Processadors de Llenguatges

Pràctica 1 – Implementació de la fase d'anàlisi lèxica

Teresa Alsinet
teresa.alsinet@udl.cat

Curs 2021-2022

Objectius

La implementació manual de la rutina d'anàlisi lèxica per a qualsevol llenguatge de programació és sempre una tasca costosa. L'objectiu d'aquesta pràctica és utilitzar una eina de generació automàtica d'analitzadors lèxics, les quals estan especialment dissenyades per ser utilitzades conjuntament amb una eina de generació automàtica d'analitzadors sintàctics.

Exercicis de programació

Exercici 1 (Lleguatge de programació)
Escolliu un llenguatge i, per a aquest llenguatge, implementeu una especificació lèxica que calculi el nombre d'ocurrències per a cada component del llenguatge. En particular, comptabilitzarem el nombre de paraules reservades, identificadors, constants (enteres, reals, de caràcter, de cadena -string- ....), operadors (aritmètics, lògics, relacionals, d'assignació ....), símbols delimitadors ('(', ')', '[', ']', '', '', ',', ';' ...), símbols de separació (blancs, tabuladors, new line ...) i comentaris, que apareixen en un programa escrit en el llenguatge. Podeu ignorar qualsevol altra component. Per a cada categoria de components del llenguatge, indiqueu la proporció que representa respecte del total analitzats. Proporcioneu el nom del fitxer d'entrada com millor us vagi depenent del sistema escollit. La sortida la podeu escriure directament a la sortida estàndard.

Exercici 2 (Llenguatge de la lògica proposicional)
Implementeu una especificació lèxica que reconegui els components del llenguatge fòrmules proposicionals definides sobre les variables proposicionals denotades mitjançant qualsevol lletra en majúscula, és a dir, sobre el rang [A-Z]

Consideracions:

- Els operadors a considerar són: la negació !, la conjunció $\wedge$, la disjunció $\vee$, la implicació que representarem amb el string "$->$", i la doble implicació que representrarem amb el string "$<->$".

- Les fòrmules proposicionals podran estar parentitzades, és a dir, els símbols de parèntesis "(" i ")" són elements del llenguatge.
- El caràcter nova línia actuarà com a marca de final fòrmula i la marca de final d'arxiu com a final d'entrada. El processat acaba amb la marca de final d'arxiu.
- Desprecieu els caràcters blancs i tabuladors existents a l'entrada.
- Definir un format per a incloure línies de comentaris.
- Al detectar un error, l'analitzador lèxic emetrà un missatge indicant el caràcter que l'ha produït i el número de línia del programa font on ha estat detectat. La tècnica de recuperació associada és el mode pànic: l'analitzador lèxic despreciarà tots els símbols fins el caràcter nova línia o la marca de final d'arxiu.
- Proporcioneu el nom del fitxer d'entrada com millor us vagi depenent del sistema escollit. La sortida seran únicament els possibles errors i la podeu escriure directament a la sortida estàndard.

Exemples de fòrmules lèxicament correctes són:

- $P \wedge Q <->(!R \vee (Q->T))$
- $P\wedge <->(!R \vee (Q->T))$
- $P \wedge Q <->(!R \vee (Q->T$
- $P \;\; Q <->(!R \;\; (Q->T))$

És a dir, a nivell lèxic no identifiquem possibles errors sintàctics. En canvi, **exemples de fòrmules lèxicament incorrectes són**:

- $p \wedge q <->(!R \vee (Q->T))$
- $P + Q * (!R \vee (Q->T))$
- $[P \wedge Q <->[!R \vee (Q->T)]]$
- $P \wedge Q <=>(!R \vee (Q => T))$
- $P \wedge Q <->(!R \vee (Q->T));$

Exercici 3 (Lleguatge de les expressions regulars)
Implementeu una especificació *lex* que reconegui els components del llenguatge expressions regulars sobre l'afabet de les lletres minúscules, és a dir, sobre el rang [a-z].

Consideracions:

- Els operadors a considerar són: | . $*$ + ? ( ) [ ] −, on el símbol . denota la concatenació.

