Grammar - Compiler Construction Group - 29 Shawya Marwah - 2019 B3 A70790 Ruchin Kumbhare - 201985 170650P < fragram > > / program > \$ A shevin Merali - 2019 B2 A70957P Hari Sanker - 2019 B3 A70564 P Food (< program'7) = & DRIVERDEF, DEF, DECLARE & Delip Ventraleth - 2010 A7PS 1203P \* Modulederlandors >> (program > ) / module Declarations > Lother Modules > Lother Modules > Lother Modules > First (program >) = & DRIVERDEF, DEF, DECLARE & 2 malule Declarations 7 > < 10 module Veclaration 7 (module Declarations > / E First (2 module Dedorations >) = & PELLARE, E} Follow (< module Delorations >) = { DEF, DRIVERDEF} < Module Declaration > -> DECLARE MODULE 10 SEMILOL Forst (< module Declaration >) = { DECLARE & Edlow (< module Declaration >) - { < other Modules > < other Modules > 16 Flirst (Zother Modules 7) = & DEF, E& Foollow (Zother Modules >) = & DRIVERDE F & Lower Modele > -> DRIVERDEF DRIVERAPROGRAM DRIVERENDDEF < madule POF> First ( & Source Module >) = { DRIVERDEF }

```
Lmodule > JDGF MODULE ID ENDDEF TAKES INPUT SQBO Linguit - plast >

GABC SEMICOL Lord > cmoduleDef 7

First < module 7) = SDEF &
```

\( \text{input-plot} \) \( \text{> 1D (0LON < dela Type > < new-input-plot) } \)
 \[
 \]
 \( \text{First (< input-plot)} = \( \text{\$\left( \text{plot} \) } \) \( \text{\$\text{\$\left( \text{\$\tex{\$\text{\$\text{\$\text{\$\text{\$\text{\$\text{\$\text{\$\tex{\$\text{\$

9.

(new-input-plat > > (OMMA 10 COLOMON & data Typo (new-output-plat > 1 C

First Canew-input-plat >) = { COMMA, E}

Follow (< new-input-plat >) = { SQBC }

Lought-plot >-1D (010N < datatype > < new\_support plot > 1 C

II.
< new-output-plat > > COMMA 10 COCON & datatype > < new-output-plat > 1 &

Fort & mew-output-plat > ) = & COMMA, & &

Follow (< mew-output-plat > ) = & SQBC &

First (2 odata Type >) \( \text{0} = \xi \) INTEGER I REAL | BOOLEAN | ARRAY \( \xi \text{0} \) OF (\text{type}) \( \text{0} = \xi \) INTEGER GER, REAL, BOOLEAN, ARRAY ?

```
upe > INTE OGER IREAL /BOOLEAN
                         First (< type >) = {INTEGER, REAL, BOOLEAN }
     < module Deef > -> START < stalements > END
                                 First (< module Dief >) = { .START }
    < statements >>> 2 -statement 7 < statements >16
                                   First ((slotemente)) = & GET_VolALVE, PRINT, ID, SQBO, VSG, DECLARE, STOPE FOR
                                                                         WHILE, SWITCH &, E }
                                  FOLLOW ( < statements > ) = {BREAK, ENDY
  < slatement > -> < costent > 1 < semple 5 timb > 1 < bdeclare 5 limt > 1 < conditional 5 timb > 1 < docater 5 timb > 1 < conditional 5 timb > 1 < docater 5 
                Forst (calaborent ) = & GET. Vai ALUE, PRINT, 10, SQ80, VSE, DECLARE, FOR, WHILE
 (io Start) -> GET_VALVE BO ID BC SEMICOL | PRINT BO (var And Bood) > PC SEMICOL
    (var 7 > 10 WHICH < which I Pd > | NUM | R NUM
                          Fost ( van ) = { 10, NUM, RNUM }
< bool 7 -> TRUE | FALSE
```

