Elliott FALGUEROLLE / Antonin JEAN Gr 4 : L3 IGM UGE

Analyseur Syntaxique : Rapport

Tout d'abord, le cours d'analyse syntaxique est un cours complexe. Celui-ci mêle de nouvelles technologies auxquelles nous n'avions jamais touché : les outils Flex et Bison. A des éléments théoriques complexes tels que les automates, les grammaires… Notre travail pour ce projet a donc commencé en TP où il nous à fallu bien comprendre et s'exercer sur les différents aspects requis à la création d'un analyseur syntaxique.

Ensuite, il s'est agit de bien comprendre le sujet, et comment tous les composants décrits dans celui-ci s'emboitent. Par exemple, la façon dont les terminaux du lexer vont être utilisés dans la grammaire. Ou encore, quelles sont les spécificités du langage que l'on veut analyser, en quoi ressemble et diffère-t-il du C ?
Tout cela afin de réaliser un projet conforme à ce qui est décrit dans le sujet, et réaliser de la manière la plus propre.

Pendant la réalisation plusieurs éléments ont pu nous poser problème.

Tout d'abord, la création de l'arbre abstrait à été compliquée. Nous comprenions bien le code fourni et comment l'utiliser, le problème à été de savoir quels nœuds utiliser comme père ou comme fils. Nous avions aussi cru comprendre que l'arbre ne devait avoir aucun non terminal, ce qui le rendait bien moins facile à lire.
Mais nous avons ensuite pu bien comprendre quels non terminaux il était judicieux d'utiliser, et quelle hiérarchie appliquer aux différents nœuds, tout ceci dans un souci de réaliser un arbre propre et facile à lire.

Ensuite, nous n'avions pas tous les deux déjà utilisé getopt, et n'avions jamais implémenté des options longues. Il nous a fallu bien apprendre l'outil getopt pour pouvoir implémenter les options requises par le sujet.

Enfin, il à été compliqué de réaliser l'implémentation du switch. Celui-ci ressemble, mais est différent de celui du C dans certains aspects. Il nous a fallu d'abord bien comprendre le fonctionnement du switch en C, pour connaître les fonctionnalités à implémenter. Et ensuite bien prendre en compte les différences apportées par celui du sujet.
La création des règles de grammaire du switch n'a aussi pas été tâche facile à entreprendre. Mais nous avons pu remarquer qu'en nous inspirant de la structure des règles déjà présentent, il était plutôt simple de créer de nouvelles règles pour le switch. Par exemple, pour une fonction on a la règle :
DeclFonct: EnTeteFonct Corps
Pour un switch on pourrait alors avoir :
Switch: EnTeteSwitch CorpsSwitch