COMPILER DESIGN PRINCIPLES
THE LEXICAL ANALYZER

Dr. Mohammad Reza Razzazi
Amirkabir University of Technology
SPRING 2017

ALI GHOLAMI - 9531504
AYDA DOOST MOHAMMADI - 9431550

Token and Keyword Detection Code (.flex)

```
IMPORT JAVA.IO.FILENOTFOUNDEXCEPTION;
IMPORT JAVA.IO.FILEREADER;
IMPORT JAVA.IO.IOEXCEPTION;
IMPORT JAVA.LANG.*;

CLASS CLASS_MAIN
{
        INT SYMBOL_TABLE_SIZE = 100;
        PUBLIC STRING [] SYMBOL_TABLE = NEW STRING[SYMBOL_TABLE_SIZE];
        INT ENTRY_POSITION = 0;
        BOOLEAN EXISTS = FALSE;

        PUBLIC STATIC VOID MAIN(STRING ARGS[])
        {
                        FILEREADER FR = NULL;
              STRING INPUT = "GLOBALTEST2.SHL";
              TRY {
                 FR = NEW FILEREADER(INPUT);
              } CATCH (FILENOTFOUNDEXCEPTION E) {
                 E.PRINTSTACKTRACE();
              }
              SYSTEM.OUT.PRINTLN("LEXEME\TTOKEN\TATTRIBUTE");
              YYLEX YYLEX = NEW YYLEX(FR);
              TRY {
                 YYLEX.YYLEX();
              } CATCH (IOEXCEPTION E) {
                 E.PRINTSTACKTRACE();
              }
        }
```

```
}
%%
/* DECLARATION SECTION */
/* IDENTIFIER */
%BYACCJ
/* TOKENS */
LETTER = [A-ZA-Z]
NONZERO_DIGIT = [1-9]
DIGIT = "0"|{NONZERO_DIGIT}

/* MAIN & PROGRAM KEYWORDS */
PROGRAM_KW = (PROGRAM)
MAIN_KW = (MAIN)

/* VARIABLE TYPES KEYWORDS */
INTEGER_KW = (INT)
REAL_KW = (REAL)
CHARACTER_KW = (CHAR)
BOOLEAN_KW = (BOOL)
PROCEDURE_KW = (PROCEDURE)

/* CONDITION KEYWORDS */
IF_KW = (IF)
THEN_KW = (THEN)
ELSE_KW = (ELSE)
DO_KW = (DO)
WHILE_KW = (WHILE)
SWITCH_KW = (SWITCH)
CASE_KW = (CASE)
DEFAULT_KW = (DEFAULT)
END_KW = (END)
```

```
WHEN_KW = (WHEN)

/* UTILITY KEYWORDS */
FOR_KW = (FOR)
RETURN_KW = (RETURN)
EXIT_KW = (EXIT)
UPTO_KW = (UPTO)
DOWNTO_KW = (DOWNTO)

/* LOGICAL OPERATIONS KEYWORDS */
AND_KW = (AND)
OR_KW = (OR)
NOT_KW = (NOT)

/* SEPARATOR KEYWORDS */
SEMICOLON_KW = [;]
COLON_KW = [:]
ASSIGN_KW = (:=)
OPENBRACKET_KW = "["
CLOSEBRACKET_KW = "]"
OPENACCOLADE_KW = [{]
CLOSEACCOLADE_KW = [}]
OPENPARENTHESIS_KW = [(]
CLOSEPARENTHESIS_KW = [)]
COMMA_KW = [,]

/* RELATIONAL OPERATORS */
LT_KW = [<]
GT_KW = [>]
EQ_KW = [=]
NEQ_KW = (<>)
```

```
GTE_KW = (>=)
LTE_KW = (<=)

/* OTHER KEYWORDS */
DOT_KW = "\."
SINGLE_QUOTE_KW = "\U0027"

/* ARITHMETIC OPERATORS */
ADD_KW = [+]
SUB_KW = [-]
MUL_KW = [*]
DIV_KW = [/]
MOD_KW = [%]
SHARP_KW = [#]

BOOLCONST = (TRUE)|(FALSE)
CHARCONST = {SINGLE_QUOTE_KW} ({LETTER} | {DIGIT}) {SINGLE_QUOTE_KW}
REALCONST =
{SHARP_KW}((({DIGIT})|({NONZERO_DIGIT}({DIGIT})*))({DOT_KW})({DIGIT})*{NONZERO_DIGIT})
NUMCONST = {SHARP_KW}({DIGIT}|{NONZERO_DIGIT}{DIGIT}*)
ID = {LETTER}+


%%
/* RULES SECTION */
{PROGRAM_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "PROGRAM_KW\T" + '-');
}
{MAIN_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "MAIN_KW\T" + '-');
}
```

```
{PROCEDURE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "PROCEDURE_KW\T" + '-');
}
{INTEGER_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "INTEGER_KW\T" + '-');
}
{REAL_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "REAL_KW\T" + '-');
}
{CHARACTER_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "CHARACTER_KW\T" + '-');
}
{BOOLEAN_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "BOOLEAN_KW\T" + '-');
}
{IF_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "IF_KW\T" + '-');
}
{THEN_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "THEN_KW\T" + '-');
}
{ELSE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "ELSE_KW\T" + '-');
}
{DO_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "DO_KW\T" + '-');
}
{WHILE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "WHILE_KW\T" + '-');
