

کامپایلر

فاز اول پروژه

فاطمه کمانی و سیده صدیقه مکی

۹۷۰۱۲۲۶۸۱۰۰۸ – ۹۷۰۱۲۲۶۸۰۰۳۳

برای انجام فاز اول پروژه ابتدا باید اسکندر تولید کنیم پس jflex و cup را نصب میکنیم .

پس از نصب jflex در پوشه examples فایللی به نام java.flex وجود دارد که اسکندر جاوا است در این پروژه از این فایل استفاده شده و تغییرات گفته شده در صورت پروژه بر این فایل اعمال شده .

در صورت پروژه به مواردی که اسکندر باید تشخیص دهد اشاره شده که در ادامه آنها را بررسی می کنیم

non case sensitive بودن:

با استفاده از آپشن caseless این خواسته بر آورده شد.

```
java.flex 23
26%public
27%class Scanner
28%extends sym
29
30%unicode
31
32%caseless
33
```

۱- اعداد مثبت، منفی و اعشاری:

```
tst.java 1 +1 -1 -0.4 +2.7
Console Problems Progress Debug Shell Search History Git Staging
<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (J
Lexing [tst.java]
line:1 col:1 ---PLUS--
token: line 1, column 1, sym: 75
line:1 col:2 --1--INTEGER_LITERAL--
token: line 1, column 2, sym: 108, value: '1'
line:1 col:4 ----MINUS--
token: line 1, column 4, sym: 76
line:1 col:5 --1--INTEGER_LITERAL--
token: line 1, column 5, sym: 108, value: '1'
line:1 col:7 ----MINUS--
token: line 1, column 7, sym: 76
line:1 col:8 --0.4--FLOATING_POINT_LITERAL--
token: line 1, column 8, sym: 109, value: '0.4'
line:1 col:12 ---PLUS--
token: line 1, column 12, sym: 75
line:1 col:13 --2.7--FLOATING_POINT_LITERAL--
token: line 1, column 13, sym: 109, value: '2.7'
line:1 col:16 ----EOF--
token: line 1, column 16, sym: 0
No errors.
```

۲- کلمات nice و hello

```
tst.java  java.flex
1 hELLO
2 NiCe

<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-
Lexing [tst.java]
line:1 col:1 --hELLO--HELLO--
token: line 1, column 1, sym: 4
line:2 col:1 --NiCe--NICE--
token: line 2, column 1, sym: 5
line:2 col:5 ---EOF--
token: line 2, column 5, sym: 0
No errors.
```

۳- فضای خالی یا White Space (توکن WS):

در فایل اصلی java.flex توکن های مربوط به comment و whitespace ایگنور میشدند و شناسایی نمیشدند. اما در صورت پروژه از ما خواسته شده بود که آنها را شناسایی کنیم.
پس برای آنها نیز return میگذاریم تا شناسایی شوند:

```
java.flex
297 {Comment} { return symbol(CM); }
298
299 /* whitespace */
300 {WhiteSpace} { return symbol(WS); }
301
```

```
tst.java  java.flex
1 hELLO FRom The OthEr Side

<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Jun 25, 20
Lexing [tst.java]
line:1 col:1 --hELLO--HELLO--
token: line 1, column 1, sym: 4
line:1 col:6 -- --WS--
token: line 1, column 6, sym: 2
line:1 col:7 --FRom--IDENTIFIER--
token: line 1, column 7, sym: 113, value: 'FRom'
line:1 col:11 -- --WS--
token: line 1, column 11, sym: 2
line:1 col:12 --The--IDENTIFIER--
token: line 1, column 12, sym: 113, value: 'The'
line:1 col:15 -- --WS--
token: line 1, column 15, sym: 2
line:1 col:16 --OthEr--IDENTIFIER--
token: line 1, column 16, sym: 113, value: 'OthEr'
line:1 col:21 -- --WS--
token: line 1, column 21, sym: 2
line:1 col:22 --Side--IDENTIFIER--
token: line 1, column 22, sym: 113, value: 'Side'
line:1 col:26 ---EOF--
token: line 1, column 26, sym: 0
No errors.
```

۴- علامت های پشتیبانی شده:

```
tst.java  java.flex
1  [(+&^%)]

<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (J
Lexing [tst.java]
line:1 col:1 --[--LBRACK--
token: line 1, column 1, sym: 25
line:1 col:2 --(--LPAREN--
token: line 1, column 2, sym: 34
line:1 col:3 --*--MULT--
token: line 1, column 3, sym: 29
line:1 col:4 --&--AND--
token: line 1, column 4, sym: 91
line:1 col:5 --^--XOR--
token: line 1, column 5, sym: 92
line:1 col:6 --%--MOD--
token: line 1, column 6, sym: 80
line:1 col:7 --)--RPAREN--
token: line 1, column 7, sym: 35
line:1 col:8 --]--RBRACK--
token: line 1, column 8, sym: 26
line:1 col:9 ----EOF--
token: line 1, column 9, sym: 0
No errors.
```