Forst (O( bool)) = & TRUE, FALSE }

- 1 de landon

```
< var aAnd Baol > tre > Evan > / clool >
           First ( Var And Bool ) = & ID, NUM, RNUM, TRUE, FALSE
& < which 100 -> SQNO 10 SQBC 1E
         First (< which 10d) >) = $5080, 6 }
         Follow (< which Id>) = & BC }
( - 2 Simple Strut > > Cossignment Strut > 1 < module Reuse Strut >
          First (c.simple Strat >) = { 10, 5000, USE }
   < assignment Start >> 10 < which 5 tant >>

First (< assignment 5 tant d>) = $10 }
   Lwhich Slmt > = D < 1 value 10 slmt > 1 < 1 value ARRS tmt >
               First (< which start >) = 5 ASSIGNOP, SQBOZ
< 1 value 1 d Start > => ASSIGN OP L enpression > S.EMICOL
             First (< 1 value 1 d slabment > = { ASSIGNOP}
    2/ ratue ARR Start > D -> SQBO Linder > GQBC ASSIGNOP L'enprission > SEMICOL
              Forst (<1 value ARR stmt >) = { SaBC }
   <index > -) NUM /ID
            Forst Cinden > - & NUM, 10 }
```

```
· Calandon
      mobile Rouse Start > > Cophonal > USE MODULE 10 WITH PARAMETERS
        Ou First (< module Reuse Stant >) ={ SABO, USE }
 < oplional > -> 5080 cid Lust > 5080 ASSIGNOPLE
           First (<optional>) = { SQBO, E }
           Follow (< optional>) = & USE }
Lid List > > 10 ( rid List's >
          First (led be List > = 210 }
 (id List') -> COMMA 10 (id List') 16
            Follow ((id List')) = S COMMA, E &
Follow ((id List'))= S SEMICOL, SQBC, COLON&
( enphysion > -> PLUS < enpression'> (MINUS < enpression') | W bool Conjunct Aruthmoly >
   First (<enpression >) - SPLUS, MINUS, BO, BID, RNUM, NUM }
 33.

<enpression'> -> < ran > 80 < authortExpr > 80
        First (<expression'>) = { PO, 1D, NVM, RNVM}
L faction ' & -> ( book > ) = 286, 10, Nom, RNOWS
2 bool Conjund Arellmete 7 -> to Cfactors' > C bool Conjunct Arellmeter'>
    first (< book organist Archimete >) = { 80,10, RNUM, NUM }
```

```
2 lool Conjind Arithmete 1 > > < logical OpTo (factor 1 > < bool Conjunt trulmete;)
                 Ford ( Lhool Conjund Aruthmetic '7) = { E, AND, OR }
Follow ( chool Conjunct Aruthmetic' > 7 = { SEMICOL

Clastion '7 -XA ritmeter Enger > Lactor "> Kbool > Lfactor "> | Lbool >
         First (<factor' >) = { 80,10, NUM, RNUM }
 < factor"> > < & relational Op > (arithmetic Earps > < factor") 1 €
         First 6(< factor">)= & LT, LE, GT, GE, EQ, NE, E}
          Follow (<factor") = SSEMICOLZ
2 Kovarelhynetic Expr > > BO < bool Conjund + relhmete > BC | < tom | > (arelhmete Expr >) = \( 80, 10, Nvm, RNvm \)
Avullmente Expr'> > <op1> < born 1> (arullmelie Expr'7) &

Forst (<arullmente Expr'>) = & PLUS, MINUS, & &

Follow (carullmente Expr'>) = & tomand LT, LE, GT, GE, EQ, NE, AND

OR, SEMICOLZ, BC&
             forst ( torm 1 > ) = { 80,10, Num, RNUM }
(term1') -> < op1'> < op2 > < lorm1'>16
           first (cterm1'>) = & MUL, DIV, E &
          follow (< term 1/7 = & PLUS, MINUS, COMMA, LT, LE, GT, GE, EQ, NE, AND
OR, SEMICOL, BC
```

```
opl > > PLUS IMINUS
        First (< op1>) = & PLUS, MINUS &
 ( op 21/5 -> MULIDIN
          First (< 0,211) ) = { MUL, DIV }
  Firef (< 000 0p27)= { BO, 10, NOVIM, RNVM }
  < logical Op > -> AND OR
           First ( logued Op7) -> & AND, OR
 < Brelinal Op 7 -> LT/LE 167/6E | EQ[NE
          First (< relational Op7) = {LT, LE, GT, GE, EQ, NEG
  47.

< D declare Stant > -> DECLARE < idList > COLON < dala Type > SEMICOL
          first (< declare Strut >) = & DECLARE }
  conditional Start > > SWITCH BO ID BC START < CaseStart > ( default > END
           first (< conditional Start >) = & SWITCH &
  149.

L Case Sturt > -> local CASE < value > COLOMON < slabour > BREAK SEMICOL
         First (case Strut >) = Go & CASE }
< (cose Strut' > -> < case Strut > / E
         Forst ( case Strut 1 >) = SCASE, E}
```

Follow (< case Stimt' >) = { DDE FAULT, ENU}

CE 51, <value > -> NUMITRUE | FALSE First Ce value > ) = & NUM, TRUE, FALSE &

< default > > DEFAULT COLORN < SIdements > BREAK SEVICOL/E first (< default >) = & DEFAULT, EZ follow ( < expanlt > ) = & ENDb

( storature Start ) -> FOR DO ID WIN NUM RANGEOF NUM DC START Colomony END 1 WHILE GO < book Conjunt Arellmete > BC START < slabouret > OFNO

first (charaline Strat > ) = SFOR, WHILE }

654. L range 7-> 10 RANGEORP (ronge') NUM RANGEOP ( range') forst (< range ] = \$ \$ \$ 10, NUM }

55. ( rungl '7 -210/ NUM first ( < grange 7) = {10, NVM}

- 1. Precedence order considered for expressions Arithmetic Thelabord to be logical. We have accordingly introduced new non-terminales to incorporate this precedence
- I. The points that have been laken inter cond consideration for modifying the grammar are as follows -

a. Left Recursion

b. Left Factoring

c. Precedence

d. First Sets

e. Follow gets

f. LL 1 compatibility

3. Since a boolean expression can include an withmetic expression, we have introduced a new non-terminal boolean punct arithmetic to resolve left factoring in the following rule

( expression > -> ( withmetrit ups > 10 ( boolean Expr >

- 4. The wulhmelie operators \* and / I have I more precedence of over '+' and '- 'have been incorporated using op! and op!' as the new non-terminals
  - 5. In the original grammar, case stolement nont terminal did not have an inding condition. We have introduced a new non-terminal casestament casestalement 'so that infinite no- of case stalements are not generated along with preserving the conclution that alleast one case stalement must exist.

- 6. Since a range statement can derive expressions with edentifier, as well we have accordingly modified the rules for range non-branead and remove left factoring
- I All the rules that have been menhored in the mails as well as language specifications report have been thoroughly sauchest and incorporated