}
{FOR_KW} {
```

```
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "FOR_KW\T" + '-');
}
{SWITCH_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "SWITCH_KW\T" + '-');
}
{CASE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "CASE_KW\T" + '-');
}
{DEFAULT_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "DEFAULT_KW\T" + '-');
}
{END_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "END_KW\T" + '-');
}
{RETURN_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "RETURN_KW\T" + '-');
}
{EXIT_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "EXIT_KW\T" + '-');
}
{WHEN_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "WHEN_KW\T" + '-');
}
{AND_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "AND_KW\T" + '-');
}
{OR_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "OR_KW\T" + '-');
}
{NOT_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "NOT_KW\T" + '-');
```

```
}

{SEMICOLON_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "SEMICOLON_KW\T" + '-');
}
{COLON_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "COLON_KW\T" + '-');
}
{COMMA_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "COMMA_KW\T" + '-');
}
{SINGLE_QUOTE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "SINGLE_QUOTE_KW\T" + '-');
}
{UPTO_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "UPTO_KW\T" + '-');
}
{DOWNTO_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "DOWNTO_KW\T" + '-');
}
{ASSIGN_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "ASSIGN_KW\T" + '-');
}
{OPENPARENTHESIS_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "OPENPARENTHESIS_KW\T" + '-');
}
{CLOSEPARENTHESIS_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "CLOSEPARENTHESIS_KW\T" + '-');
}
{OPENBRACKET_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "OPENBRACKET_KW\T" + '-');
```

```
}
{CLOSEBRACKET_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "CLOSEBRACKET_KW\T" + '-');
}

{OPENACCOLADE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "OPENACCOLADE_KW\T" + '-');
}
{CLOSEACCOLADE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "CLOSEACCOLADE_KW\T" + '-');
}
{DOT_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "DOT_KW\T" + '-');
}
{LT_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "LT_KW\T" + '-');
}
{GT_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "GT_KW\T" + '-');
}
{EQ_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "EQ_KW\T" + '-');
}
{NEQ_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "NEQ_KW\T" + '-');
}
{GTE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "GTE_KW\T" + '-');
}
{LTE_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "LTE_KW\T" + '-');
```

```
}
{ADD_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "ADD_KW\T" + '-');
}
{SUB_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "SUB_KW\T" + '-');
}
{MUL_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "MUL_KW\T" + '-');
}
{DIV_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "DIV_KW\T" + '-');
}
{MOD_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "MOD_KW\T" + '-');
}
{SHARP_KW} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "SHARP_KW\T" + '-');
}
{BOOLCONST} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "BOOLCONST\T" + '-');
}
{CHARCONST} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "CHARCONST\T" + '-');
}
{REALCONST} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "REALCONST\T" + '-');
}
{NUMCONST} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "NUMCONST\T" + '-');
}
```

```
{ID} {
        SYSTEM.OUT.PRINTLN(YYTEXT() + "\T" + "ID\T" + "SYMBOL TABLE ENTRY");
}
"\S"|"\N"|"\R"|"\T" {
}
. {
}
```

