Théorie des langages et automates

Projet

Informations pratiques

Ce projet est à réaliser en binôme. Il doit être rendu sur Celene au plus tard le **vendredi 10 janvier 2020 à 18h**, sous forme d'une archive eclipse.

Ce projet doit être implémenté en Java, avec une interface graphique en JavaFX ou Swing (si vous connaissez bien cette librairie).

Pensez à indenter correctement vos fichiers, à ne pas laisser de code inutile ni de fichier vide, et à faire commencer les noms de classes par une majuscule.

Vous décrirez votre grammaire dans les commentaires de la classe d'analyse syntaxique.

Introduction et exemples

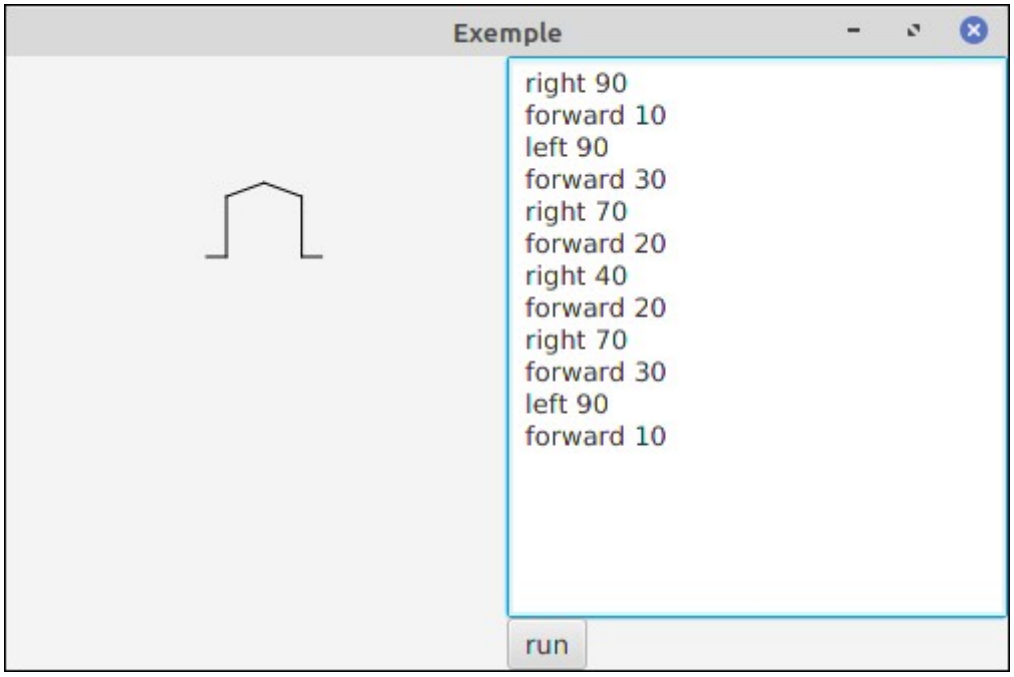
Le langage logo est un langage de programmation créé en 1967 par Wally Feurzeig, Seymour Papert et Cynthia Solomon. Son concept le plus connu est celui d'une tortue réalisant des dessins.

La tortue est un objet (au sens de la programmation objet). Ses propriétés sont notamment ses coordonnées et son orientation (ici en degré) sur la zone de dessin.

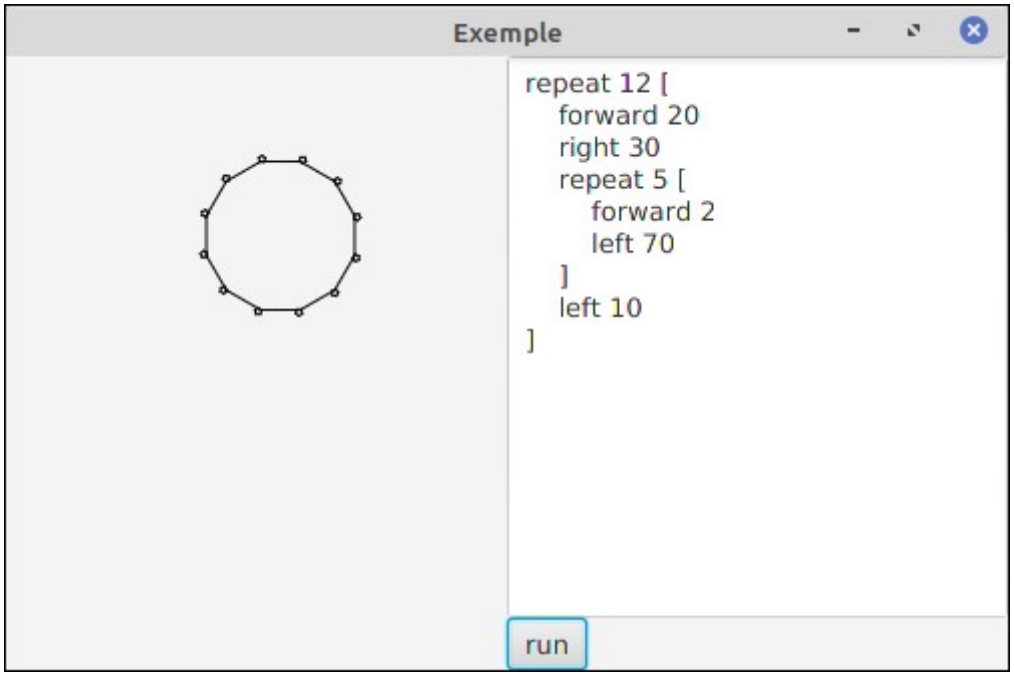
Pour ce projet, vous implémenterez un interpréteur logo réduit à quelques instructions permettant l'utilisation de la tortue.

