Manuel utilisateur

Natural to Python Compiler

Nino Jeannet et Sébastien Peiris

Prérequis

Pour utiliser et compiler un code source avec le compilateur "Natural to Python" assurez vous d'avoir le package PLY installé sur votre machine. Si ça n'est pas le cas, il suffit de l'installer à l'aide de pip.
pip install ply

Dans le cas où vous souhaitez avoir une représentation visuelle des arbres syntaxiques, il vous faut installer le logiciel Graphviz ainsi que le module python PyDot qui permet d'utiliser Graphviz.

Nous avons utilisé la version 3.5+ de python la version 2.7 étant dépréciée en 2020.
Voici les versions de modules nécessaires :

- Graphviz 2.38**
- ply 3.11
- pydot 1.4.1

** Pour graphviz, il est obligatoire que le chemin de son exécutable soit dans la variable d'environnement PATH.
(exemple avec le dossier d'installation par défaut : C:/Program Files (x86)/Graphviz2.38/bin/).
Sans cela, le programme ne pourra pas fonctionner.

Pour que ply fonctionne correctement, il est important de créer un fichier "generated" à la racine du projet.

Compilation

Créez votre code en Natural de sorte qu'il respecte les conditions suivantes :

- Programme simple et séquentiel ne contenant pas de fonctions et d'appels à un programme externe.
- Boucles de type for uniquement
- Respect des règles de syntaxe du langage

Si tous les points ci-dessus sont respectés, il vous suffit de lancer, en ligne de commande, le fichier *compiler.py* avec en argument votre fichier texte contenant votre code. Un fichier ".py" du même nom sera instantanément créé et contiendra le code traduit en python. (exemple: py compiler.py test.txt)

Pour voir les différentes étapes de l'application il est possible d'exécuter les différentes parties :

- lex.py (exemple : `py lex.py test.txt`) : affiche les lexèmes du programme passé en argument.
- naturalParser.py (exemple : `py naturalParser.py test.txt`) : génére un fichier pdf contenant le graph de l'arbre du programme.
- threader.py (exemple : `py threader.py test.txt`) : génére un fichier pdf contenant le graphe de l'arbre du programme avec la couture (flux d'éxecution).
- compiler.py (exemple : `py compiler.py test.txt`) : génére le fichier test.py avec le code compilé en python.

Fichiers de tests

Afin de simplifier la visualisation de notre projet, nous avons préparés 3 fichiers de tests afin de montrer les différentes fonctionnalités:

- test.txt : contient condition if imbriquées, division pas 0, variable utilisée avant l'instanciation
- test1.txt : contient boucle for avec 2 conditions if imbriquées, affichage de variable et de texte et valeur float (2,45)
- test2.txt : contient plusieurs boucles, boucles imbriquées, scope inatteignable dans l'imbrication, scope inatteignable dans une boucle différente.