Input Program (Validation Test)

```
program globaltest
        int a;
        int b:=#3;
        real f;
        real k:=#0.4;
        real l:=#3.5000;
        char c:='w';
        char h;
        bool s;
        bool g:=true;
        int array[#2]:={#1,#7};
        char chars [0..2]:={'c','d','7'};
        procedure p ( int w , char t){
                switch w
                        case #10 :{
                                w:=+(w,w);
                                w:=-(+(*(w,w),w),w);

                        }
                        case #20:{
                                t:= and (t,t);
                                w:= or(w,w);
                                do
                                        for i:=#1 upto #10 do
                                                w:=%(w,#32);
                                while <>(w,#1);
                        }
```

```
                case #30:{
                        if not =(w,#4) then w:=5; else w:=#9;
                }
        default :{
                if and (t,t) and then or(t,#0)
                        then w:=-w;
        }
        end;
};
main{
                array [#2] := array [#2] - (char [#2] = 'd');
                k := and (or else (+(*(#2,k),false),true), array[#1]),#1);
                return -(l,k);
                exit when not(>(array[#2],#0));
}
```

Yylex Output – Tokens and Keywords

Lexeme	Token	Attribute
program	PROGRAM_KW	-
globaltest	ID	Symbol Table Entry
int	INTEGER_KW	-
a	ID	Symbol Table Entry
;	SEMICOLON_KW	-
int	INTEGER_KW	-
b	ID	Symbol Table Entry
:=	ASSIGN_KW	-
#3	NUMCONST	-
;	SEMICOLON_KW	-
real	REAL_KW	-
f	ID	Symbol Table Entry
;	SEMICOLON_KW	-
real	REAL_KW	-
k	ID	Symbol Table Entry
:=	ASSIGN_KW	-
#0.4	REALCONST	-
;	SEMICOLON_KW	-
real	REAL_KW	-
l	ID	Symbol Table Entry
:=	ASSIGN_KW	-
#3.5	REALCONST	-
;	SEMICOLON_KW	-
char	CHARACTER_KW	-
c	ID	Symbol Table Entry
:=	ASSIGN_KW	-
'w'	CHARCONST	-
;	SEMICOLON_KW	-

char	CHARACTER_KW	-
h	ID	Symbol Table Entry
;	SEMICOLON_KW	-
bool	BOOLEAN_KW	-
s	ID	Symbol Table Entry
;	SEMICOLON_KW	-
bool	BOOLEAN_KW	-
g	ID	Symbol Table Entry
:=	ASSIGN_KW	-
TRUE	BOOLCONST	-
;	SEMICOLON_KW	-
int	INTEGER_KW	-
array	ID	Symbol Table Entry
[	OPENBRACKET_KW	-
#2	NUMCONST	-
]	CLOSEBRACKET_KW	-
:=	ASSIGN_KW	-
{	OPENACCOLADE_KW	-
#1	NUMCONST	-
,	COMMA_KW	-
#7	NUMCONST	-
}	CLOSEACCOLADE_KW	-
;	SEMICOLON_KW	-
char	CHARACTER_KW	-
chars	ID	Symbol Table Entry
[	OPENBRACKET_KW	-
.	DOT_KW	-
.	DOT_KW	-
]	CLOSEBRACKET_KW	-
:=	ASSIGN_KW	-

{	OPENACCOLADE_KW	-
'c'	CHARCONST	-
,	COMMA_KW	-
'd'	CHARCONST	-
,	COMMA_KW	-
'7'	CHARCONST	-
}	CLOSEACCOLADE_KW	-
;	SEMICOLON_KW	-
procedure	PROCEDURE_KW	-
p	ID	Symbol Table Entry
(	OPENPARENTHESIS_KW	-
int	INTEGER_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-
char	CHARACTER_KW	-
t	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
{	OPENACCOLADE_KW	-
switch	SWITCH_KW	-
w	ID	Symbol Table Entry
case	CASE_KW	-
#10	NUMCONST	-
:	COLON_KW	-
{	OPENACCOLADE_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-
+	ADD_KW	-
(	OPENPARENTHESIS_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-

w	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-
-	SUB_KW	-
(	OPENPARENTHESIS_KW	-
+	ADD_KW	-
(	OPENPARENTHESIS_KW	-
*	MUL_KW	-
(	OPENPARENTHESIS_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-
w	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
,	COMMA_KW	-
w	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
,	COMMA_KW	-
w	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
}	CLOSEACCOLADE_KW	-
case	CASE_KW	-
#20	NUMCONST	-
:	COLON_KW	-
{	OPENACCOLADE_KW	-
t	ID	Symbol Table Entry
:=	ASSIGN_KW	-
and	AND_KW	-

(	OPENPARENTHESIS_KW	-
t	ID	Symbol Table Entry
,	COMMA_KW	-
t	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-
or	OR_KW	-
(	OPENPARENTHESIS_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-
w	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
do	DO_KW	-
for	FOR_KW	-
i	ID	Symbol Table Entry
:=	ASSIGN_KW	-
#1	NUMCONST	-
upto	UPTO_KW	-
#10	NUMCONST	-
do	DO_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-
%	MOD_KW	-
(	OPENPARENTHESIS_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-
#32	NUMCONST	-

