”,
“if”,
“then”,
“else”,
“while”,
“for”,
“not”,
“div”,
“mod”,
“and”,
“or”,
“{“,
“}”,
“(“,
“)”,
“++”,
“--”,
“<”,
“>”,
“!=”,
“-”,
“+”,
“|”,
“^”,
“*”
}

\\ Regular expression description -

alpha = [A-Za-z]

alphanumeric = [A-Za-z0-9_]
pos_integer = [1-9]
non_neg_int = [0-9]

id = {alpha}{alphanumeric}*
boolean = "true" | "false"
integer = “0” | {pos_integer}{non_neg_int}*

\\ Operator precedence and associativity -

Precedence in order of highest to lowest
The lower the number the higher the precedence

Description	Operator	Precedence	associativity
post-increment	++	None	left
post-decrement	−−	None	left
pre-increment	++	None	right
pre-decrement	−−	None	right
Parenthesis	()	None	None
Absolute value	|	None	None
Exponent	^	1	right
Negation	not	2	right
unary minus	−	2	right
Multiply	*	3	left
Division	div	3	left
Modulus	mod	3	left
Add	+	4	left
Subtract	−	4	left
Less-than	<	5	left

Greater-than	>	5	left
Equal-to	=	5	left
Not equal-to	!=	5	left
Conjunction	and	6	left
Disjunction	or	7	left