**22100409 Siaw Jia Yuin**

**LL1 Parser**

**BNF**

S -> BEGIN\_BLOCK BEGIN\_STMT

BEGIN\_BLOCK -> program PROGRAM\_NAME

PROGRAM\_NAME -> id

BEGIN\_STMT -> begin STMT\* end

STMT -> DECLARATION

| IF\_STMT

| PRINT\_STMT

| WHILE\_STMT

| FOR\_STMT

| BREAK\_STMT

DECLARATION -> int OPERATOR ;

| OPERATOR ,

OPERATOR -> id = OP\_STMT

| OP\_STMT

OP\_STMT -> TERM

| OP\_STMT + TERM

| id ++

TERM -> TERM \* FACTOR

| FACTOR

FACTOR -> number\_literal

| id

PRINT\_CONTENT -> string\_literal

| id

IF\_STMT -> if ( COM\_STMT ) BEGIN\_STMT

| if ( COM\_STMT ) BEGIN\_STMT else\_if ( OP\_STMT ) BEGIN\_STMT

| if ( COM\_STMT ) BEGIN\_STMT else\_if ( OP\_STMT ) BEGIN\_STMT else BEGIN\_STMT

PRINT\_STMT -> print\_line ( PRINT\_CONTENT ) ;

| display ( PRINT\_CONTENT ) ;

WHILE\_STMT -> while ( STMT ) BEGIN\_STMT

FOR\_STMT -> for ( STMT ) BEGIN\_STMT

BREAK\_STMT -> break ;

COM\_STMT -> LT\_STMT

| EQ\_STMT

LT\_STMT -> FACTOR < FACTOR

EQ\_STMT -> FACTOR == FACTOR

**BNF without Left Recursion**

S -> BEGIN\_BLOCK BEGIN\_STMT

BEGIN\_BLOCK -> program id

BEGIN\_STMT -> begin STMTS end

STMTS -> STMT STMTS

STMTS -> ''

STMT -> DECLARATION\_LIST

STMT -> IF\_STMT

STMT -> PRINT\_STMT

STMT -> WHILE\_STMT

STMT -> FOR\_STMT

STMT -> BREAK\_STMT

DECLARATION\_LIST -> DECLARATION DECLARATION\_LIST'

DECLARATION\_LIST -> ‘’

DECLARATION\_LIST' -> , DECLARATION DECLARATION\_LIST’

DECLARATION\_LIST' -> ''

DECLARATION -> TYPE ID\_LIST ;

DECLARATION -> ID\_LIST ;

TYPE -> int

TYPE -> integer

ASSIGNMENT -> = OP\_STMT

ASSIGNMENT -> ‘’

ID\_LIST -> id ASSIGNMENT ID\_LIST'

ID\_LIST' -> , id ASSIGNMENT ID\_LIST'

ID\_LIST' -> ''

OP\_STMT -> TERM OP\_STMT'

OP\_STMT' -> + TERM OP\_STMT'

OP\_STMT’ -> ++

OP\_STMT' -> ''

TERM -> FACTOR TERM'

TERM' -> \* FACTOR TERM'

TERM' -> ''

FACTOR -> number\_literal

FACTOR -> id

PRINT\_CONTENT -> string\_literal

PRINT\_CONTENT -> identifier

IF\_STMT -> if ( COM\_STMT ) BEGIN\_STMT IF\_STMT'

IF\_STMT' -> else\_if ( COM\_STMT ) BEGIN\_STMT IF\_STMT'

IF\_STMT' -> else BEGIN\_STMT

IF\_STMT' -> ''

PRINT\_STMT -> print\_line ( PRINT\_CONTENT ) ;

PRINT\_STMT -> display ( PRINT\_CONTENT ) ;

WHILE\_STMT -> while ( COM\_STMT ) BEGIN\_STMT

FOR\_STMT -> for ( DECLARATION COM\_STMT ; OP\_STMT ) BEGIN\_STMT

BREAK\_STMT -> break ;

COM\_STMT -> TERM COM\_STMT’

COM\_STMT’ -> < FACTOR

COM\_STMT’ -> == FACTOR

COM\_STMT’ -> ‘’

