Revised Grammar

<program> ::= <fdecls> <declarations> <statement_seq>.

<fdecls> ::= <fdec>;<fdecls_r> | <fdecls_r>
<fdecls_r> ::= <fdec>;<fdecls_r> | Ɛ
<fdec> ::= def <type> <fname> (<params>) <declarations> <statement_seq> fed

<params> ::= <type><var><params_opt> | Ɛ
<params_opt> ::= , <params> | Ɛ
<fname> ::= <id>

<declarations> ::= <decl>;<declarations_r> | <declarations_r>
<declarations_r> ::= <decl>;<declarations_r> | Ɛ
<decl> := <type> <varlist>

<type> := int | double

<varlist> ::= <var><varlist_opt>

<varlist_opt> ::= , <var><varlist_opt> | Ɛ

<statement_seq> ::= <statement><statement_seq_opt>
<statement_seq_opt> ::= ; <statement><statement_seq_opt> | Ɛ
<statement> ::= <var> = <expr> |
if <bexpr> then <statement_seq> <statement_opt> |
while <bexpr> do <statement_seq> od |
print <expr> |
return <expr> |
Ɛ
<statement_opt> ::= fi | else <statement_seq> fi

<expr> ::= <term><expr_r>
<expr_r> ::= + <term><expr_r> | - <term><expr_r> | Ɛ
<term> ::= <factor><term_r>
<term_r> ::= * <factor><term_r> | / <factor><term_r> | % <factor><term_r> | Ɛ
<factor> ::= <id><factor_opt> | <number> | (<expr>)
<factor_opt> ::= (<exprseq>) | [<expr>] | Ɛ

<exprseq> ::= <expr><exprseq_opt> | Ɛ
<exprseq_opt> ::= , <expr><exprseq_opt> | Ɛ

<bexpr> ::= <bterm><bexpr_r>
<bexpr_r> ::= or <bterm><bexpr_r> | Ɛ
<bterm> ::= <bfactor><bterm_r>
<bterm_r> ::= and <bfactor><bterm_r> | Ɛ

<bfactor> ::= (<bfactor_opt> | not <bfactor>
<bfactor_opt> ::= <bexpr>) | <expr><comp><expr>)

<comp> ::= < | > | == | <= | >= | <>

<var> ::= <id><var_opt>
<var_opt> ::= [<expr>] | Ɛ

<number> ::= <integer> | <double>

<letter> ::= a | b | c | ... | z

<digit> ::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0

<id> ::= <letter> | <id><letter> | <id><digit>