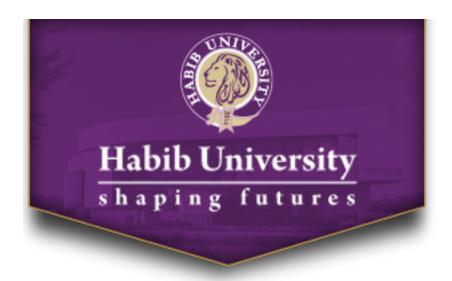
HABIB UNIVERSITY

CS 421 (COMPILER DESIGN AND CONSTRUCTION) - FALL 2022

Homework 2

Submitted by: Fizza Rubab fr06161 Iqra Siddiqui is06176

October 15, 2022



LL(1) GRAMMAR

```
Program \rightarrow dt id(ParamList)Stmts
ParamList \rightarrow dt id Plist
PList \rightarrow dt id PList \mid \epsilon
Stmts \rightarrow Stmts'
Stmts' → DecStmt Stmts' | AssignStmt Stmts' | ForStmt Stmts' | IfStmt Stmts' | ReturnStmt
Stmts' \epsilon
DecStmt \rightarrow dt id OptionalAssign List
List \rightarrow, dt Optional Assign List | \epsilon
OptionalAssign \rightarrow = \text{Expr}; |\epsilon|
Expr \rightarrow TE'
E' \rightarrow +TE' | \epsilon
T \rightarrow FT
T' \rightarrow *FT' \mid \epsilon
F \rightarrow (Expr) \mid id
For Stmt \rightarrow for (Type id Expr; Expr relop Expr; id++)Stmts
Type \rightarrow dt \mid \epsilon
If Stmt \rightarrow if (Expr relop Expr) Stmts Optional Else
AssignStmt \rightarrow id = Expr;
OptionalElse \rightarrow else Stmts|\epsilon|
ReturnStmt \rightarrow return Expr;
```

As it can be seen there is no need for left recursion elimination or left factoring required in the grammar. Now we need to confirm whether this grammar is LL(1) or not.

Computing FIRST sets for each non-terminal

```
FIRST(Program) = \{dt\}
FIRST(ParamList) = \{dt\}
FIRST(PList) = \{, \epsilon\}
FIRST(Stmts) = FIRST(Stmts') = FIRST(DecStmt Stmts') \cup FIRST(AssignStmt Stmts')
\cup FIRST(ForStmt Stmts') \cup FIRST(IfStmt Stmts') \cup FIRST(ReturnStmt Stmts') \cup \epsilon
= \{dt\} \cup \{if\} \cup \{for\} \cup \{id\} \cup \{return\} \cup \epsilon = \{dt, if, for, id, return, \epsilon\}
FIRST(Stmts') = \{dt, if, for, id, return, \epsilon\}
FIRST(DecStmt) = \{dt\}
FIRST(List) = FIRST(dt \ Optional Assign \ List) \cup \epsilon = \{,\epsilon\}
FIRST(OptionalAssign) = FIRST(=Expr;) \cup \epsilon = \{=, \epsilon\}
FIRST(Expr) = FIRST(TE') = FIRST(FT') = FIRST(F) = \{(,id)\}
FIRST(E') = FIRST(+TE') \cup \epsilon = \{+, \epsilon\}
FIRST(T) = FIRST(FT') = FIRST(F) = \{(,id)\}
FIRST(T') = FIRST(*FT') \cup \epsilon \{*, \epsilon\}
FIRST(F) = \{(,id)\}
FIRST(ForStmt) = \{for\}
```

```
FIRST(Type) = \{dt, \epsilon\}
FIRST(ifStmt) = \{if\}
FIRST(AssignStmt) = \{id\}
FIRST(OptionalElse) = \{else, \epsilon\}
FIRST(ReturnStmt) = \{return\}
   Computing FOLLOW sets for each Production
FOLLOW(Program) = \{\$\}
FOLLOW(ParamList) = \{\}
FOLLOW(PList) = FOLLOW(ParamList) = \{\}
FOLLOW(Stmts) = \{\}\}
FOLLOW(Stmts') = FOLLOW(Stmts) = \{\}\}
FOLLOW(DecStmts) = (FIRST(Stmts') - \epsilon) \cup FOLLOW(Stmts') \cup FOLLOW(Stmts)
= \{dt, if, for, id, \}\}
FOLLOW(List) = FOLLOW(DecStmts) = \{dt, if, for, id, \}\}
FOLLOW(OptionalAssign) = (FIRST(List) - \epsilon) \cup FOLLOW(List) \cup FOL-
LOW(DecStmts) = \{,,dt,if,for,id,\}\}
FOLLOW(Expr) = \{;,relop,\}
FOLLOW(E') = FOLLOW(Expr) = \{; relop, \}
FOLLOW(T) = FIRST(E') - \epsilon \cup FOLLOW(Expr) = \{+,;,relop,\}
FOLLOW(T') = FOLLOW(T) = \{+,;,relop,\}
FOLLOW(F) = FIRST(T') - \epsilon \cup FOLLOW(T) \cup FOLLOW(T') = \{*,+,;,relop,\}
FOLLOW(ForStmt) = FIRST(Stmts') - \epsilon \cup FOLLOW(Stmts') = \{dt,if,for,id,return,\}\}
FOLLOW(Type) = \{id\}
FOLLOW(IfStmt) = FIRST(Stmts') - \epsilon \cup FOLLOW(Stmts') = \{dt, if, for, id, return, \}\}
FOLLOW(AssignStmt)
                              FIRST(Stmts') - \epsilon \cup FOLLOW(Stmts')
{dt,if,for,id,return,}}
FOLLOW(OptionalElse) = FOLLOW(IfStmt) - \epsilon \cup FOLLOW(Stmts')
{dt,if,for,id,return,}}
                              FIRST(Stmts') - \epsilon \cup FOLLOW(Stmts')
FOLLOW(ReturnStmt)
                         =
{dt,if,for,id,return,}}
```