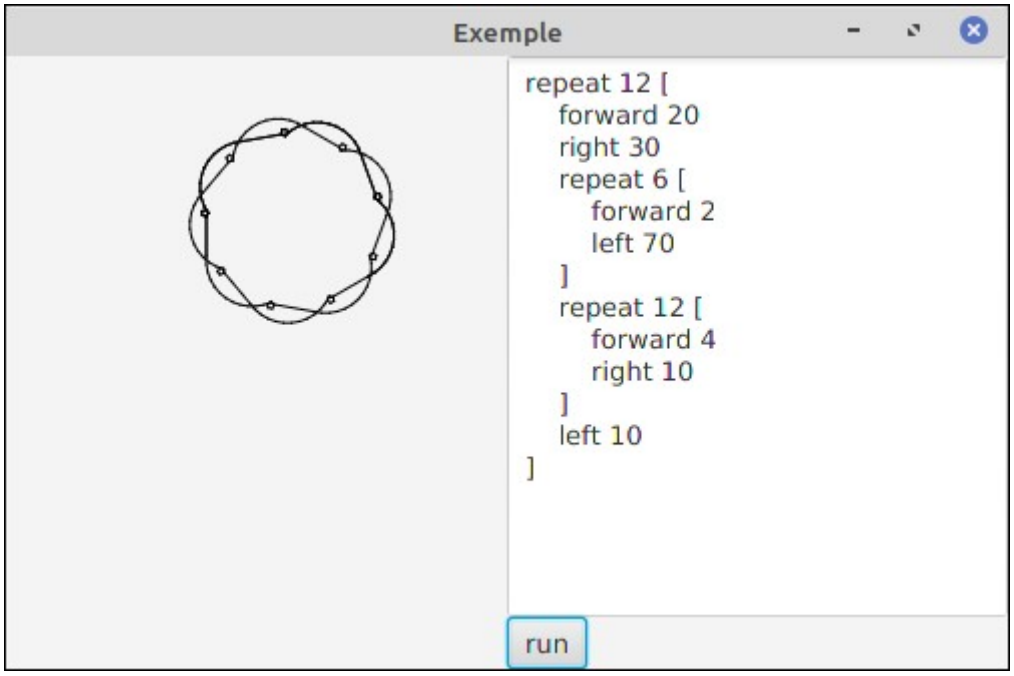
Instruction	Action à réaliser
forward *n*	Avance la tortue de *n* pixels
left *n*	Tourne la tortue de *n* degrés vers la gauche
right *n*	Tourne la tortue de *n* degrés vers la droite
repeat *n* […]	Répète *n* fois les instructions entre crochets

Voici différents exemples de programme en logo :



L'une des principales difficultés est l'implémentation de l'instruction repeat. Les blocs d'instruction repeat peuvent être imbriqués, comme le montrent les deux exemples suivants :





Etapes de réalisation

L'analyse lexicale et l'analyse syntaxique doivent implémenter des algorithmes vus dans les chapitres 8 (Analyse lexicale) et 9 (Analyse syntaxique descendante) du cours. Il est conseillé de reprendre et adapter les programmes que vous avez réalisés lors des TP.

Les erreurs qui peuvent survenir durant l'analyse lexicale et l'analyse syntaxique peuvent être affichés dans la console (à l'aide de System.out.println)

Pour implémenter l'analyse lexicale et l'analyse syntaxique, vous ne devez pas utiliser des générateurs de code comme par exemple JavaCC, SableCC, Yacc, Bison ou ANTLR, ni faire appel à des librairies extérieures au JDK.