- A nivell lèxic, representarem ϵ mitjançant la paraula reservada (string) "BUIDA" o "buida".
- El caràcter nova línia actuarà com a marca de final d'expressió regular i la marca de final d'arxiu com a final d'entrada. El processat acabarà amb la marca de final d'arxiu.
- Desprecieu els caràcters blancs i tabuladors existents a l'entrada.
- Definir un format per a incloure línies de comentaris.
- Al detectar un error, l'analitzador lèxic emetrà un missatge indicant el caràcter que l'ha produït i el número de línia del programa font on ha estat detectat. La tècnica de recuperació associada és el mode pànic: l'analitzador lèxic despreciarà tots els símbols fins el caràcter nova línia o la marca de final d'arxiu.
- Proporcioneu el nom del fitxer d'entrada com millor us vagi depenent del sistema escollit. La sortida seran únicament els possibles errors i la podeu escriure directament a la sortida estàndard.

Exemples de fòrmules lèxicament correctes són:

- $(a|\text{BUIDA}).c.(d.\text{buida})?$
- $a \;\; b \;\; (c|d)+$
- $((a|b|c).(\text{buida}|b + .c*)?$
- $(\text{buida}| + .c\ *)?$
- $[a - z]+$
- $[-]+$
- $|. * +$
- $aeiou$

És a dir, a nivell lèxic no identifiquem possibles errors sintàctics. En canvi, **exemples de fòrmules lèxicament incorrectes són**:

- $(A|\text{BUIDA}).C.(D.\text{buida})?$
- $[A - Z]+$
- $[a, b]+$
- $(a|\text{Empty}).c.(d.\text{buida})?$
- $(a|\text{BUIDA}).c.(d.\text{buida})?;$
- $a = b \;\; (c|d)+$

Exercici 4 (Preprocessador)
Estem interessats en dissenyar un preprocessador de llenguatges tipus C, el qual ha de permetre la inclusió d'arxius i l'expansió de constants sense paràmetres (etiquetes o constants simbòliques). Implementa una especificació lèxica, que agafi com a entrada un programa i que generi com a sortida el programa equivalent però sense les sentències de preprocessador esmentades. És a dir, el programa equivalent contindrà el codi complet dels fitxers inclosos en el programa mitjançant una sentència de preprocessador, i totes les referències a constants simbòliques substituides per la definició associada en el programa d'entrada. Proporcioneu el nom del fitxer d'entrada i de sortida com millor us vagi depenent del sistema escollit.

Exercici 5 Exercici opcional (+ 0,5 punts)
Amplieu la solució a l'exercici 4 amb el processat de macros amb paràmetres. Podeu definir les restriccions lèxiques que considereu oportunes. En aquest cas, prepareu un fitxer README indicant les característiques implementades.

Lliurament

La documentació a lliurar per a cada exercici de programació és la següent:

1. Especificació lèxica. Per implementar la solució podeu utilitzar l'eina de generació d'analitzadors lèxics que més us agradi: lex o flex (eina per generar analitzadors lèxics en C i C++), Jlex (eina per generar analitzadors lèxics en Java), PLY (implementació de lex en Python), Alex (eina per generar analitzadors lèxics en Haskell), camllex (eina per generar analitzadors lèxics en OCaml).
2. Mòduls auxiliars emprats en la implementació de la solució.
3. Joc de proves utilitzat per validar l'exercici.
4. Si ho considereu oportú, un fitxer README amb les particularitats pròpies de la vostra implementació.

Lliurament de la pràctica al `cv.udl.cat` dins d'activitats. Lliurar un arxiu comprimit que agrupi, mitjançant la utilitat `tar`, tots els fitxers fonts dels exercici de programació, els jocs de proves utilitzats per a la validació dels exercicis i els possibles fitxers README.

Avaluació

- La pràctica la podeu realitzar de manera individual o en grups de 2 o 3 persones. Indicar els components del grup.
- El pes d'aquesta pràctica és d'un 15% sobre la nota final de l'assignatura.
- La data límit per lliurar la pràctica és el 8 de març per a l'avaluació contínua. Pels que no la tingueu acabada el 8 de març, la podeu lliurar fins el 7 d'abril (data de l'examen del 1r parcial de l'assignatura).