**First Set**

S = {program}

BEGIN\_BLOCK = {program}

BEGIN\_STMT = {begin}

STMTS = { /eps, if, print\_line, display, while, for, break, int, integer, id }

STMT = { /eps, if, print\_line, display, while, for, break, int, integer, id }

DECLARATION\_LIST = { /eps, int, integer, id }

DECLARATION\_LIST’ = { /eps , , }

DECLARATION = { int, integer, id }

TYPE = { int, integer }

ASSIGNMENT = { =, /eps }

ID\_LIST = {id}

ID\_LIST’ = { , , /eps }

OP\_STMT = { number\_literal, id }

OP\_STMT’ = { +, ++, /eps }

TERM = { number\_literal, id }

TERM’ = { \*, /eps }

FACTOR = { number\_literal, id }

PRINT\_CONTENT = { string\_literal, id }

IF\_STMT = { if }

IF\_STMT’ = { else\_if, else, /eps }

PRINT\_STMT = { print\_line, display }

WHILE\_STMT = { while }

FOR\_STMT = { for }

BREAK\_STMT = { break }

COM\_STMT = { number\_literal, id }

COM\_STMT’ = { <, ==, /eps }

**Follow Set**

S = { $ }

BEGIN\_BLOCK = { begin }

BEGIN\_STMT = { $, else\_if, else, end, if, print\_line, display, while, for, break, int, integer, id }

STMTS = { end }

STMT = { end, if, print\_line, display, while, for, break, int, integer, id }

DECLARATION\_LIST = { end, if, print\_line, display, while, for, break, int, integer, id }

DECLARATION\_LIST’ = { end, if, print\_line, display, while, for, break, int, integer, id }

DECLARATION = { end, if, print\_line, display, while, for, break, int, integer, id, , ,DECLARATION\_LIST’, number\_Literal }

TYPE = { id }

ASSIGNMENT = { , , ; }

ID\_LIST = {;}

ID\_LIST’ = { ; }

OP\_STMT = { ), , , ; }

OP\_STMT’ = { ), , , ; }

TERM = { +, ++, <, ==, ), ;, , }

TERM’ = { +, ++, <, ==, ), ;, , }

FACTOR = { \*, +, ++, ), ;, <, ==, , }

PRINT\_CONTENT = { ) }

IF\_STMT = { end, if, print\_line, display, while, for, break, int, integer, id }

IF\_STMT’ = { end, if, print\_line, display, while, for, break, int, integer, id }

PRINT\_STMT = { end, if, print\_line, display, while, for, break, int, integer, id }

WHILE\_STMT = { end, if, print\_line, display, while, for, break, int, integer, id }

FOR\_STMT = { end, if, print\_line, display, while, for, break, int, integer, id }

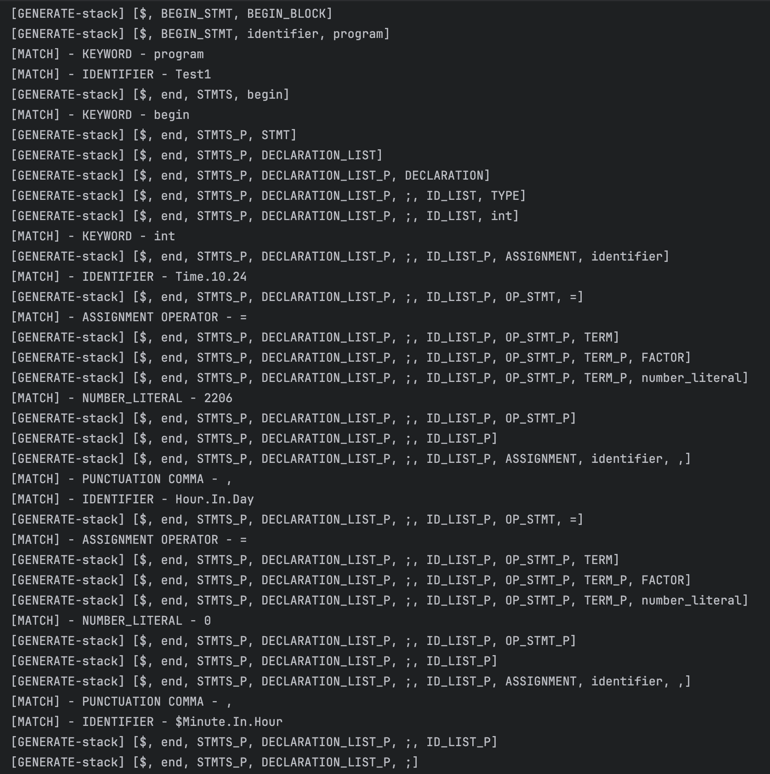
BREAK\_STMT = { end, if, print\_line, display, while, for, break, int, integer, id }

