Travaux Dirigés de Compilation n°4

ESIPE - Filière informatique

———

Analyse lexicale avec `antlr`

Le but de ce TD est de pratiquer l'utilisation d'`antlr` : on compare les opérateurs réticents avec les opérateurs classiques correspondants, et on apprend à modulariser un analyseur lexical en modes.

▸ Exercice 1. *Opérateurs réticents*

1. *Écrivez un programme* `antlr` *qui trouve comme lexèmes :*
   - *toutes les séquences entre crochets (comme celles dans le fichier* `error.log`*),*
   - *suffisamment d'autres séquences pour qu'il n'y ait pas d'erreurs de reconnaissance de tokens quand on exécute* `TestRig` *avec une option autre que* `-tokens`*.*

   On demande 2 versions :

   (a) *sans opérateur réticent dans la règle qui reconnait les séquences entre crochets*

   (b) *avec un opérateur réticent dans la règle qui reconnait les séquences entre crochets*

   Testez sur quelques lignes du fichier `error.log`*.*

2. *Écrivez un programme* `antlr` *qui trouve comme lexèmes :*
   - *toutes les séquences entre guillemets qui ne contiennent pas de caractères de fin de ligne ni de guillemets à l'intérieur,*
   - *suffisamment d'autres séquences pour qu'il n'y ait pas d'erreurs de reconnaissance de tokens quand on exécute* `TestRig` *avec une option autre que* `-tokens`*.*

   On demande 2 versions :

   (a) *sans opérateur réticent dans la règle qui reconnait les séquences entre guillemets*

   (b) *avec un opérateur réticent dans la règle qui reconnait les séquences entre guillemets*

   Testez sur le fichier `weblogic-stack-trace.log`*.*

3. *Écrivez un programme* `antlr` *qui trouve comme lexèmes :*
   - *toutes les séquences entre guillemets qui ne contiennent pas de guillemets à l'intérieur, sauf s'ils sont déspécialisés par un* `\`*,*
   - *suffisamment d'autres séquences pour qu'il n'y ait pas d'erreurs de reconnaissance de tokens quand on exécute* `TestRig` *avec une option autre que* `-tokens`*.*

On demande 2 versions :

(a) *sans opérateur réticent dans la règle qui reconnait les séquences entre guillemets*

(b) *avec un opérateur réticent dans la règle qui reconnait les séquences entre guillemets*

Testez sur le fichier `c-strings.c`.

▸ **Exercice 2.** ***Modes***

Écrivez un programme `antlr` *qui prend un programme C et affiche tous les noms de fonctions utilisés dans ce programme. Attention : si un nom apparait dans une chaîne entre guillemets ou dans un commentaire, que ce soit avec la syntaxe /* ... */ ou avec la syntaxe //, ce n'est pas un nom de fonction. Si l'on prend en entrée le programme suivant (*`test-td4-ex2.c`*) :*

```
/* la fonction plus(int,int) renvoie
   la somme de ses paramètres */
int plus
   (int a,int b) {
   return a+b;
}
int main (void) {
   printf ("plus(4,7)=%d\n",plus(4,7));
   getchar(); // getchar() attend un retour chariot
   return 0;
}
```

on devra obtenir la liste suivante : `plus main printf plus getchar`

Indication : définir 3 modes pour

— *les chaines de caracctères entre guillemets,*

— *les commentaires en* `//` *sur une ligne,*

— *les commentaires en* `/*` *et* `*/`.