Théorie des langages et automates

Projet

Informations pratiques

Ce projet est à réaliser en binôme. Il doit être rendu au plus tard le **mardi 22 décembre 2020**, sur la page du cours sur Celene.

Votre projet doit être rendu sous forme de fichier zip, contenant :

1/ Les fichiers du projet en Java
- Les fichiers sources doivent être correctement indentés et commentés
- Ne pas laisser de code inutile, jamais appelé ou en commentaire.
- Faire commencer les noms de classe par une majuscule

2/ un document PDF décrivant
- La grammaire de votre langage
- Les ensembles first et follow des symboles terminaux et non terminaux
- La table d'analyse LL

Vous devrez implémenter une analyse lexicale et syntaxique descendante en vous inspirant des TP, et ne pas utiliser de générateur de code comme par exemple JavaCC, SableCC, Yacc, Bison ou ANTLR, ni faire appel à des librairies extérieures au JDK.

Présentation

Le langage logo est un langage de programmation créé en 1967 par Wally Feurzeig, Seymour Papert et Cynthia Solomon. Son concept le plus connu est celui d'une tortue permettant de tracer des dessins.

Les propriétés de la tortue sont notamment ses coordonnées et son orientation (en degré) sur la zone de dessin.

Dans ce projet, vous implémenterez ce langage logo réduit à quelques instructions permettant l'utilisation de la tortue.

Pour commencer, vous devez partir du squelette de projet, nommé projet.zip et disponible sur la page du cours sur Celene, modifier certains des fichiers sources du squelette et y ajouter d'autres fichiers sources.

Ce squelette de projet implémente :

une interface graphique en Java Swing	À ne pas modifier
la création « en dur » d'un arbre syntaxique (méthode exampleAst)	À remplacer par une analyse lexicale et syntaxique descendante portant sur le programme en logo saisie par l'utilisateur dans l'interface graphique.
un interpréteur de cet AST (méthode evalRoot)	À compléter

Grammaire, analyse lexicale et syntaxique

Vous devez écrire une grammaire prenant en charge les instructions suivantes :

Instruction	Action à réaliser
forward *n*	Avance la tortue de *n* pixels
left *n*	Tourne la tortue de *n* degrés vers la gauche
right *n*	Tourne la tortue de *n* degrés vers la droite
repeat *n* […]	Répète *n* fois les instructions entre crochets
color *n*	Change la couleur du tracé, *n* étant une valeur numérique de 0 à 12, à faire correspondre aux couleurs prédéfinies dans la classe java.awt.Color
procedure *ident* […]	Défini une procédure nommée *ident*, dont les instructions se trouvent entre crochets
call *ident*	Si une procédure nommée *ident* a été au préalable définie, appelle cette procédure (c'est à dire : exécute les instructions de cette procédure).

Votre analyse lexicale et syntaxique descendante ne doit pas intepréter les instructions (c'est à dire ne pas effectuer de tracé), mais doit retourner un AST correspondant aux instructions et construit avec les mêmes classes que l'exemple en dur.

Cet AST doit être passé à la méthode evalRoot qui interprète l'AST, c'est à dire qu'elle effectue les tracés selon le contenu de l'AST.

Interpréteur

L'interpréteur (méthode evalRoot) interprète un AST dont les nœuds correspondent aux instructions du langage. Dans le squelette du projet, deux types de ces nœuds ne sont pas traités :

- les nœuds de type nColor, changeant la couleur du tracé
- les nœuds de type nCall, effectuant un appel de procédure

Vous devrez modifier l'interpréteur pour ces nœuds soient correctement traités.

Exemple de programme