۵- عملگر ها:

```
tst.java  java.flex
1  [(+&^+.=/%)]

<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (J
Lexing [tst.java]
line:1 col:1 --[--LBRACK--
token: line 1, column 1, sym: 25
line:1 col:2 --(--LPAREN--
token: line 1, column 2, sym: 34
line:1 col:3 --*--MULT--
token: line 1, column 3, sym: 29
line:1 col:4 --&--AND--
token: line 1, column 4, sym: 91
line:1 col:5 --^--XOR--
token: line 1, column 5, sym: 92
line:1 col:6 --+--PLUS--
token: line 1, column 6, sym: 75
line:1 col:7 --=-MINUSEQ--
token: line 1, column 7, sym: 101
line:1 col:9 --/--DIV--
token: line 1, column 9, sym: 79
line:1 col:10 --|--OR--
token: line 1, column 10, sym: 93
line:1 col:11 --%--MOD--
token: line 1, column 11, sym: 80
line:1 col:12 --)--RPAREN--
token: line 1, column 12, sym: 35
line:1 col:13 --]--RBRACK--
token: line 1, column 13, sym: 26
line:1 col:14 ----EOF--
token: line 1, column 14, sym: 0
No errors.
```

۶- کامنت (توکن CM) :

```
tst.java  java.flex
1  //HELLO FROM thE oThEr side

<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (J
Lexing [tst.java]
line:1 col:1 --//HELLO FROM thE oThEr side--CM--
token: line 1, column 1, sym: 3
line:1 col:28 ----EOF--
token: line 1, column 28, sym: 0
No errors.
```

۷- وجود ۵ کلمه کلیدی:

```
tst.java
1 print;
2 Clock;
3 LocAtiOn;
4 uSErNaME;
5 OS;

Console
<terminated> TestLexer (2) [Java Application] /usr/lib/jvm/java-11-openj...
Lexing [tst.java]
line:1 col:1 --print--PRINT--
token: line 1, column 1, sym: 12
line:1 col:6 --;--SEMICOLON--
token: line 1, column 6, sym: 28
line:2 col:1 --Clock--CLOCK--
token: line 2, column 1, sym: 13
line:2 col:6 --;--SEMICOLON--
token: line 2, column 6, sym: 28
line:3 col:1 --LocAtiOn--LOCATION--
token: line 3, column 1, sym: 14
line:3 col:9 --;--SEMICOLON--
token: line 3, column 9, sym: 28
line:4 col:1 --uSErNaME--USERNAME--
token: line 4, column 1, sym: 15
line:4 col:9 --;--SEMICOLON--
token: line 4, column 9, sym: 28
line:5 col:1 --OS--OS--
token: line 5, column 1, sym: 16
line:5 col:3 --;--SEMICOLON--
token: line 5, column 3, sym: 28
line:5 col:4 ----EOF--
token: line 5, column 4, sym: 0
No errors.
```

کامپایل کلمات کلیدی:

```
java12.cup  tst.java
1 PuBLic cLass teST {
2   puBLic sTATic vOid MaIn(sTrInG[] aRgS) {
3     prInT;
4     cLoCK;
5     loCaTiOn;
6     usErNaME;
7     OS;
8   }
9 }
```

```
Console
<terminated> JavaParser (1) [Java Application] /usr/lib/jv...
Parsing [tst.java]
PRINT: HI THERE
CLOCK: 19:02:16
LOCATION: Asia/Tehran
USERNAME: anita
OS: Linux: 5.8.0-59-generic
No errors.
```

۸- وجود Identifier : به مراتب در عکس های بالا نشان داده شد.
بخش for و while: در این بخش دو گرامر fur و during اضافه شد.

```
java12.cup  tst.java
1 Public cLASS test {
2   PubLic sTATic VOID MaIn(STrInG[] ARGS) {
3
4     fUR I = 8 UntiL 12 WHerE
5     bEGiN
6     end
7
8     dUrIng i < u wherE
9     bEGIN
10    eNd
11  }
```

```
Console
<terminated> JavaParser (1) [Java Appli...
Parsing [tst.java]
No errors.
```

مشاهده می شود که کامپایل بدون خطا انجام شد.

تصویر از محیط JFlex :

```
java.flex 184
185 "hello" { return symbol(HELLO); }
186 "nice" { return symbol(NICE); }
187 "where" { return symbol(WHERE); }
188 "print" { return symbol(PRINT); }
189 "clock" { return symbol(CLOCK); }
190 "location" { return symbol(LOCATION); }
191 "username" { return symbol(USERNAME); }
192 "os" { return symbol(OS); }
193
194 "during" { return symbol(DURING); }
195 "fur" { return symbol(FUR); }
196 "where" { return symbol(WHERE); }
197 "begin" { return symbol(BEGIN); }
198 "end" { return symbol(END); }
199 "until" { return symbol(UNTIL); }
200
```

پس همانطور که مشاهده شد اسکنر قابلیت تشخیص تمامی موارد گفته شده در صورت پروژه را دارد.

