

19CSE401 - Compiler Design

Cup Parser

Lab sheet-6

Done By:

N Sai Pavan Krishna
AM.EN.U4CSE19347

EXAMPLE 1: A grammar for simple declaration statement. Carefully go through the rules and make sure you understand the significance of every line in the grammar. Follow the step-by-step instructions below to create the grammar, compile and execute:

Exercises:

1. Extend the same grammar to accept the declarations of the form:

- i. int a,b,c;
 - ii. float a,b,sum;
 - iii. char ch,ch1;

The screenshot shows a VS Code interface with two terminals and a code editor.

TERMINAL 1:

```
(base) nsapk@Crisp lab6 % java Main
LA int
LA
LA a
LA ,
LA b
LA ,
LA c
LA ;
LA \n
LA float
LA ,
LA a
LA ,
LA b
LA ,
LA sum
LA ,
LA \n
LA char
LA ,
LA ch
LA ,
LA ch1
LA ;
LA \n
LA \n
LAEOF
```

TERMINAL 2:

```
Valid declaration
```

CODE EDITOR:

File: Main.java 2

```
1 import java_cup.runtime.*;
2 scan with { return getScanner().next_token(); };
3 terminal INT,STRING,SEMI,COMMA,ID,SPACE,EOLFILE,ASSIGN,STRIN
4 non terminal prog,stmt,decln,varlist,type,s;
5 s::=prog { System.out.println("Valid declaration"); }
6 EOLFILE{System.exit(0);}
7 prog::=prog stmt |stmt ;
8 stmt ::= decln;
9 decln::=type SPACE varlist SEMI|ID ASSIGN STRING_VALUE SEMI|ID
10 type::=INT | STRING | FLOAT | CHAR;
11 varlist::=ID COMMA varlist | ID ;
12
```

File: declaration.cup

```
1 import java_cup.runtime.Symbol;
2 import java_cup.runtime.Scanner;
3 %
4 %cup
5 %eoval{
6 System.exit(0);
7 %eoval}
8 %
9 ":" {System.out.println("LA "+yytext());return new Symbol(sym);
10 ","
11 ":" {System.out.println("LA "+yytext());return new Symbol(sym);
12 "=" {System.out.println("LA "+yytext());return new Symbol(sym);
13 "\"[^\\\"\\"]\" {System.out.println("LA "+yytext());return new Symbol(sym);
14 "[0-9]+ {System.out.println("LA "+yytext());return new Symbol(sym);
15 "int" {System.out.println("LA "+yytext());return new Symbol(sym);
16 "float" {System.out.println("LA "+yytext());return new Symbol(sym);
17 "char" {System.out.println("LA "+yytext());return new Symbol(sym);
18 "string" {System.out.println("LA "+yytext());return new Symbol(sym);
19 "[a-z][a-zA-Z0-9]* {System.out.println("LA "+yytext());return new Symbol(sym);
20 "[\\n] {System.out.println("LA \\n");}
21 "EOF" {System.out.println("LA "+EOF);return new Symbol(sym);EOF}
```

PROBLEMS: 89 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Ln 4, Col 1 (selected) Spaces: 4 UFT-8 LF ⌂ Java ⌂ zxplore (zosmf) ⌂

2. Given below are some exercises that enhances the simple declaration statement to recognize a wide variety of other declarations.

- Multiline declaration: Extend the syntax rules to accept declaration statements with newline in between.

Example: int a,b,c;
 int c,d;

- Optional Definition: Extend the syntax rules to accept declaration statements of the form: var int tokens = 2, tip, args = 53; var string xyz = “hello”;

- Floating point: Extend the lexical and syntax rules to recognize floating point numbers (2.345) as FLOAT and in declaration statement.

Example: var float time = 12.34;

- Constant Declaration: Extend the syntax rules to recognize declaration of constants of the form: const int pi = 3.14;. Note, constant declaration can be done one at a time and definition during declaration is compulsory. i.e. const string xyz; is invalid whereas const string xyz = “hello”; is valid.

- NULL value: Extend your syntax rules to accept nil as a value during declaration

Example: var int i = NULL; var string s = NULL;

- Array Declaration: Extend your rules to recognize array declarations of the form: int value[10]; string colours[5];

- Array Initialization: Extend your rules to recognize array initializations of the form: int value[10] = {1, 25, 43, }; string colors[5] = {"red", "green", "blue", "orange", "yellow"};

The screenshot shows a code editor interface with the following details:

- EXPLORER:** Shows a project structure with files like `LAB6`, `java_cup`, `META-INF`, `.zshrc`, `commands`, `declaration`, `declaration.cup`, `input`, `java-cup-11b-runtime.jar`, `java-cup-11b.jar`, `Main.class`, `Main.java`, `parser.class`, `parser.java`, `parserCUPparser$action...`, `sym.class`, `sym.java`, `Yylex.class`, `Yylex.java`, and `Yylex.java~`.
- EDITOR:** Displays the `input` file content:

```
1 int a,b,c;
2 char x,y,z;
3 float p,q,r;
4 #
```
- TERMINAL:** Shows the command-line output of running the Java program:

```
(base) nspk@Crisp lab6 % java Main
LA int
LA a
LA b
LA c
LA ;
LA \n
LA char
LA x
LA ,
LA y
LA ,
LA z
LA ;
LA \n
LA float
LA p
LA q
LA d
LA r
LA ;
LA \n
LA EOF
Valid declaration
(base) nspk@Crisp lab6 %
```

