Rapport de projet: analyseur syntaxique

Analyse syntaxique (L3)

ATTIG Léo & SAUTIER Sébastien

Sommaire

I. Choix spécifique

Le premier choix qu'ils nous a été donner de faire a été l'implémentation de la nouvelle règle de grammaire (déclaration + assignation).
Étant donnée qu'il ne faillait pas pouvoir déclarer et assigné des variable global, nous avons fais le choix de séparer complètement toutes les règles de déclaration de variable global et local. Nous avons donc deux règle principale: **DeclGlobalVars** et **DeclVars**.
Ces deux règles mènent a des « sous règles » différente, qui permettent la déclaration/ assignation pour **DeclVars**.

Le deuxième réelle choix que nous avons du opérer c'est la présentation de l'arbre abstrait. Nous avons choisi de partir du module tree.c fournie dans les TP s'analyse, puis bous l'avons ajusté pour qu'il corresponde a ce que nous voulions faire.
Pour la répésaétion de l'arbre nous Avon choisi d'indiquer chaque instruction comme un noeud de l'arbre et si jamais c'est instruction nécessite une précision (par exemple: nom d'une variable ou type de cette dernière) on affiche ces précision sur la même ligne.
Les blocs « {} » sont représenter le fait que les instructions sont fils de la déclaration d'une fonction ou bien d'un if par exemple.
Les structure de contrôle, sont représenter par un noeud avec comme premier fils la condition, puis le code a exécuter dans le cas ou la condition est validé et dans le cas d'un « if » il peut y avoir un troisième fils correspondant au « else »

Le dernier choix a été la manière dont nous associons des information spécifique une instruction (variable, constante, etc). Pour cela nous avons ajouter a l'union de la grammaire deux champ: **char* name** et **int const_val**.
name -> correspond a tout les noms que l'on puisse trouver dans le fichier, comme les noms de variable ou de fonction.
const_val -> correspond a toutes les valeurs de constante que l'on peut trouver en TP-C, étant donnée que les seuls constante peuvent être des char ou des int, un champ de type int suffis.
Ensuite du côté de tree.c on stock ces variable dans des champs de la structure Node:
- const_val
- name
- type

Ces champs servent aussi bien pour une variable que pour une fonction (type de retour et nom).
Pour l'affichage on regarde seulement qu'elle est la valeur « label » du node, et on affiche les information supplémentaire nécéssaire.

II. Difficultés

Heureusement pour nous, nous n'avons rencontrer aucune difficulté notable, toutes les notions on été comprise rapidement et facilement. Et toutes les choix on été mis en place sans bug incompréhensible ou mystique.

Le seul moment qui nous a demander de la réflexion c'était la manière dont nous allions présenter notre arbre abstrait.