COM\_STMT = { ), ; }

COM\_STMT’ = { ), ; }

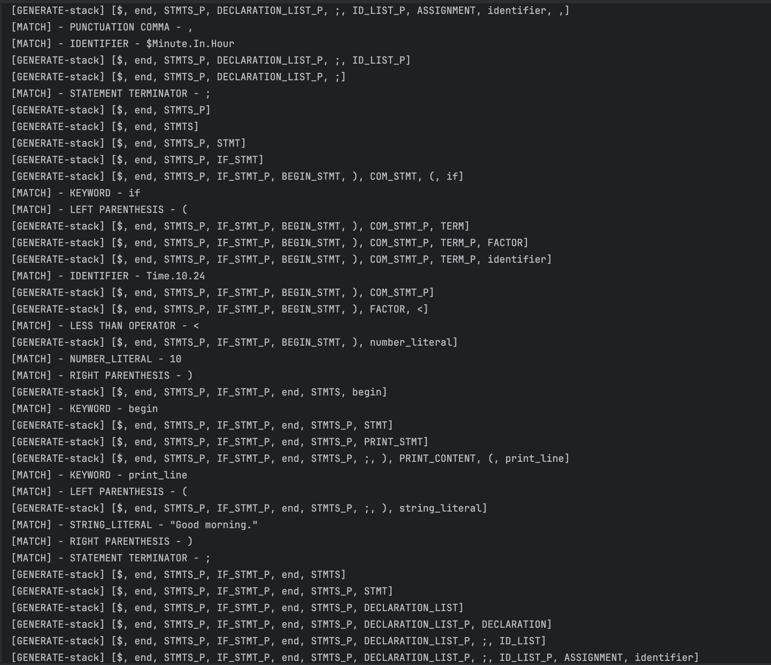
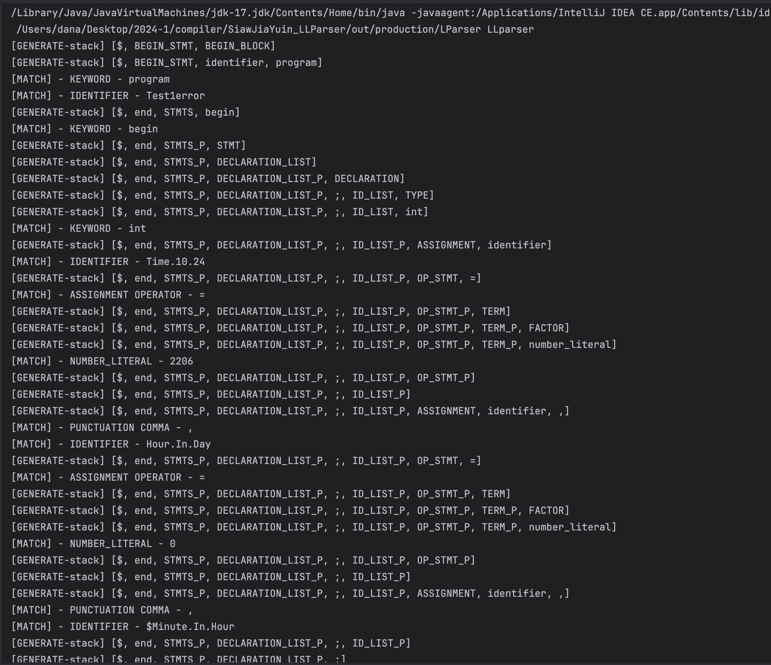
**Parsing Table** 