Terminal 1:

```
(base) nspk@Crisp lab6 % jflex declaration
Reading "declaration"
Constructing NFA : 74 states in NFA
Converting NFA to DFA :
.
.
.
41 states before minimization, 38 states in minimized DFA
Old file "yflex.java" saved as "yflex.java"
Writing code to "yflex.java"
(base) nspk@Crisp lab6 % java java_cup.Main declaration.cup
----- CUP v0.11b 20160615 (GIT 4ac7450) Parser Generation Summary -----
0 errors, 0 warnings
18 terminals, 11 non-terminals, and 22 productions declared,
producing 46 unique parse states,
0 terminals declared but not used,
0 non-terminals declared but not used.
0 productions never reduced.
0 conflicts detected (@ expected).
Code written to "parser.java", and "sym.java".
----- (CUP v0.11b 20160615 (GIT 4ac7450))
Note: Main.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
(base) nspk@Crisp lab6 % java Main
LA var
LA
LA int
LA tokens
LA ;
LA tip
LA ,
LA args
LA =
LA ;
LA ;
LA \n
LA var
LA string
LA xyz
LA =
LA "hello"
LA ;
LA \n
LA EOF
Valid declaration
(base) nspk@Crisp lab6 %
```

Terminal 2:

```
(base) nspk@Crisp lab6 % javac *.java
Note: Main.java uses or overrides a deprecated API.
Note: Recompile with -Xlint:deprecation for details.
(base) nspk@Crisp lab6 %
```

Terminal 3:

```
(base) nspk@Crisp lab6 % java Main
LA int
LA a,b,c;
LA a
LA b
LA c
LA ;
LA \n
LA var
LA float
LA time
LA 12.25
LA ;
LA \n
LA EOF
Valid declaration
(base) nspk@Crisp lab6 %
```

Three screenshots of a Java development environment showing the progression of a grammar declaration.

Screenshot 1:

```

EXPLORER          Main.java 2  declaration.cup  commands  input  declaration  sym.java
LAB6
> java_cup
> META-INF
$ .zshrc
commands
declaration
input
java-cup-11b-runtime.jar
java-cup-11b.jar
Main.class
Main.java 2
parser.class
parser.java 9+
parser$CUP$parser$action...
sym.class
sym.java
Yylex.class
Yylex.java 9+

```

TERMINAL

```
(base) nsapk@Crisp lab6 % java Main
LA int
LA a
LA =
LA NULL;
LA ;
LA \n
LA float
LA abc
LA =
LA NULL;
LA ;
LA \n
LAEOF
Valid declaration
(base) nsapk@Crisp lab6 %

```

Screenshot 2:

```

EXPLORER          Main.java 2  declaration.cup  commands  input  declaration  sym.java
LAB6
> java_cup
> META-INF
$ .zshrc
commands
declaration
input
java-cup-11b-runtime.jar
java-cup-11b.jar
Main.class
Main.java 2
parser.class
parser.java 9+
parser$CUP$parser$action...
sym.class
sym.java
Yylex.class
Yylex.java 9+

```

TERMINAL

```
(base) nsapk@Crisp lab6 % java Main
LA int
LA value
LA [
LA @
LA ;
LA :
LA \n
LA string
LA [
LA colours
LA colours
LA [
LA 5
LA ]
LA ;
LA \n
LAEOF
Valid declaration
(base) nsapk@Crisp lab6 %

```

Screenshot 3:

```

EXPLORER          Main.java 2  declaration.cup  commands  input  declaration  sym.java
LAB6
> java_cup
> META-INF
$ .zshrc
commands
declaration
input
java-cup-11b-runtime.jar
java-cup-11b.jar
Main.class
Main.java 2
parser.class
parser.java 9+
parser$CUP$parser$action...
sym.class
sym.java
Yylex.class
Yylex.java 9+

```

TERMINAL

```
(base) nsapk@Crisp lab6 % java Main
LA int
LA value
LA [
LA @
LA 10
LA ;
LA :
LA \n
LA string
LA [
LA colors
LA colors
LA [
LA 5
LA ]
LA =
LA {
LA "red"
LA "green"
LA "blue"
LA "orange"
LA "yellow"
LA ;
LA 567
LA ]
LA ;
LA ;
LA \n
LA string
LA [
LA colors
LA [
LA 5
LA ]
LA =
LA {
LA "red"
LA "green"
LA "blue"
LA "orange"
LA "yellow"
LA ;
LA \n
LAEOF
Valid declaration
(base) nsapk@Crisp lab6 %

```

OUTLINE 127 △ 0

Ln 2, Col 57 Spaces: 4 UTF-8 LF () Java ⌂ explore (zosmf) ⌂ ⌂