//-- simple operands

%type <real\_t> REAL

%type <real\_t> INT

%type <int\_t> ALPHACHAR

%type <poly\_t> primary

//-- simple operands

//-- one operation between simple operands

%type <int\_t> operation

//-- possible operations between simple operands

%type <poly\_t> expr\_equal

%type <poly\_t> expr\_add

%type <poly\_t> expr\_mul

%type <poly\_t> expr\_pow

// io operator

%type <int\_t> operator

%type <int\_t> out\_operator

%%

/////////////------------ io operators

operator: out\_operator

// out operator

out\_operator: '$'VAR

// out expression

out\_operator: out\_operator '<''<' expr\_equal

// out const string

out\_operator: out\_operator '<''<' CSTR

// out \n

out\_operator: out\_operator '<''<' INT '$'

out\_operator: out\_operator '<''<' '$'

/////////////------------ io operators

///////////////------------///////////---- main block

// empty operator

operation: ';'

// io operator

operation: operator ';'

// expression

operation: expr\_equal ';'

// combination

operation: operation operator ';'

operation: operation expr\_equal ';'

operation: operation ';'

///////////////------------///////////---- main block

////////////////////////////------ expr

// binary and unary operations

//------- priority =

expr\_equal: expr\_add '=' expr\_equal

expr\_equal: expr\_add;

//------- priority =

//------- priority + -

expr\_add: expr\_mul;

expr\_add: expr\_add '+' expr\_mul

expr\_add: expr\_add '-' expr\_mul

//------- priority + -

//------- priority \* /

// it's possible to multiply number on letter: 2a, ab

expr\_mul: expr\_mul ALPHACHAR

expr\_mul: expr\_mul '(' expr\_add ')'

expr\_mul: expr\_pow;

expr\_mul: expr\_mul '\*' expr\_pow

debug\_out\_p($$, "bin '\*'");}

//------- priority \* /

//------- priority ^

expr\_pow: primary;

expr\_pow: primary '^' expr\_pow

debug\_out\_p($$, "bin '^'");}

expr\_pow: '-' primary

debug\_out\_p($$, "unary '-'"); }

expr\_pow: '-' primary '^' expr\_pow

//------- priority ^

//------- highest priority: number, letter

// it's a letter or real/int number

primary: ALPHACHAR

primary: REAL

primary: INT

// it's a variable

primary: VAR

// it's an expression in brackets

primary: '(' expr\_add ')'

////////////////////////////------ expr

%%