Analyse lexicale

Adapter et simplifier l'analyse lexicale réalisée en TP. La reconnaissance des mots-clés peut être implémentée à partir de la ligne 102 de Lexer.java, en comparant la variable ***buf*** avec les mots-clés attendus, et en produisant les tokens correspondants (au lieu de ***TokenClass.ident*** en ligne 104).

Analyse syntaxique

1/ Afficher l'arbre syntaxique au fil de la reconnaissance des productions (de la même façon que dans le TP3 : à l'aide de System.out.println).

2/ La tortue ne doit pas être gérée comme un automate, mais comme un objet qui possède des attributs (orientation, coordonnées x,y), et des méthodes.

Faire agir cet objet tortue au fil de la reconnaissance des productions, sans tenir compte des instructions ***repeat*** (pour l'instant).

3/ Tenir compte des instructions entre crochets faisant suite aux instructions ***repeat***, en les faisant fonctionner une seule fois, comme si n'y avait pas de répétition (pour l'instant).

4/ Implémentation complète de ***repeat***.

L'analyse syntaxique descendante se base sur un *ArrayList<Token> tokens* et une variable *int pos* pour connaître le prochain token non lu.

Il est possible d'implémenter une boucle de façon simple (et peu efficace) en faisant « rejouer » l'analyse syntaxique sur les tokens entre crochets, et ce, en agissant sur la variable *pos*.

N° token	Token
0	avance
1	30
2	tourne
3	70
4	repeat
5	15
6	[
7	avance
8	30
…	…
n	]
…	...

```
reprise = 7;      // définition du point de reprise
                  // (premier token après le crochet ouvrant)

for(int i=0;i<15;i++) {
        pos = reprise;   // positionne le prochain token à lire
                         // au point de reprise


    reconnaissance des productions du corps de la boucle

}
```

Remarques sur JavaFX

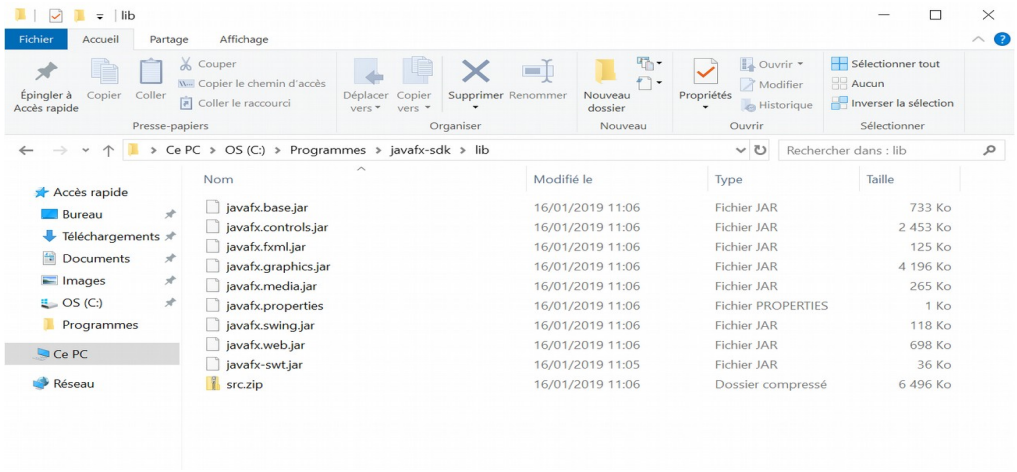
Si vous utilisez une version du JDK ≥ 11, ou un OpenJDK, JavaFX peut ne pas être présent dans ce JDK. Dans ce cas :

- Ubuntu : installer le paquet openjfx

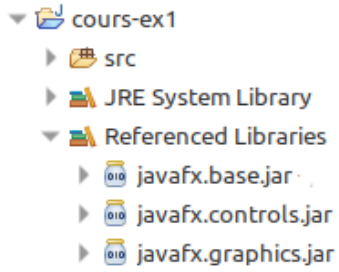
- Mac et Windows : télécharger sur la page https://gluonhq.com/products/javafx/ le « JavaFX Windows SDK » ou le « JavaFX Mac OS X SDK »

En supposant que l'archive est décompressée dans le dossier **C:\Program Files\javafx-sdk**

- les fichiers jar vont se trouver dans C:\Program Files\javafx-sdk\lib



- ajouter les jar externes suivants au projet eclipse :



- dans « Run configuration … », onglet Arguments, « VM Arguments », ajouter :

```
--module-path "C:\Program Files\javafx-sdk\lib" --add-modules javafx.controls
```