Tema 0 LFA 2015-2016
Expresii regulate

George Daniel MITRA

23 noiembrie 2016

Rezumat

Tema constă în familiarizarea cu generatorul de analizoare lexicale FLEX (sau jflex) prin analiza și completarea unui schelet de cod (sau reimplementarea lui pentru cei ambițioși).

1 Specificații temă

1.1 Cerință

Se dă un program care citește expresii regulate în forma infixată și le afișează în forma prefixată. Să se extindă funcționalitatea lui prin adăugarea a doi operatori postfixați, $^+$ și ?.

1.2 Specificații program

1.2.1 Intrări

Programul va citi dintr-un fișier numit „input” o expresie regulată în forma infixată.

Se consideră corect orice primește programul ca intrare.

1.2.2 Ieșiri

Programul va afișa la ieșirea standard expresia în forma prefixată.

1.3 FLEX

FLEX este o unealtă pentru generarea de analizoare lexicale. Variantele acceptate sunt:

- flex(C/C++) [1, 2, 3]
- jflex(Java) [4, 5, 6]

2 Noțiuni introductive

2.1 Limbajul de descriere

Limbajul este descris printr-o gramatică BNF și folosește următoarea convenție de culori:

- albastru - neterminali
- verde - operatori ai limbajului BNF și paranteze ajutătoare
- rosu - terminali (elemente care fac parte efectiv din limbajul descris)

Pentru a simplifica sintaxa, Se folosesc operatorii $*$, $+$ și ? cu semnificația din expresiile regulate.

2.2 Simbol, Alfabet, Șir

2.2.1 Simbol

Un simbol poate fi literă, cifră sau caracter special:
<symbol> ::= <lower-case letter> | <digit> | <other>
<lower-case letter> ::= (a | b | c | d | f | g | h | i | j | k | l | m | n | o | p | q | r | s | t | u | v | w | x | y | z)
<digit> ::= (0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9)
<other> ::= (! | # | $ | % | & | - | . | / | : | ; | < | > | = | @ | [|] | ^ | ‘ | ~)

2.2.2 Alfabet

Un alfabet este orice mulțime finită și nevidă de simboluri.
<alphabet> ::= { <symbol> (, <symbol>)* }

2.2.3 Șir

Un șir este o secvență finită de simboluri. Șirul vid se notează cu e, în timp ce alte șiruri reprezintă o concatenare de unul sau mai multe simboluri. Din acest motiv, e nu se consideră simbol, deci nu va face niciodată parte din niciun alfabet.
<word> ::= e | (<symbol>)+

3 Expresii Regulate(ER)

3.1 Descriere

Expresiile regulate reprezintă o metodă de reprezentare finită a limbajelor. Ele pot descrie orice succesiune finită de operații de reuniune, concatenare și Kleene Star.

Fie Σ un alfabet. O expresie regulată este o secvență finită de simboluri din $\Sigma \cup \{\cup, ^*, (,), \varnothing, e\}$, ținând cont de următoarele proprietăți:

1. $\varnothing$ și e sunt expresii regulate, reprezentând limbajul vid, $L_1 = \emptyset$, respectiv limbajul care conține doar șirul vid, $L_2 = \{e\}$;

2. $\forall a \in \Sigma$, a este expresie regulată, reprezentând limbajul ce conține un singur cuvânt format din simbolul a, $L = \{a\}$;

3. $\forall \alpha, \beta$ expresii regulate, $\alpha \cup \beta$ este expresie regulată, reprezentând reuniunea limbajelor descrise de expresiile regulate α și β, $\mathcal{L}(\alpha \cup \beta) = \mathcal{L}(\alpha) \cup \mathcal{L}(\beta)$;

4. $\forall \alpha, \beta$ expresii regulate, $\alpha\beta$ este expresie regulată, reprezentând concatenarea limbajelor descrise de expresiile regulate α și β, $\mathcal{L}(\alpha\beta) = \mathcal{L}(\alpha) \circ \mathcal{L}(\beta)$;

5. $\forall \alpha$ expresie regulată, α^* este expresie regulată, reprezentând aplicarea operatorului Kleene Star limbajului descris de expresia regulată α, $\mathcal{L}(\alpha^*) = \mathcal{L}(\alpha)^*$;

6. $\forall \alpha$ expresie regulată, (α) este expresie regulată. Parantezele cresc prioritatea operatorilor. Operatorii, ordonați de la prioritate maximă la minimă sunt: Kleene Star, operator de concatenare, operator de reuniune;

7. Orice altceva nu este expresie regulată.

Notații ajutătoare:

- Fie α o expresie regulată. Notăm $\alpha^+ = \alpha\alpha^*$.
- Fie α o expresie regulată. Notăm $\alpha? = (\alpha \cup e)$

3.2 Specificații

În temă, din cauza faptului că le folosim pentru a desemna elemente constitutive expresiilor regulate, caracterele $\{|, *, +, ?, O, e, (,)\}$ nu pot face parte din niciun alfabet. O expresie regulată se definește în felul următor:

<RE> ::= <alphabet> : <expression>

<expression> ::= O | e | <symbol> | (<expression> | <expression>) | (<expression> <expression>) | (<expression> *) | (<expression> +) | (<expression> ?) | (<expression>)

În temă, | reprezintă operatorul de reuniune, O reprezintă expresia limbajului vid, e reprezintă expresia limbajului care conține doar șirul vid.

Forma prefixată a unei expresii se reprezintă în felul următor:

<PRE > ::= phi | epsilon | (symbol ␣ <symbol>) | (union ␣ <expression> (␣ <expression>)+)| (concat ␣ <expression> (␣ <expression>)+) | (kleene ␣ <expression>) | (plus ␣ <expression>) | (optional ␣ <expression>)

În fișierul de intrare spațiile pot fi ignorate. În fișierul de ieșire, ele fiind separatori ai operanzilor, trebuie să apară.

4 Punctaj

Tema 0 este o temă de familiarizare cu instrumentul, așa că are un volum de muncă foarte redus și punctaj 0. Are ca termen de predare termenul temei 1.

5 Sugestii

Înainte de a vă uita pe cod, e recomandat să citiți secțiunile "Format of the input files" și "Start conditions" din manualul FLEX [2], respectiv "Lexical specification" și "A simple example..." din manualul jflex [5].

C: Orientați-vă după cum sunt folosite constantele din expression.h pentru celelalte tipuri de expresii. $^+$ și ? sunt foarte similare cu Kleene Star. Java: Implementați clase pentru cei doi operatori care sunt unari postfixați. Seamănă foarte mult cu Kleene Star.

Bibliografie

[1] flex homepage

[2] Lexical Analysis with Flex

[3] Using flex

[4] jflex homepage

[5] jflex user manual

[6] jflex user manual in japanese

[7] Laborator 1 SO: Makefile