

Compiler Design Principles The Lexical Analyzer

Dr. Mohammad Reza Razzazi

*Amirkabir University of Technology*

*SPRING 2017*



aLI gHOLAMI - 9531504

aYDA DOOST MOHAMMADI - 94315504

**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 | - |