Compiler Development (CMPSC 401)
Lexical Analysis: JFlex

Janyl Jumadinova

February 5, 2019
Getting jFlex

jFlex package and documentation can be obtained from www.jflex.de
jFlex Program Format

/* User code */

/* Options and declarations */

/* Lexical Rules */

1 User Code (e.g. import statements), included top of generated Java; often empty.
2 Options “Marcos” (named REs); code to be spliced into generated Java class.
3 Rule = Pattern + Action.
4 Pattern = Regular Expression.
5 Action = Snippet of Java code (Actions triggered whenever pattern matched).
<table>
<thead>
<tr>
<th>pattern</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>a</code></td>
<td>character <code>a</code></td>
</tr>
<tr>
<td>&quot;a&quot;</td>
<td>character <code>a</code> (even special chars.)</td>
</tr>
<tr>
<td><code>abc</code></td>
<td><code>a</code> followed by <code>b</code> followed by <code>c</code> (no explicit concat. symbol)</td>
</tr>
<tr>
<td>`a</td>
<td>b`</td>
</tr>
<tr>
<td><code>a*</code></td>
<td>zero or more rep. of <code>a</code></td>
</tr>
<tr>
<td><code>a+</code></td>
<td>one or more rep. of <code>a</code></td>
</tr>
<tr>
<td><code>a?</code></td>
<td>optional <code>a</code></td>
</tr>
<tr>
<td><code>(a)</code></td>
<td><code>a</code> itself</td>
</tr>
<tr>
<td><code>[abc]</code></td>
<td>any (one) of <code>a</code> or <code>b</code> or <code>c</code></td>
</tr>
<tr>
<td><code>[^abc]</code></td>
<td>any char. except <code>a</code>, <code>b</code> or <code>c</code></td>
</tr>
<tr>
<td>.</td>
<td>any char. except newline</td>
</tr>
</tbody>
</table>
jFlex Example

Pattern:

("+" | "-")?(0-9)+("." [0-9]+)?

Meaning?
jFlex Example

Pattern:

("+" | "-" )? [0-9]+("." [0-9]+)?

Meaning?

- Optional sign
- One or more digits
- Optional decimal point one or more digits
jFlex Notes

- RE . (dot): matches any character except newline.
- %standalone option: generated code includes a main method.
- Special characters, such as ( ) - + ^ [ ] ─ *, must be quoted if they appear as themselves.
- Use backslash to quote a single symbol (example: ).
- Can attach names (example: \{Digit\}) to REs for brevity/clarity.
- When multiple rules apply: take longest match (maximum munch), and use order of rule appearance to break ties.
- Material enclosed in %\{ and \}\% is included directly in the generated java program.
{ Example program in TINY language}
read num;
if num > 0 then
    factorial := 1;
    repeat
        factorial := factorial · num;
        num := num-1
    until num = 0;
write factorial
end

• TINY is a simple toy language.
• Uses Pascal-like syntax.
• if-then-end, if-then-else-end, repeat-until, assignment, read and write.
TINY Programming Language

- Semicolons are separators, not terminators.
- No declarations.
- Integer variables only.
- **Arithmetic expressions**: variables, constants, +, −, *, /, ()
- **Boolean expressions**: arithmetic expressions, <, =
- read, write operations, perform simple input/output.
- Comments enclosed in { }
Reserved Words: if, then, else, end, repeat, until, read, write
Special Symbols: + − / = < () ; :=
Numbers: One or more digits
Identifiers: One or more letters
Comments: Any sequence of symbols (other than \}) enclosed in \{
...\}