Maël GAONACH - Paul BAVAZZANO

Résumé de chaque séance :

- Séance 1 (02/10/2020) : Première lecture de l'article et prise en main par un premier exemple : celui-ci a été rédigé dans ses deux versions en haskell mais nous n'avons pas encore eu le temps de tester son exécution. Prochaine étape : valider le fonctionnement de cet exemple et en faire un second un peu plus complexe.
- Séance 2 (08/10/2020) : Validation du premier exemple et rédaction/test de deux autres exemples plus complets. Prochaine étape : définir le modèle d'arbre syntaxique pour la partie Java.
- Séance 3 (09/10/2020) : Création du diagramme de classe de l'arbre syntaxique et génération du squelette de code. Prochaine étape : compléter le code Java.
- Séance 4 (15/10/2020) : Modification du diagramme de classe de l'arbre syntaxique. Construction de l'arbre syntaxique des types primitifs et de l'exemple 1. Prochaine étape : continuer la construction de l'arbre syntaxique de l'exemple 1 et commencer la construction des suivants.
- Séance 5 (16/10/2020) : Construction d'une première grammaire (type primitif, fonction, classe) et construction d'exemples d'utilisation. Prochaine séance : continuer la défintion de la grammaire.
- Séance 6 (20/10/2020) : Finition d'une première version de la grammaire. Construction du diagramme de classes (version 1.3) correspondants sur Visual Paradigm, génération du code Java correspondant. Prochaine séance : instancier un arbre syntaxique avec les classes Java obtenues.
- Séance 7 (05/11/2020): Finalisation de la construction de l'arbre syntaxique du programme (exemples 1, 2 et 3).
- Séance 8 (10/11/2020): Correction de l'arbre syntaxique. Création de la méthode `toHaskell()` permettant d'obtenir un fichier contenant le code Haskell généré. Développement du passage de l'arbre syntaxique avec classes de type à un arbre syntaxique sans classe de type.