ALGORITHMIQUE ET PROGRAMMATION

PROJET COMPILATEUR LOGO

ANTOINE SCHERRER

1 INTRODUCTION

Le but de ce projet est d'écrire un **compilateur** de programme Logo, c'est-à-dire un programme qui prend en entré un fichier exprimé en langage `LOGO-ELDLC`, et qui fourni en sortie un fichier `SVG`. Le langage Logo (`http://en.wikipedia.org/wiki/Logo_programming_language`) est né dans les années 1960 avec comme objectif une approche pédagogique de la programmation (le but était d'apprendre aux enfants à programmer). De nombreuse déclinaisons et outils ont été proposées, mais nous allons ici nous cantonner à un ensemble réduit d'instructions, et c'est ce sous-ensemble qui sera dénommé `LOGO-ELDLC`.

Le but de ce projet est d'une part de réaliser un premier projet informatique de volume non négligeable, d'autre part d'acquérir des compétences élémentaires en transformation de programme source à source.

2 LE LANGAGE LOGO-ELDLC

Le langage `LOGO-ELDLC` a pour objectif de dessiner un graphique, en exprimant le chemin que va suivre le crayon. Historiquement ce crayon est représenté par une tortue. Le langage comporte 4 instructions, toutes avec paramètres, `FORWARD`, `LEFT`, `RIGHT` et `REPEAT`.

- L'instruction `FORWARD 10` fait avancer le crayon de 10 points dans la direction courante.
- L'instruction `LEFT 5` fait tourner la direction courante de 5 degrés à gauche (sens inverse des aiguilles d'une montre).
- L'instruction `RIGHT 5` fait tourner l'orientation courante de 5 degrés à droite (sens des aiguilles d'une montre).
- L'instruction `REPEAT 10 [ xxx ]` répète 10 fois la suite d'instructions entre crochet.

Il n'y a pas de point-virgule en `LOGO-ELDLC`, les séparateurs valides sont l'espace, le retour chariot ou la tabulation.

La figure 1 donne un exemple de programme avec le dessin associé.

```
FORWARD 100
REPEAT  4 [FORWARD 50 LEFT 90]
FORWARD 100
```

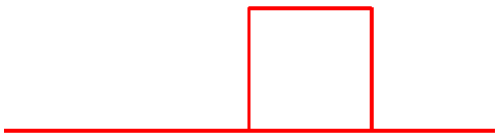

FIGURE 1 – Un programme `LOGO-ELDLC` simple et la représentation graphique décrite par le programme.

3 GÉNÉRATION D'UN FICHIER AU FORMAT SVG

SVG (*Scalable Vector Graphics*) est un format de données conçu pour décrire des ensembles de graphiques vectoriels basé sur XML. Vous pouvez vous familiariser avec la syntaxe des fichiers SVG en utilisant wikipédia. Voici par exemple le fichier SVG associé à la figure 1.

Les fichiers SVG peuvent être ouvert avec un navigateur web.

```
<?xml version="1.0" encoding="utf-8"?>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="200" height="200">
<title>Exemple LOGO</title>
<desc>Du LOGO.</desc>
<line x1="100.000" y1="100.000" x2="200.000" y2="100.000" stroke="red" />
<line x1="200.000" y1="100.000" x2="250.000" y2="100.000" stroke="red" />
<line x1="250.000" y1="100.000" x2="250.000" y2="150.000" stroke="red" />
<line x1="250.000" y1="150.000" x2="200.000" y2="150.000" stroke="red" />
<line x1="200.000" y1="150.000" x2="200.000" y2="100.000" stroke="red" />
<line x1="200.000" y1="100.000" x2="300.000" y2="100.000" stroke="red" />
</svg>
```

FIGURE 2 – Exemple de code SVG pouvant correspondre au programme LOGO-ELDLC de la figure 1

4 DÉMARCHE DE TRAVAIL

Je vous propose ici une démarche de travail suivant le principe de *diviser pour régner*, c'est-à-dire en décomposant le projet en petits problèmes pas trop difficile à résoudre. Libre à vous de suivre cette démarche ou non, l'objectif reste d'écrire un programme fonctionnel répondant aux attentes.

1. Écrire un programme qui génère un fichier SVG qui contient une ligne horizontale.
2. Écrire un programme qui lit un programme LOGO-ELDLC en entrée mais qui ne comprend qu'une instruction FORWARD avec un paramètre, et qui génère le fichier SVG associé (une ligne de taille variable).
3. Écrire un programme qui lit un programme LOGO-ELDLC en entrée et qui comprend une séquence d'instructions FORWARD avec paramètres, et qui génère le fichier SVG associé (toujours une ligne de taille variable).
4. Ajouter la gestion de l'instruction RIGHT. Vous aurez besoin de mobiliser vos souvenirs de trigonométrie !
5. Ajouter la gestion de l'instruction LEFT. vous pouvez maintenant créer des formes intéressantes !
6. A ce stade, vous arrivez à la partie plus complexe, avec la gestion de l'instruction REPEAT. Dans un premier temps cherchez à gérer un programme qui ne contient pas de REPEAT dans l'instruction REPEAT. Vous devriez maintenant être capable de générer le SVG pour l'exemple donné plus haut.
7. Vous devez maintenant terminer votre programme en gérant plusieurs niveaux d'imbrication d'instructions REPEAT, pour cela vous serez presque obligé d'utiliser une fonction **récursive**.
8. Maintenant que vous avez terminé votre programme, je vous propose de construire des fichiers LOGO-ELDLC pour le tester. Essayer par exemple de dessiner :
   - ⋆ Un cercle
   - ⋆ Le logo de l'École LDLC (sans le texte)
   - ⋆ Le logo de l'École LDLC (avec le texte)
   - ⋆ etc.

5 EXTENSIONS

Une fois le programme de base terminé, je vous propose d'imaginer des extensions en laissant libre cours à votre imagination. Je vous donne quelques exemples ci-dessous :

- ⋆ Ajouter la possibilité de lever le crayon lors du tracé
- ⋆ Ajouter la possibilité de changer la couleur
- ⋆ etc.