)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
while	WHILE_KW	-
<>	NEQ_KW	-
(	OPENPARENTHESIS_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-
#1	NUMCONST	-
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
}	CLOSEACCOLADE_KW	-
case	CASE_KW	-
#30	NUMCONST	-
:	COLON_KW	-
{	OPENACCOLADE_KW	-
if	IF_KW	-
not	NOT_KW	-
=	EQ_KW	-
(	OPENPARENTHESIS_KW	-
w	ID	Symbol Table Entry
,	COMMA_KW	-
#4	NUMCONST	-
)	CLOSEPARENTHESIS_KW	-
then	THEN_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-
;	SEMICOLON_KW	-
else	ELSE_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-

#9	NUMCONST	-
;	SEMICOLON_KW	-
}	CLOSEACCOLADE_KW	-
default	DEFAULT_KW	-
:	COLON_KW	-
{	OPENACCOLADE_KW	-
if	IF_KW	-
and	AND_KW	-
(	OPENPARENTHESIS_KW	-
t	ID	Symbol Table Entry
,	COMMA_KW	-
t	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
and	AND_KW	-
then	THEN_KW	-
or	OR_KW	-
(	OPENPARENTHESIS_KW	-
t	ID	Symbol Table Entry
,	COMMA_KW	-
#0	NUMCONST	-
)	CLOSEPARENTHESIS_KW	-
then	THEN_KW	-
w	ID	Symbol Table Entry
:=	ASSIGN_KW	-
-	SUB_KW	-
w	ID	Symbol Table Entry
;	SEMICOLON_KW	-
}	CLOSEACCOLADE_KW	-
end	END_KW	-
;	SEMICOLON_KW	-

}	CLOSEACCOLADE_KW	-
;	SEMICOLON_KW	-
main	MAIN_KW	-
{	OPENACCOLADE_KW	-
array	ID	Symbol Table Entry
[	OPENBRACKET_KW	-
#2	NUMCONST	-
]	CLOSEBRACKET_KW	-
:=	ASSIGN_KW	-
array	ID	Symbol Table Entry
[	OPENBRACKET_KW	-
#2	NUMCONST	-
]	CLOSEBRACKET_KW	-
-	SUB_KW	-
(	OPENPARENTHESIS_KW	-
char	CHARACTER_KW	-
[	OPENBRACKET_KW	-
#2	NUMCONST	-
]	CLOSEBRACKET_KW	-
=	EQ_KW	-
'd'	CHARCONST	-
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
k	ID	Symbol Table Entry
:=	ASSIGN_KW	-
and	AND_KW	-
(	OPENPARENTHESIS_KW	-
or	OR_KW	-
else	ELSE_KW	-
(	OPENPARENTHESIS_KW	-

+	ADD_KW	-
(	OPENPARENTHESIS_KW	-
*	MUL_KW	-
(	OPENPARENTHESIS_KW	-
#2	NUMCONST	-
,	COMMA_KW	-
k	ID	Symbol Table Entry
)	CLOSEPARENTHESIS_KW	-
,	COMMA_KW	-
FALSE	BOOLCONST	-
)	CLOSEPARENTHESIS_KW	-
,	COMMA_KW	-
TRUE	BOOLCONST	-
)	CLOSEPARENTHESIS_KW	-
,	COMMA_KW	-
array	ID	Symbol Table Entry
[	OPENBRACKET_KW	-
#1	NUMCONST	-
]	CLOSEBRACKET_KW	-
)	CLOSEPARENTHESIS_KW	-
,	COMMA_KW	-
#1	NUMCONST	-
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
return	RETURN_KW	-
-	SUB_KW	-
(	OPENPARENTHESIS_KW	-
l	ID	Symbol Table Entry
,	COMMA_KW	-
k	ID	Symbol Table Entry

)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
exit	EXIT_KW	-
when	WHEN_KW	-
not	NOT_KW	-
(	OPENPARENTHESIS_KW	-
>	GT_KW	-
(	OPENPARENTHESIS_KW	-
array	ID	Symbol Table Entry
[	OPENBRACKET_KW	-
#2	NUMCONST	-
]	CLOSEBRACKET_KW	-
,	COMMA_KW	-
#0	NUMCONST	-
)	CLOSEPARENTHESIS_KW	-
)	CLOSEPARENTHESIS_KW	-
;	SEMICOLON_KW	-
}	CLOSEACCOLADE_KW	-