Paradigmes et Langages de Programmation

Haute École d'Ingénierie et de Gestion du Canton de Vaud

Devoir 5

2023

1 Introduction

Ce devoir va consister pour vous en la première étape de l'implémentation Haskell d'un interpréteur d'un langage de programmation. À cet égard, il vous est demandé de spécifier la syntaxe d'un langage de programmation fonctionnelle à partir de zéro et d'implémenter les phases d'analyse lexicale et syntaxique au moyen d'outils tiers.

2 Parsing

Dans ce devoir, on vous propose de concevoir un langage de programmation caractérisé par des constructions de programmation typiques du paradigme fonctionnel. Celui-ci a pour but de vous faire pratiquer la conception des langages de programmation telle qu'étudiée jusqu'à maintenant. La donnée du problème se veut être délibérément concise et exige de votre part un certain degré d'investissement quant à l'élaboration de votre solution et la recherche voire demande de compléments d'information. Notez que tout comportement non spécifié dans l'énoncé signifie que vous pouvez l'adresser comme bon vous semble, pour autant que cela n'affecte en rien la robustesse, la pertinence et la cohérence de votre solution. En cas de doute, utilisez sans autre le forum de discussions pour poser vos questions.

2.1 Spécification

Votre langage de programmation fonctionnelle doit fournir deux catégories de constructions de programmation :

Déclaration	**Expression**
Variable constante	Littéraux entiers, booléens et tuples
Fonction à un ou plusieurs paramètres	Référence à des symboles
	Application de fonctions
	Expressions parenthésées et conditionnelles
	Déclarations locales avec un corps
	Pattern matching avec les motifs universel, variable, littéral et tuple
	Opérations unaires et binaires d'arithmétique, de logique et de comparaison

Le langage doit être statiquement et explicitement typé. Il doit inclure des annotations de type lors de chaque déclaration, qu'il s'agisse de variable, de paramètre ou de valeur de retour.

2.2 Lexer

Écrivez un module Haskell, **lexer.x**, qui expose une fonction permettant d'analyser lexicalement des termes de votre langage fonctionnel. Pour ce faire, vous utiliserez la bibliothèque Alex en vous appuyant sur la grammaire lexicale décrite au préalable. Vous devrez également définir une représentation intermédiaire décrivant les unités lexicales (*tokens*) d'un terme que vous exposerez et utiliserez lors de tâches ultérieures.

2.3 Parser

Écrivez un module Haskell, **parser.y**, qui expose une fonction permettant d'analyser syntaxiquement des termes de votre langage fonctionnel. Pour ce faire, vous utiliserez la bibliothèque Happy en vous appuyant sur la grammaire syntaxique décrite au préalable. Vous devrez également définir une représentation intermédiaire décrivant l'arbre syntaxique abstrait d'un terme que vous exposerez et utiliserez lors de tâches ultérieures.

2.4 Cheatsheet

Écrivez un document Markdown, **cheatsheet.md**, qui décrit la syntaxe de votre langage de programmation au moyen d'exemples. Il doit servir de guide illustrant ce qu'il est possible d'exprimer en termes de code à travers votre langage. Ce document peut également être utilisé pour expliquer et justifier vos choix de conception.

3 Évaluation

Pour ce devoir, l'évaluation de votre langage s'appuiera sur les critères suivants :

- Éxactitude — Le langage est correct au sens des exigences énoncées
- Ambiguïté — Le langage est dépourvu d'ambiguïté
- Complexité — Le langage est facile à lire et à comprendre
- Documentation — Le langage est documenté de manière pertinente
- Style — Le langage est implémenté dans un style fonctionnel

Chaque critère aura le même poids sur le nombre de points obtenus pour ce devoir. Un critère vous rapportera au maximum un point sur le total de la note, qui sera calculée selon la formule :

$$N = 1 + E_x + A_x + C_x + D_x + S_x$$

où E_x, A_x, C_x, D_x *et* S_x correspondent à l'évaluation de votre langage selon chaque critère.

4 Rendu

Le rendu du devoir comprend au minimum trois fichiers tels que mentionnés dans ce document, c'est-à-dire *lexer.x*, *parser.y* et *cheatsheet.md*. Les fichiers en question doivent inclure un entête sous forme de commentaire indiquant le(s) auteur(s) du langage. Libre à vous de travailler seul ou en binôme. Le livrable doit être le résultat de votre propre production. Le plagiat, de quelque façon que ce soit et quelle qu'en soit la source, sera considéré comme de la tricherie et dénoncé en conséquence. Sauf avis contraire, il ne vous est pas permis d'utiliser des fonctionnalités du langage Haskell qui n'ont pas été présentées en cours. La date de rendu pour ce devoir est fixée au **mardi 16 mai 2023 à 10h25**. Aucun retard ne sera admis et la note de 2 vous sera automatiquement attribuée si cela devait se produire.

5 Conclusion

Ce cours est votre première expérience avec la conception de langages de programmation. Il est donc important pour vous de commencer l'implémentation relativement tôt. À vous d'y investir le temps que vous jugerez nécessaire. Le forum de discussions est à votre entière disposition pour poser des questions et demander des conseils sur le devoir. Toutefois, ce forum ne doit pas devenir le lieu pour déboguer votre code. En cas de problème, il est de votre responsabilité d'identifier la source du bug et de le corriger. À cet égard, les outils de debugging sont vos meilleurs alliés.

Bon travail !