

## Convert the given grammar to LL(1) Grammar.

### Grammar:

*Function* → *Type identifier ( ArgList ) CompoundStmt*  
*ArgList* → *Arg | ArgList , Arg*  
*Arg* → *Type identifier*  
*Declaration* → *Type IdentList ::*  
*Type* → *Adadi | Ashriya | Harf | Matn | Mantiqi*  
*IdentList* → *identifier , IdentList | identifier*  
*Stmt* → *ForStmt | WhileStmt | Expr :: | IfStmt | CompStmt | Declaration | ::*  
*ForStmt* → *for ( Expr :: OptExpr :: OptExpr ) Stmt*  
*OptExpr* → *Expr |  $\wedge$*   
*WhileStmt* → *while ( Expr ) Stmt*  
*IfStmt* → *Agar ( Expr ) Stmt ElsePart*  
*ElsePart* → *Wagarna Stmt |  $\wedge$*   
*CompStmt* → *{ StmtList }*  
*StmtList* → *StmtList Stmt |  $\wedge$*   
*Expr* → *identifier := Expr | Rvalue*  
*Rvalue* → *Rvalue Compare Mag | Mag*  
*Compare* → *== | < | > | <= | >= | != | <>*  
*Mag* → *Mag + Term | Mag - Term | Term*  
*Term* → *Term \* Factor | Term / Factor | Factor*  
*Factor* → *( Expr ) | identifier | number*