تصویر از محیط java۱۲.cup:

در پوشه examples فایلی به نام java۱۲.cup نیز وجود دارد که گرامر های زبان جاوا هست . ابتدا ترمینال هایی که به java.flex اضافه کردیم را به آن اضافه میکنیم سپس برای گرامر های جدیدی که می‌خواهیم بنویسیم نان ترمینال های مورد نظر را اضافه میکنیم . (با استفاده از cup از java۱۲.cup در مسیر گفته شده sym.java و parser.java ایجاد میشود و این فایل ها را نیز باید به پروژه خود اضافه کنیم و برای اجرای فایل ها نیاز داریم که از jar file های موجود در فایل cup استفاده کنیم و آنها را نیز پروژه خود اضافه کنیم.)

تعریف ترمینال ها:

```
@ java12.cup 19
19 //////////////////////////////////////////////////
80terminal WS; //white space token
81terminal CM; //comment token
82terminal HELLO; //hello token
83terminal NICE; //nice token
84 //////////////////////////////////////////////////
85terminal FUR; //FUR token used for the requested for function
86terminal DURING; //DURING token used for the requested while function
87terminal WHERE; //where token
88terminal BEGIN; //BEGIN token
89terminal END; //END token
90terminal UNTIL; //UNTIL token
91//
92//
93terminal PRINT; //print_stmt// a keyword to print "HI THERE"
94terminal CLOCK; //clock_stmt// a keyword to print clock
95terminal LOCATION; //location_stmt// a keyword to print location
96terminal USERNAME; //username_stmt// a keyword to print username
97terminal OS; //os_stmt// a keyword to print os name and version
98 //////////////////////////////////////////////////
99
```

تعریف نان ترمینال ها:

```
@ java12.cup 100
100
101
102 //////////////////////////////////////////////////
103non terminal my_statement; // a statemnt used for declaring project things
104non terminal keyword_statement; // a statement used for keywords
105non terminal print_stmt; // a statement used for PRINT keyword
106non terminal clock_stmt; // a statement used for CLOCK keyword
107non terminal location_stmt; // a statement used for LOCATION keyword
108non terminal username_stmt; // a statement used for USERNAME keyword
109non terminal os_stmt; // a statement used for OS keyword
110//
111//
112non terminal fur_stmt; // a statement used for fur token
113non terminal during_stmt; // a statement used for during token
114 //////////////////////////////////////////////////
115
```

اضافه کردن نان ترمینال my_statement به گرامر statement :

```
java12.cup
617 statement ::= statement_without_trailing_substatement
618 | labeled_statement
619 | if_then_statement
620 | if_then_else_statement
621 | while_statement
622 | for_statement
623 | my_statement
624 ;
```

گرامر ها my_statement و keyword_statement :

```
java12.cup
972
973
974 //////////
975 my_statement ::=
976 | keyword_statement SEMICOLON
977 | during_stmt
978 | fur_stmt
979 ;
980
981 //////////////////////////////////
982 //////////////////////////////////
983 keyword_statement ::=
984 | print_stmt
985 | clock_stmt
986 | location_stmt
987 | username_stmt
988 | os_stmt
989 ;
990
```

گرامر های keyword ها و گرامر fur و during : برای قسمت دوم پروژه که باید گرامر های جدید تعریف میشد ، چون گرامر های for و while از قبل وجود دارد، به جای for و while از کلمات fur و during استفاده شد و گرامر های گفته شده نوشته شد.

```
@ java12.cup 88
985 | clock_stmt
986 | location_stmt
987 | username_stmt
988 | os_stmt
989 | ;
990
991 print_stmt ::=
992 | PRINT { : System.out.println("PRINT: HI THERE"); : }
993 ;
994 clock_stmt ::=
995 | CLOCK { :String timeStamp = new java.text.SimpleDateFormat("HH:mm:ss")
996 | .format(java.util.Calendar.getInstance().getTime());
997 | System.out.println("CLOCK: "+timeStamp); : }
998 ;
999 location_stmt ::=
1000 | LOCATION { : System.out.println("LOCATION: "+java.util.TimeZone.getDefault().getID()); : }
1001 ;
1002 username_stmt ::=
1003 | USERNAME { : System.out.println("USERNAME: " + System.getProperty("user.name")); : }
1004 ;
1005 os_stmt ::=
1006 | OS { : System.out.println("OS: " + System.getProperty("os.name") + ": " + System.getProperty("os.ver
1007 | ;
1008
1009 ///////////////
1010 ///////////////
1011 during_stmt ::=
1012 | DURING conditional_expression WHERE BEGIN block_statements_opt END
1013 ;
1014 fur_stmt ::=
1015 | FUR assignment UNTIL unary_expression WHERE BEGIN block_statements_opt END
1016 ;
```