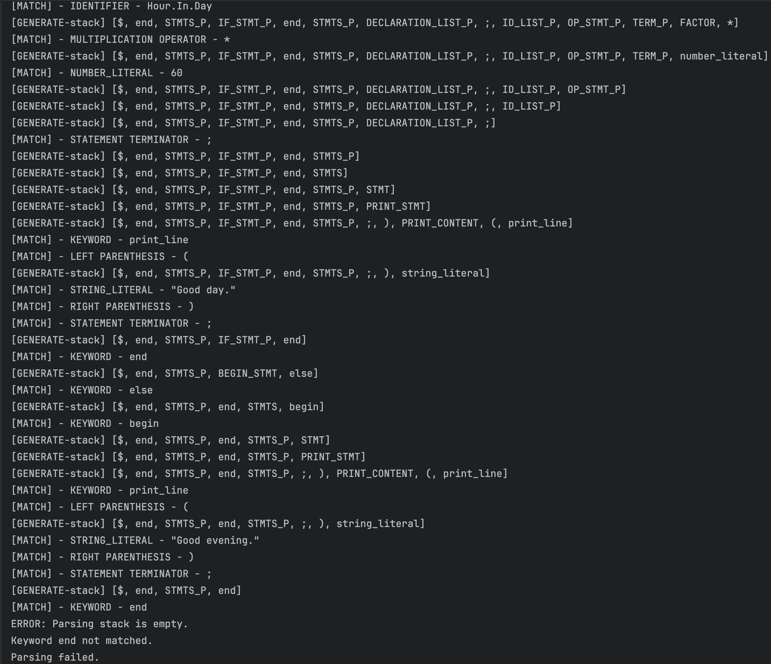
**Program Output**

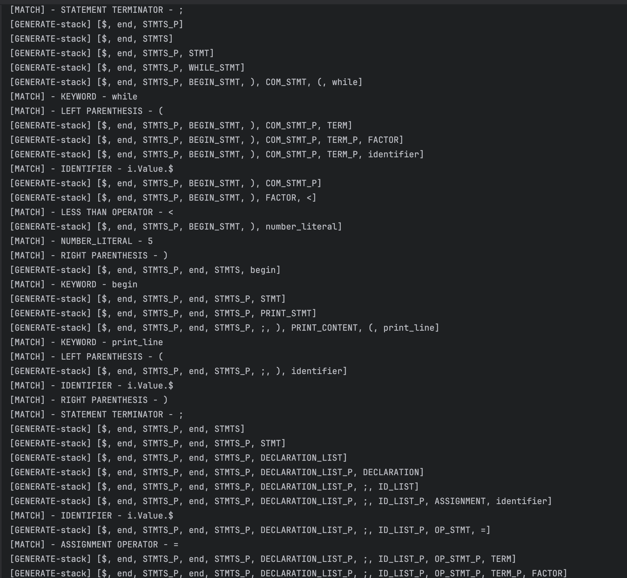
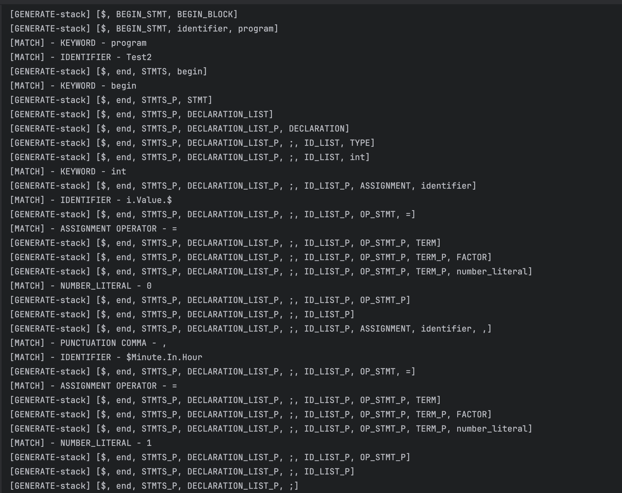
1. Test1A screenshot of a computer program

   Description automatically generatedA screen shot of a computer

   Description automatically generatedA screen shot of a computer

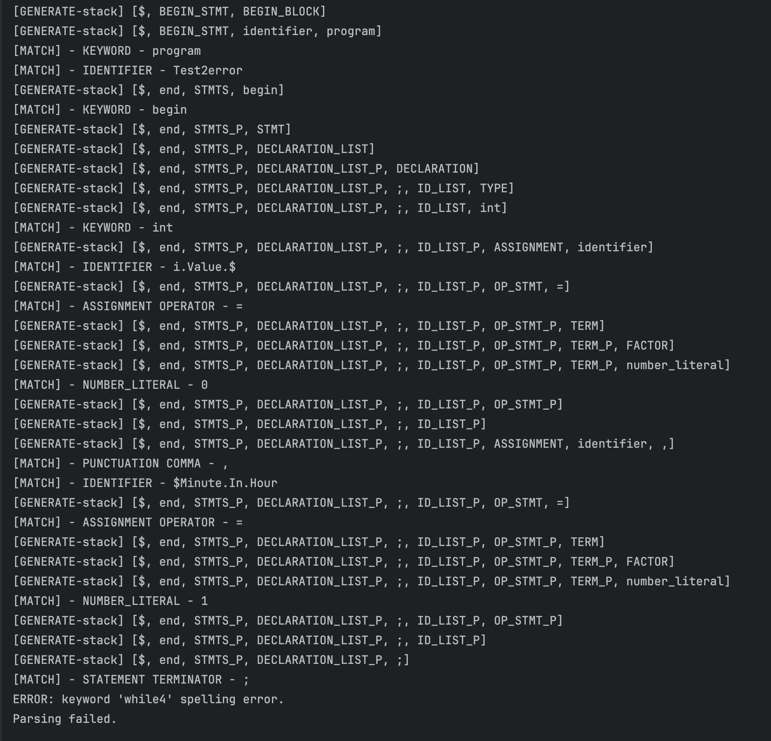
   Description automatically generated
2. Test1errorA computer screen shot of white text

   Description automatically generated
3. Test2

****A screen shot of a computer program

Description automatically generated

1. Test2error

****

1. Test3

**A screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screenshot of a computer program

Description automatically generatedA screen shot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated**

1. Test3error

**A screenshot of a computer program

Description automatically generatedA screen shot of a computer

Description automatically generatedA screenshot of a computer program

Description automatically generated**