GRAMMAR BAIL-IN

```
Program          := Block*

Block            := @ TypeList @ Identifier ( ( ε | DeclarationList ) ) { CommandList return ExpressionList ; }

CommandList      := Command*

Command          := if $ Expression $ { CommandList } ( ε | else { CommandList } )
                    | loop $ ( ε | Expression ) $ { CommandList }
                    | Declaration ;
                    | Assignment ;
                    | Function ;

DeclarationList  := Declaration ( : Declaration )*

Declaration      := Type Identifier ( ε | [ IntegerLiteral ] ) ( , Identifier ( ε | [ IntegerLiteral ] ) )*

ExpressionList   := Expression ( , Expression )*

Assignment       := IdentifierList <- ExpressionList

IdentifierList   := IdentifierItem ( , IdentifierItem )*

IdentifierItem   := Identifier ( ε | [ Expression ] )

Expression       := Quaternary ( ε | OperatorQ Expression )

Quaternary       := Tertiary ( ε | OperatorT Quaternary )

Tertiary         := Secondary ( ε | OperatorS Tertiary )

Secondary        := Primary ( ε | OperatorP Secondary )

Primary          := IdentifierItem
                    | IntegerLiteral
                    | (Expression)
                    | Function

Function         := @ ( readBool | writeBool | readChar | writeChar | readInt | writeInt | Identifier ) ( ( ε | ExpressionList ) )

Operator         := ++ | -- | ** | // | %% | ^^ | >= | <= | << | >> | == | <> | && | || | ## | !& | !| | !#

TypeList         := Type ( , Type )*

Type             := int
                    | bool

IntegerLiteral   := A-Z | a-z | 0-9 | TRUE | FALSE
```