NF11 – TP5 : GENERATION D'ANALYSEUR LEXICAL ET SYNTAXIQUE (4EME, 5EME SEANCES)

ETAPE 4 : PROCEDURES ET FONCTIONS

On considèrera que toutes les procédures et les fonctions doivent être déclarées avant le programme principal. Sinon, il faudrait plusieurs visites de l'arbre de dérivation.

REPRESENTER

- Les procédures sans et avec paramètres.

  On sera amené à créer une classe Procedure. Un objet de type Procedure devra conserver lors du parsing de la déclaration d'une procédure tous les attributs nécessaires à son exécution. Toute variable utilisée dans une procédure sera considérée locale à la procédure. Elle devra masquer lors de l'exécution de la procédure une variable de même nom du programme principal et de tout autre procédure active (une procédure peut en appeler une autre).

  Quel est le type de passage de paramètres que vous envisagez ?

  Exemple de procédure :

```
pour tracecarre :n
 repete 4 [
   av :n td 90
 ]
fin
tracecarre (100) donne "cote 200 tracecarre (:cote)
```

  Pour des raisons d'ambiguité on encadrera les paramètres effectifs lors de l'appel d'une procédure par des parenthèses.

- Les fonctions. Les procédures et les fonctions sont appelées différemment. Quel est le problème spécifique des fonctions ?

  Exemple :

```
pour carre :c
  rends :c * :c
fin
av carre 10
```

  L'instruction rends d'une fonction doit arrêter l'exécution de sa liste d'instructions d'une fonction. On activera les méthodes `stopIteration` et `continueIteration` héritée de la classe de base `LogoStoppableTreeVisitor` dans les méthodes de visite adéquates. Elles permettent d'arrêter et de reprendre la visite des nœuds.

  Exemple :

```
pour doublesi :n
  si :n >= 100 [rends 2 * :n]
  rends :n
fin
```

  Dans cet exemple, doublesi (100), doit calculer 200.

Analysez des instructions comme av doublesi 90 + 20. Sans parenthèse, elle peut a priori se réécrire comme av 110 ou comme av 220 suivant l'ordre de priorité des opérateurs. On sera donc amenés à ajouter des parenthèses autour des paramètres des fonctions pour lever l'ambiguïté.

- Une table des procédures enregistre la déclaration des procédures et des fonctions (ou deux tables séparées si on autorise une procédure et une fonction à avoir le même nom. Peut-on autoriser la surcharge des procédures (même nom mais nombre de paramètres différents).
- Une prodédure ou une fonction peut appeler une autre procédure. Il faut donc gérer la pile d'exécution d'un programme. Au moment de l'appel d'une procédure ou d'une fonction, il faut créer une nouvelle table et y insérer la valeur des paramètres effectifs. Elle est ensuite empilée sur la pile d'exécution. Les variables utilisées dans la procédure ou la fonction seront stockées dans cette table en sommet de pile.
- La table du programme principal devra être empilée avant la visite de l'arbre. Modifier la recherche et l'initialisation d'une variable pour qu'elles soient faites uniquement par rapport à la table en sommet de pile. Ainsi une variable locale ou un nom de paramètre effectif masquera une variable globale de même nom.
- Les procédures récursives. Vérifier si la modélisation actuelle vous permet de créer des procédures et des fonctions récursives.

ANALYSE SEMANTIQUE

Lors de l'utilisation des procédures et des fonctions il faudra prendre en compte les erreurs possibles et utiliser la même stratégie que précédemment pour les traiter. Lors de l'exécution, les problèmes sémantiques suivants peuvent arriver :

- L'appel à une procédure non déclarée
- L'arité d'une procédure ou d'une fonction non respectée (nombre de paramètres formels et effectifs différent, à relier éventuellement à la surchage des procédures et fonctions).

EVALUATION DU PROJET

- Rendre sur Moodle :
  - La grammaire
  - Le LogoTreeVisitor
  - Un fichier contenant ou ou plusieurs programmes et une copie écran de leur réalisation.