A6 Grammar with Left Recursion Elimination

Pgm = kwdprog BBlock
BBlock = brace1 Vargroup Stmts brace2
Vargroup = kwdvars PPvarlist | eps
PPvarlist = parens1 Varlist parens2
Varlist = Vardecl semi Varlist | eps
Vardecl = Basekind Varid
Basekind = 'int' | 'float' | 'string'
Varid = id
Stmts = Stmt semi Stmts | eps
Stmt = Stasgn | Stprint | Stwhile
Stasgn = Varid equal Expr
Stprint = kprint PPexprs
Stwhile = kwdwhile PPexpr1 BBlock
PPexprs = parens1 Exprlist parens2
PPexpr1 = parens1 Expr parens2
Exprlist = Expr Moreexprs
Moreexprs = comma Exprlist | eps
Expr Before LRE:
-Expr = Expr Oprel Rterm | Rterm
After LRE:
-Expr = Rterm QExpr
-QExpr = Oprel Rterm QExpr | eps
Rterm Before LRE:
-Rterm = Rterm Opadd Term | Term
After LRE:
-Rterm = Term QRterm
-QRterm = Opadd Term QRterm | eps
Term Before LRE:
-Term = Term Opmul Fact | Fact
After LRE:
-Term = Fact QTerm
-QTerm = Opmul Fact QTerm | eps
Fact = int | float | string | Varid | PPexpr1
Oprel =
opeq | opne | Lthan | ople | opge | Gthan
Lthan = angle1
Gthan = angle2
Opadd = plus | minus
Opmul = aster | slash | caret

First Sets For Revised A6 Grammar

F(#1) = { kwdprog }
F(#2) = { brace1 }
F(#3) = { kwdvars }
F(#5) = { parens1 }
F(#6) = {F(#8)} = { 'int' , 'float' , 'string' }
F(#8) = {F(#9) + F(#10) + F(#11)}
= { 'int' , 'float', 'string' }
F(#9) = { 'int' }
F(#10) = { 'float' }
F(#11) = { 'string' }
F(#12) = {id}
F(#13) = {F(#15)} = { id }
F(#15) = {F(#18)} = { id }
F(#16) = {F(#19)} = { kprint }
F(#17) = {F(#20)} = { while }
F(#18) = {F(#12)} = { id }
F(#19) = { kprint }
F(#20) = { kwdwhile }
F(#21) = { parens1}
F(#22) = { parens1}
F(#23) = { F(#32)} =
{int, float, string, id, parens1}
F(#24) = {comma}
F(#26) = {F(#32)} =
{int, float, string, id, parens1}

F(#27) =
{F(#40)+ F(#41) + F(#42) + F(#43) + F(#44)
+ F(#45} =
{opeq, opne, angle1, ople, angle2}

F(#29) = {F(#32)} =
{int, float, string, id, parens1}
F(#30) = {F(#48)+ F(#49)} = { plus , minus}

F(#32) =
{F(#35) + F(#36) + F(#37) + F(#38) +
F(#39)} =
{int, float, string, F(#12) + F(#22) } =
{ int, float, string, id, parens1 }

F(#33) = { F(#50) + F(#51) + F(#52) } =
{aster, slash, caret}

F(#35) = {int}
F(#36) = {float}
F(#37) = {string}
F(#38) = {F(#12)} = { id}
F(#39) = {F(#23)} = {parens1}
F(#40) = {opeq}
F(#41) = {opne}
F(#42) = {F(#46)} = { angle1 }
F(#43) = { ople }
F(#44) = { opge }
F(#45) = {F(#47)} = { angle2 }
F(#46) = {angle1}
F(#47) = {angle2}
F(#48) = {plus}
F(#49) = {minus}
F(#50) = {aster}
F(#51) = {slash}
F(#52) = {caret}

Follow sets for LHS Non-Terminal Symbols

W(Vardecl) = { semi }
W(Basekind) = {F(Varid)} = { id }
W(Stmt) = { semi }
W(Stasgn) = {W(Stmt) } = { semi }
W(Strprint) = {W(Stmt)} = { semi }
W(Stwhile) = {W(Stmt)} = { semi }
W(Varid) = { equal , W(Vardecl)} =
{ equal, semi }

W(Expr) = { parens2 , F(Moreexprs),
W(Moreexprs), W(Stasgn) } = { comma,
parens2, semi }

W(Rterm) = {F(QExpr) , W(QExpr) } =
{ opeq, opne, angle1, ople, angle2, opge,
comma, parens2, semi }

W(Term) = {F(QRterm) , W(QRterm) } =
{ plus, minus, opeq, opne, angle1, ople,
angle2, opge, comma, parens2, semi }

W(Oprel) = {F(Rterm)} = { int, float, string,
id, parens1}

W(Opadd) = { F(Term)} = {int, float, string,
id, parens1 }

W(Fact) = {F(Qterm), W(Qterm} =
{aster, slash, caret, plus, minus, opeq,
opne, angle1, ople, angle2, opge, comma,
parens2, semi }

W(Opmul) = {F(Fact)} = {int, float, string, id,
parens1}

W(PPexpr1) = {F(BBlock, W(Fact)} =
{brace1, aster, slash, caret, plus, minus,
opeq, opne, angle1, ople, angle2, opge,
comma, parens2, semi }

W(Lthan) = {Oprel}= { int, float, string, id,
parens1}

W(Gthan) = {Oprel} = { int, float, string, id,
parens1}

W(Varid) = {F(equal), w(Vardecl)} = {semi,
equal}